Practice Based Optimisation of Bus-Train Timetable Coordination

by

Rejitha Nath Ravindra
M.Tech Transportation Engineering

A thesis submitted for the degree of Doctor of Philosophy at
Monash University in April 2021

Public Transport Research Group
Department of Data Science & Artificial Intelligence (DS&AI)
Faculty of Information Technology

Supervisors:
Prof. Dr. Mark Wallace
Prof. Dr. Graham Currie
Dr. Daniel Harabor
Dr. Ilankaikone Senthooran
Mr. Chris Loader

To Mom and Dad

Contents

List of Tables

List of Figures

List of Algorithms

List of Abbreviations

BB	Branch and Bound
BTCP	Bus-train Timetable Coordination Problem
CG	Column Generation
CP	Constraint Programming
DoT	Department of Transport
DR	Driver Rostering
DSS	Decision Support System
DS	Driver Scheduling
FS	Frequency Setting
GA	Genetic Algorithm
I-TTVS	Integrated Timetabling and Vehicle Scheduling
IVTT	In-Vehicle Travel Time
LNS	Large Neighbourhood Search
LS	Local Search
MDVSP	Multiple-Depot Vehicle Scheduling Problem
MINLP	Mixed-Integer Non-Linear Program
MIP	Mixed-Integer Program
ND	Network Design
NP	Non-deterministic Polynomial-time
OVTT	Out-Vehicle Travel Time
PESP	Periodic Event Scheduling Problem
PTN	Public Transport Network
PT	Public Transport
QSAP	Quadratic Semi-Assignment Problem
RTC	Real-Time Control

SA Simulated Annealing

SD-TTVS Sequential-Decomposed Timetabling and Vehicle Scheduling

SDVSP Single-Depot Vehicle Scheduling Problem

SQR Synchronisation Quality Ratio

TO Transfer Optimisation

TS Tabu Search

TTS Timed-Transfer System

TT Timetabling

VRP-TW Vehicle Routing problem with Time Windows

VS Vehicle-Scheduling

Practice Based Optimisation of Bus-Train Timetable Coordination

Rejitha Nath Ravindra
rejitha.ravindra@monash.edu
Monash University, 2021

Supervisor: Prof. Dr. Mark Wallace
mark.wallace@monash.edu
Associate Supervisors: Prof. Dr. Graham Currie
Dr. Daniel Harabor
Dr. Ilankaikone Senthooran
Mr. Chris Loader

Abstract

In Public Transport (PT) planning and operations, transfers play a vital role in efficiently connecting origins and destinations that are not otherwise connected by a direct service. Convenient transfers are a direct consequence of well-coordinated timetables that minimise passenger transfer waiting time between multiple public transport services. However, solving a real-world timetable coordination problem is extremely complex as it involves mixed and conflicting objectives and constraints. While the users desire efficient services, ideal waiting times and convenient transfers, the operators aim to minimise the resource costs. Moreover, a general planning framework observed in current practices is sequential and linear, wherein the network timetables are created first and cost-minimal vehicle schedules are derived next. Such segregation simplifies the problem but affects the overall quality of timetables, and a favourable compromise between user and operator requirements is not gained.

This research aims to achieve the simultaneous optimisation of timetable coordination and vehicle cost-efficiency with minimal adverse impact on the service quality. In this context, the original contributions to knowledge emerging from this thesis include useful mathematical models and solution methods to achieve cost-effective timetable coordination. The focus of this research is the case of bus-train transfers in the *City of Wyndham* in south-western Melbourne, Australia. This thesis contains four parts.

The first part examines the relevant background and theory underpinning public transport planning stages and identifies the key limitations with modelling and solving techniques for the timetable coordination problem. Existing approaches often consider simplified instances of timetable coordination and lack the flexibility needed to solve the problem holistically. Findings suggest that identifying operable and practical constraints is critical in generating scheduling solutions that are cost-effective and implementable in the real-world. In addition, the necessity of an integrated algorithm to solve the complete problem of timetable coordination is also explored here.

The second part presents an optimisation framework that comprise a comprehensive mathematical model for timetable coordination incorporating industry favoured and practical scheduling requirements. To reflect time-of-day dependent variability in bus arrivals, certain degree of flexibility is allowed to the operational constraints. Findings indicate that the bus-train timetable coordination problem can be prohibitively large for standard optimisation techniques to find good-quality solutions, at reasonable computational time. This acts as a scaffold for the proposal of two solving approaches based on (i) decomposition and (ii) heuristics.

Several decomposition strategies are investigated first such that it ensure maximum compatibility between network timetabling and vehicle scheduling. Results indicate feasible and good-quality scheduling solutions at reasonable computational time. However, there are limitations with assessing the quality of these solutions in terms of closeness to true optimum values. This led to the necessity to investigate the role of heuristics in yielding better solutions for the coordination problem.

An integrated approach is then proposed to model the network timetabling and vehicle scheduling problems simultaneously. To tackle large network instances of the problem, a *Large Neighbourhood Search* (LNS) meta-heuristic scheme is designed. Findings show significant gains in the overall solution quality, thus indicating that even the best decomposition strategy can be improved further using heuristics.

The third part details the benefits of improved timetable coordination in terms of passenger service and bus resource requirement in the case study area. Findings indicate that the developed optimisation framework compares well with current commercial state of the practice by achieving a favourable trade-off between timetable coordination and vehicle cost efficiency.

The final part synthesises the results and findings from this thesis. In summary, this research enables an understanding of the challenges involved with solving the timetable coordination problem and explores the limitation with current practices that are not yet perfectly linked to the real-world scheduling requirements. The key findings from this research support the hypothesis that incorporating some flexibility to the scheduling constraints can significantly minimise transfer passenger waiting times and vehicle requirements. The developed optimisation framework can act as a decision making aid for planners to choose multiple, problem-specific service preferences such that comprehensive and integrated timetabling solutions are arrived at. In a bigger picture, by offering robust solutions to the real concerns of users and schedulers alike, this research looks at bridging the gap between scheduling in principle and scheduling in practice.

Practice Based Optimisation of Bus-Train Timetable Coordination

Declaration

This thesis contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

Rejitha Nath Ravindra
April 24, 2021

Acknowledgments

First and foremost, this research would not have been possible if not for my amazing supervisory team of Professor Mark Wallace, Professor Graham Currie, Dr. Ilankaikone Senthooran, Dr. Daniel Harabor and Chris Loader. I would like to extend my deepest gratitude for this opportunity. Thanks Mark, for your relentless support throughout my candidature. Your kindness and encouragement kept me motivated especially during the initial stages of this research when I found it challenging to acquaint myself with the research questions and methodology. In addition to the guidance you have provided with the practical, technical and theoretical aspects of this research, which has been instrumental, I appreciate the freedom you have provided me to think independently as a researcher. The value that your supervision has added to my research path is immeasurable. Thanks Graham, for always providing clarity and structure to my research ideas, questions and for being so supportive and patient throughout. As the other members of the SEPT-GRIP team would agree, your involvement in fostering an active research community is highly valued. Thank you, Chris, for demonstrating the value of research translation to industry practices. Your efforts in providing a consistent and realistic context to the practical challenges with timetable coordination has been significant. I immensely admire your passion for transport and look forward to learning more, as I transition to the industry. Thank you Senthoo, for your patience and guidance with the core modelling practices in MiniZinc. Your timely intervention has helped me save days or weeks of de-bugging, and I am grateful for the practical insights you provide with this research as well as my role as a researcher. Thanks Daniel, for always helping me see the bigger picture of this research with utmost clarity. I appreciate your encouragement that often helped me venture into areas out of my comfort zone.

This research is one of the 18 projects in the Sustainable and Effective Public Transport Graduate Research Industry Partnership (SEPT-GRIP) at the Public Transport Research Group (PTRG), which belongs to the Institute of Transport Studies (ITS) in the Department of Civil Engineering at Monash University. The SEPT-GRIP is a dynamic inter-disciplinary team of academics, industry practitioners and numerous others who engage extensively in research on public transport systems, users, planning and policy. My PhD journey would not have been the same without the kind of stimulating work environment we have at SEPT-GRIP. Being an international student, one of the factors that helped me settle in smoothly was the diversity within our team. Thank you for making my time in Monash more memorable.

This project was jointly funded by Monash University and the Department of Transport (DoT), Victoria (erstwhile the Department of Economic Development, Jobs, Transport and Resources) who are also the industry sponsors for this research. In an unexpected turn of events during the onset of the COVID-19 pandemic, Monash University also provided funding through the Graduate Research Completion Award, which helped towards the finalisation of this thesis. I also received the Supplementary Funding Scheme from the Monash Faculty of Information technology that facilitated the dissemination of this research at the Transportation Research Board (TRB) Annual Meeting (TRB-2020), Washington D.C. I wish to extend my sincere gratitude for the funding support from all the organisations and departments involved during my candidature.

Remarking truly on the interdisciplinarity of this research, various members of the Faculty of Information Technology and the Department of Civil Engineering have extended their assistance in this research. Special thanks to my panel members Dr. Chris De Gruyter (now of RMIT University) and Dr. Guido Tack for providing me valuable feedback at each research milestone.

I was fortunate to have received multiple opportunities to engage with the industry, as early as the first year of this research. I extend my sincere thanks to Phillip Boyle of PBA Transit Pty Ltd, Melbourne for offering me a part-time employment opportunity to work in the areas of transport planning and scheduling. I also thank Chris and Graham for creating the SEPT-GRIP PhD professional development program with the DoT where I undertook projects regarding network-wide timetable coordination in Melbourne that also aided this research. During my engagements with both companies, I gathered working knowledge of current commercial scheduling tools that further deepened my understanding of the practical challenges schedulers face on a day-to-day basis. My sincere thanks to Dr. Charles Fleurent, Senior Director (Optimization Algorithms) at GIRO, Canada for his valuable insights into the developments in optimisation algorithms in the GIRO-HASTUS software. I also thank the experts at PBA Transit and the DoT for providing me training and timely assistance with the technical aspects of HASTUS, especially the *NetPlan* module.

Lots of learning and personal growth over the past years is attributed to professional development activities that I undertook over these years and a special thanks goes to Monash Graduate Research for facilitating these. I also extend my sincere gratitude to Julie Holden for her assistance with research communication and dissemination. Her expert advice on translating complex ideas into concise yet meaningful sentences have helped me adapt well with the target audience. A debt of thanks is owed to the administrative staff at Caulfield School of IT, who have looked after my candidature requirements so well. In particular, I would like to thank Allison, May, Sheena and Rachel who have always been available to take notice of my queries.

I am immensely grateful for my pillars of strength- my family. Amma and Acha, words cannot describe how thankful I am for all your unconditional love and sacrifices throughout, and for persistently supporting me throughout my PhD journey even on the days I fell apart. You always gave me the liberty to learn, explore and flourish to my heart's content and I count myself blessed. Rakesh and Anju, thanks for always being there for me, and most importantly for bringing Medha into our lives. She makes every day of mine more meaningful and I take pride in being her aunt.

Finally, Gopi, I am incredibly lucky to have met you during my PhD journey. You walked with me through thick and thin, and none of this would have been possible without your love, care and patience for the past five years. Thank you, for your faith, understanding and constant reminder for the worth in what I do. I cannot wait to start the next exciting phase of our life together.

Rejitha Nath Ravindra

Monash University
April 2021

Part I

Background

Chapter 1

Introduction

1.1 Overview

This thesis investigates how the utility of public transportation can be maximised by designing inter-modal timetables that are well-coordinated. It aims to achieve the simultaneous optimisation of timetable coordination and vehicle cost-efficiency with minimal adverse impact on the service quality. In this context, this research develops a comprehensive optimisation framework for the timetable coordination between buses and trains, called the *Bus-train Timetable Coordination Problem* (BTCP) and propose state-of-the-art optimisation techniques to solve the problem. The focus of this research is the case of public transport in the City of Wyndham in south-western Melbourne, Australia.

This chapter introduces the necessary background and motivation behind this research in Section 1.2, followed by the main research question and the associated research objectives in Section 1.3. Section 1.4 then describes the research scope in the context of public transport in Melbourne. Section 1.5 details the contributions to knowledge emerging from this research. This chapter concludes with an outline on the thesis structure in Section 1.6.

1.2 Background and Motivation

In postmodern societies with excessive car ownership, it is a proven challenge to attract ridership to public transport (PT). Disharmony between inter-modal PT services compels the commuters to shift their means of patronage to private transport, thereby contributing to increased congestion and pollution. Transfers between inter-modal public transport (PT) systems are a necessity in providing seamless transport opportunities to passengers in connecting the broadest of origins and destinations that are otherwise not connected by a direct service (Mees, 2000). As observed in Walker (2012), although one could design direct networks with minimum number of transfers on a journey, connected networks involving passenger transfers at a strategic point (say train stations) can enable faster commute due to the higher frequencies it can offer. Thus, facilitating efficient transfers is a key factor in improving the service performance of a transit system. However, the undesirable wait times imposed on passengers transferring between multiple services (like buses and trains) is a major disincentive to the use of public transport as compared to the non-transfer alternatives like cars (Guo, 2003).

Efficient transfers are a direct consequence of well-coordinated timetables. *Timetable Coordination* is an efficient transit operational strategy that ensures temporal harmony between multiple public transport services by minimising the undesirable transfer wait times between them. A realistic problem of timetable coordination is a complex optimisation problem due to the conflicting objectives and constraints between the connection quality and operator cost minimisation (Desaulniers and Hickman, 2007; Voß, 1992). Each individual schedule is subject to its own customer service requirements and cost constraints. The users desire efficient services, ideal waiting times and convenient transfers, and the operators aim at minimising cost and resources. While devising this problem on a small network is less difficult, achieving a trade-off between such conflicting requirements is challenging in larger network instances (Currie and Bromley, 2005). Moreover, a network-wide PT system has multiple transfer points, and coordination at one point impacts the others. The resulting problem is combinatorial, with a search space of candidate solutions that grows exponentially with the size of the problem. Solving this problem soon reaches the limits of manual handling (Ceder et al., 2001; Klemt and Stemme, 1988).

The diversity with formulating and solving the timetable coordination problem lies in the underlying compromises, model requirements and varying input parameters. Most methods found in literature often consider unrealistic simplifying assumptions (Poorjafari and Yue, 2013; Schuele et al., 2009) that make the solutions inefficient and reduce overall PT reliability. The most common simplifying assumptions in timetabling and vehicle scheduling include the practice of discarding time-of-day dependent variability with service frequencies, run-time and transfer volume; exclusion of unproductive service time like layovers[1] and deadheads[2], that impact the bus operating cost significantly.

Despite the intrinsic challenges with solving the timetable coordination problem, it is conventionally addressed with a certain degree of intuition and solved using a combination of simplified automated methods and manual adjustments that do not always guarantee comprehensive solutions (Ceder et al., 2001). Although existing techniques can help prioritise the trade-off between user and operator requirements to an extent, these do not always guarantee optimal solutions in a multiple route, multi-operator planning scenario like in most metropolitan cities. A general framework observed in current practices is also sequential and linear, where the network timetables are created first (called *Timetabling*, abbr. TT) and cost-minimal vehicle schedules are derived next (called *Vehicle Scheduling*, abbr. VS) (Guihaire and Hao, 2008*a*). Such segregation simplifies the problem but renders the schedule inefficient as these do not consider the inter-dependence between planning sub-problems such as timetabling and vehicle scheduling, and thus cannot establish a balance between high-quality timetables and low-cost vehicle schedules simultaneously.

[1]The time between multiple bus trips, usually to allow for some recovery time or driver meal breaks

[2]The time that a bus travels without any passengers; typically from and to the depots, or when bus operators need to travel inter-route

Herein lies the need to develop a comprehensive scheduling tool that can simultaneously optimise the objectives of timetable coordination and vehicle optimisation.

The key research gaps emerging from this research are presented in more detail in Chapter 2. Firstly, there are limitations with current practices in solving the timetable coordination problem that are not yet perfectly linked to the realistic requirements. A comprehensive set of planning and operational constraints must be identified that has the most significant impact on decision making. Secondly, there is a need to investigate how the coordination problem can be modelled comprehensively and solved to optimality using state-of-the-art techniques. These gaps will be addressed in the upcoming chapters.

1.3 Research Questions

This thesis will investigate the following research question:

Can we generate timetables and vehicle schedules that optimise both timetable coordination and operator cost efficiency simultaneously?

To answer this question, we identify four research objectives (RO) and associated sub-questions as follows:

RO_1 : To understand the existing challenges with public transport timetable coordination.

- (1a) What are the limitations with existing approaches in solving the timetable coordination problem?
- (1b) What modelling constraints represent the problem realistically?

RO_2 : To formulate the timetable coordination problem incorporating real-world scheduling constraints.

RO_3 : To solve the timetable coordination problem using state-of-the-art optimisation techniques.

RO_4 : To evaluate the quality of optimised bus-train timetables.

- (4a) What are the benefits of optimised timetable coordination in terms of passenger service and operational cost?
- (4b) How does the developed optimisation framework compare with current commercial practices?

(RO_1) reviews the existing knowledge on the complexities involved with solving the timetable coordination problem and past compromises in models and solution approaches. Inferring from literature and industry discussions, it also focuses on identifying a range of constraints that model the problem realistically.

(RO_2) aims to mathematically formulate the coordination problem. It seeks to translate the most relevant operational constraints identified in (RO_1) into mathematical formulations.

(RO_3) aims to solve the formulated model with multiple, constraint based optimisation techniques that render good-quality solutions for large-scale problems at acceptable computational time.

(RO_4) analyses the quality of the optimised scheduling solutions in terms of passenger and operational benefits. It also evaluates the performance of the developed optimisation framework in comparison to the current commercial practices.

1.4 Scope and Melbourne context

This thesis results from a collaboration between academic researchers and transport planners, as part of the Sustainable and Effective Public Transport- Graduate Research Industry Partnership (SEPT-GRIP). Drawing from requirements prioritized by our industry partners at the Department of Transport (DoT)-Victoria, in Melbourne, Australia, this thesis predominantly studies the case of bus-train transfers in the City of Wyndham in south-western Melbourne, Australia (Figure 1.1).

Observed unevenness between bus and train frequencies at different time of the day (peak/off-peak), type of the day (weekdays/weekends), location and route direction makes PT timetable coordination a highly challenging task in a dynamic city like Melbourne. Most bus routes in the City of Wyndham have an existing average bus frequency of 20-40 minutes during weekday AM and inter-peak, targeted to meet passenger demand for many different travel purposes. Train services exhibit wide variation in Weekday AM-peak, and the inter-peak services mostly include 20 minute frequencies.

The major contributor to operator costs are the fleet size (dictated by the required number of buses dispatched at the busiest time of the day) and associated driver costs corresponding to the bus hours served (including unproductive time like deadheads and layover). With uneven bus and train frequencies at any given time of the day, achieving efficient timetable coordination at a reasonable aggregate (user and operator) cost in Wyndham is a laborious task. With cost minimisation as one of the major objectives in this research and with industry provided real-world scheduling requirements, the optimisation technology that we propose through this research is intended to capture the dynamic nature of public transport vehicle interactions in Melbourne. Since this research also has a strong industry focus, it looks at bridging the gap between scheduling in principle and scheduling in practice by proposing a decision-making tool to aid the transit agencies in realising accurate, realistic and cost-effective solutions for timetable coordination.

1.5 Contributions to Knowledge

In line with the key research gaps and scope, the contributions to knowledge emerging from this thesis are as follows:

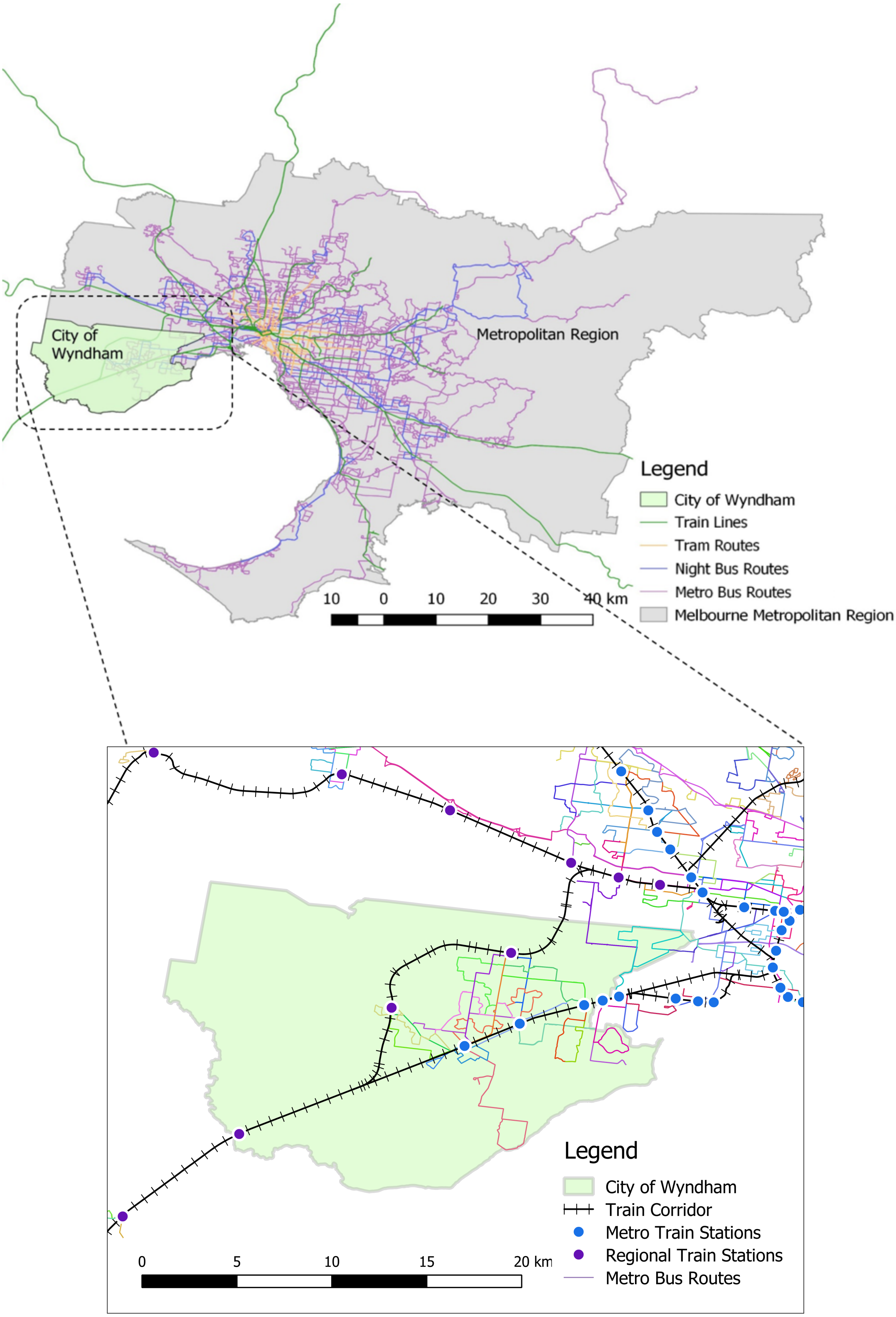


Figure 1.1: Melbourne Public Transport Network (Inset: City of Wyndham)

C_1: A Comprehensive Mathematical Model

A comprehensive mathematical model for timetable coordination based on a set of real world requirements (RO_1 and RO_2) is designed. In particular, we investigate various constraints that can represent the timetable coordination problem realistically, including those scheduling requirements that are not yet considered in the literature before. The overall objective of this model is to achieve a favourable trade-off between the contrasting objectives of improving bus-train timetable coordination and reducing operator costs, subject to a variety of operational and practical requirements.

C_2: A Two-stage Sequential-Decomposed Approach

A two-stage sequential and decomposed planning approach with solvable sub-problems for timetable coordination is presented (RO_2). The developed models perform sequential decomposition of the timetabling and vehicle scheduling sub-problems different from the traditional planning approach in such a way that it renders compatible solutions. The performance of an integer-linear solver in yielding proof of optimality is then observed (RO_3). The model is tested for scalability and solution quality on a subset of a PT network in the City of Wyndham.

C_3: A Meta-heuristic based Integrated Approach

An integrated timetabling and vehicle scheduling model is developed to solve the timetable coordination problem (RO_2). A *Large Neighbourhood Search* (LNS) meta-heuristic scheme is proposed to generate improved scheduling solutions for large-scale network instances (RO_3). The model is tested for scalability and solution quality on a subset of a PT network in the City of Wyndham.

1.6 Thesis structure

This thesis is structured into 4 parts with 8 chapters including this Introduction. Figure 1.2 presents the overall structure of this thesis.

Part I: *Background* establishes a context for this research, outlining the relevant theory and background concerned with the timetable coordination problem. It includes three chapters starting with this Introduction:

> **Chapter 2: Literature Review** presents an overview on the problems in public transport planning, with emphasis on timetable coordination. A comparison is drawn between the efficiencies of traditional sequential planning approaches vs integrated approaches. The chapter also outlines the difficulty involved with finding global solutions to timetable coordination problems, with a brief review on the performance of existing solution methodology. This is followed by highlighting the efficiency of constraint based optimisation in solving combinatorial problems such as timetable coordination.

Chapter 3: Research Methodology provides an overview of the thesis research methodology and the approaches adopted to address the key research gaps and opportunities.

Part II: *Optimisation Framework* consists of three chapters detailing the proposed mathematical models and optimisation techniques for the timetable coordination problem:

Chapter 4: Comprehensive Mathematical Model presents contribution C_1, that is, a mathematical model for the bus-train timetable coordination problem using the detailed scheduling requirements. The most relevant constraints, variables and objectives are formulated mathematically. The chapter also highlights the relevant model extensions to solve additional scheduling requirements.

Chapter 5: Sequential-Decomposed Approach for Timetable Coordination presents contribution C_2, that is, a two-stage sequential modelling approach with solvable sub-problems of timetabling and vehicle scheduling. Along with multiple decomposition strategies to solve the timetable coordination problem, this chapter outlines the associated solutions approach, numerical experiments, case studies and results.

Chapter 6: Integrated Approach for Timetable Coordination presents contribution C_3, that is, a *Large Neighbourhood Search* based meta-heuristic approach for the timetable coordination problem with integrated timetabling and vehicle scheduling. This chapter presents the associated solutions approach, numerical experiments, case studies and results.

Part III: *Timetable Evaluation* presents one chapter that discusses the case studies, solution evaluation and its implication on real-world public transport:

Chapter 7: Evaluation of Optimised Timetables details the application of the proposed optimisation framework in a real world public transport network in the City of Wyndham, Melbourne. A comparison of approaches introduced in chapters 5 and 6 is made. Through several case studies, this chapter demonstrates how the developed models and algorithm can be effective in achieving optimised coordination and cost efficiency. This chapter then evaluates the quality of timetable solutions using an existing commercial scheduling tool.

Part IV: *Conclusions* includes the final chapter from this thesis:

Chapter 8: Discussion and Conclusions synthesises the key results and contributions achieved to demonstrate how the designed research aims and objectives have been met in this thesis. This chapter also outlines the key research implications, limitations, model transferability and directions for future work.

* * *

This concludes Chapter 1: Introduction, which has provided a brief context for this research outlining its overall aim and objectives, scope and relevant contributions to knowledge. The next chapter, Chapter 2: Literature Review, outlines a theoretical background to this research by providing a detailed review on public transport planning problems with an emphasis on timetable coordination. It also highlights the relevant gaps in knowledge targeted in this research.

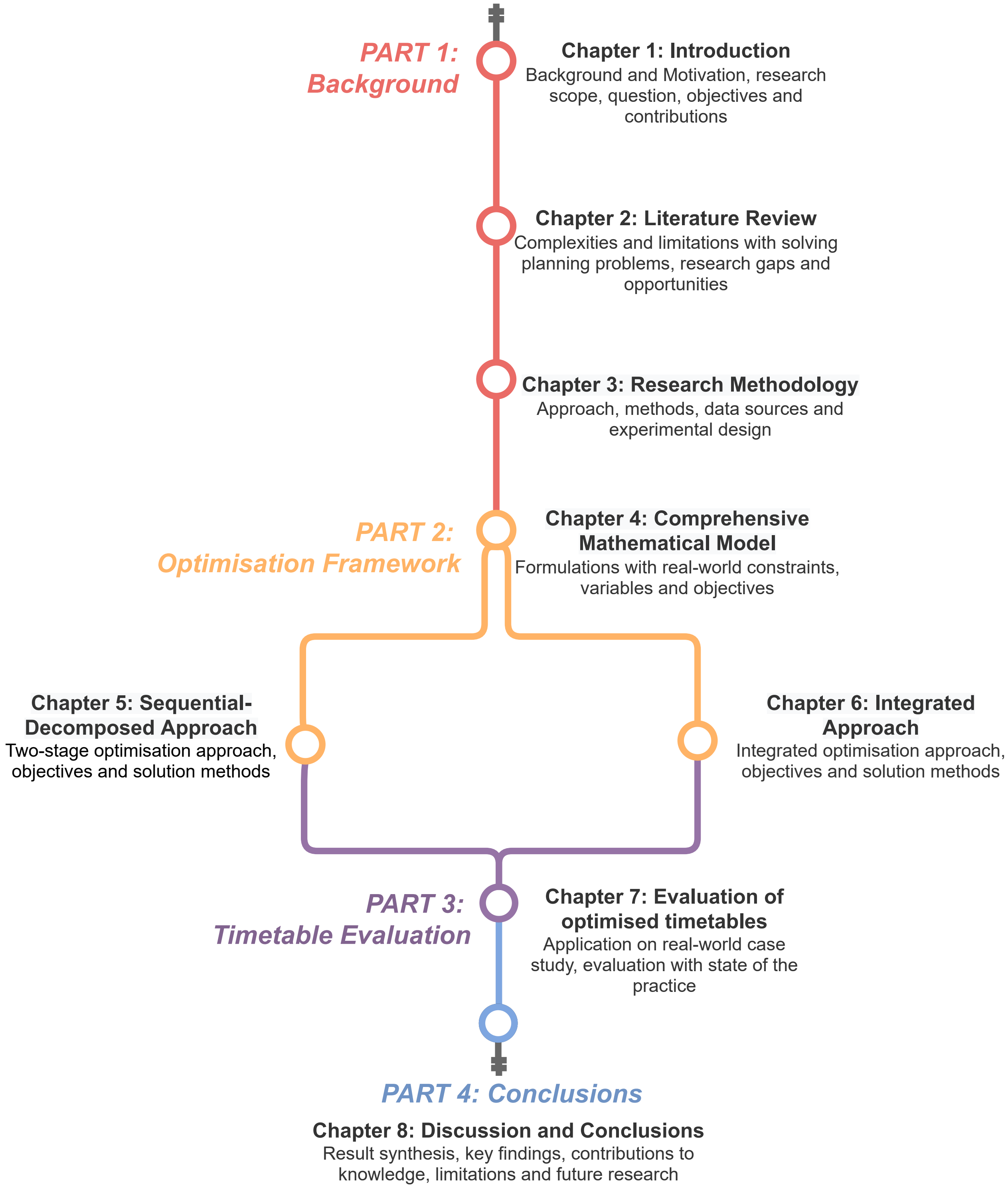


Figure 1.2: Thesis Structure

Chapter 2

Literature Review

2.1 Introduction

In this chapter we present an introduction to the fundamental concepts of public transport (PT) planning and operations required to understand the major contributions from this thesis. We begin by explaining the various levels of public transport planning process and the associated sub-problems in Section 2.2. We lay our focus on the problem of timetable coordination, detail the most relevant modelling and solving approaches available in literature, followed by an account on the limitations in these approaches in yielding holistic and practical scheduling solutions in Section 2.3. In Section 2.4, we explore the ability of constraint based optimisation techniques in solving complex combinatorial problems like timetable coordination. We then review the conventional state of the art and state of the practice in addressing planning problems sequentially in Section 2.5 and explain the rationale behind integrating these problems for optimal timetable coordination solutions in Section 2.6. This chapter concludes with a brief review on analysing the quality of optimised planning solutions from a practical perspective in Section 2.7.

2.2 Public Transport Planning process

The planning, operations and control of a public transit system is very complex, given the involvement of several actors like the transit agencies, users and operators. Such actors are also subject to individual preferences and constraints, and interact with each other based on a set of relationships. Transit agencies focus on providing good quality services at low passenger fare and thus have an overarching aim of attracting motorised passengers to public transport, contributing to reduced pollution and traffic congestion. However, due to budgetary constraints, the agencies must also maintain the efficient use of limited resources such as buses, labour and infrastructural facilities. From an urban context, the decisions made by users, non-users and operators are thus conflicting and dynamic.

In its entirety, the global problem of public transit planning that maintains a good balance between users' level of service and operator costs is not tractable. Due to its complex nature, the planning problem is commonly divided into the following set of sub-problems that span through four stages namely *strategic*, *tactical*, *operational* and *real-time control* (Ceder, 2016; Desaulniers and Hickman, 2007):

- Network Design (ND): A *strategic* planning problem that defines the public transit network (PTN) comprising a set of time-points (corresponding to stations or stops), and a set of edges (corresponding to direct connections between stations or stops). Subsequently, routes are defined by a sequence of time-points that are pair-wise connected by edges.
- Frequency Setting (FS): A *tactical* planning problem that determines a desired route frequencies to meet varying passenger demand at different periods of planning (that is, demand at AM-peak, Inter-peak, PM-peak etc) without exceeding a given limit of vehicle capacity.
- Timetabling (TT): The subsequent *tactical* planning problem that converts the desired frequency of service on each route into a schedule. Arrival and departure times of a vehicle at major time-points are defined to meet multiple objectives such as: meet desired frequencies, satisfy dynamic demand patterns, maximise synchronisation and minimise transfer wait times.
- Vehicle Scheduling or Rolling Stock Planning (VS): An *operational* planning problem that assigns the timetabled trips into vehicle schedules (for buses) or rolling stock (for trains) such that the costs associated with the acquisition and operation of the vehicle are minimised.
- Driver Scheduling (DS): The second problem in *operational planning* that focuses on assigning drivers to cover all the scheduled vehicles in a planning period. Generic daily duties that satisfy labour regulations (minimum/maximum work hours, maximum work duration without break, intervals between meal breaks etc.) are created such that the duty wages are minimised.
- Driver Rostering (DR): The subsequent *operational* planning problem that assigns the daily duties to available drivers over a period of time, say one month. The work schedules of drivers, called *rosters* are prioritised such that labour regulations are met.
- Real-Time Control (RTC): This stage devises strategies to deal with scenarios where various exogenous and endogenous factors (such as weather, traffic incidents, equipment breakdown etc) affect the service delivery of PT systems. The real-time control strategies to maintain normalcy in transit include station control, vehicle holding and stop-skipping.

Figure 2.1 shows the four stages of PT planning and the interdependence between its corresponding sub-problems. Conventionally, the solutions for upper-level sub-problems serve as inputs for lower-level sub-problems, thus following a sequential pattern. However in reality, as Ceder (2016) indicates, such a hierarchical approach is undesirable as decisions made on the lower-level sub-problems will influence those on the upper-level. Hence, using the locally "optimal" output from one of the sub-problems as inputs for consecutive sub-problems will not result in a global optimal solution. Finding a global solution to planning problems is an extremely challenging task as the network size and vehicle fleet increases, and motivates the use of advanced models and algorithms.

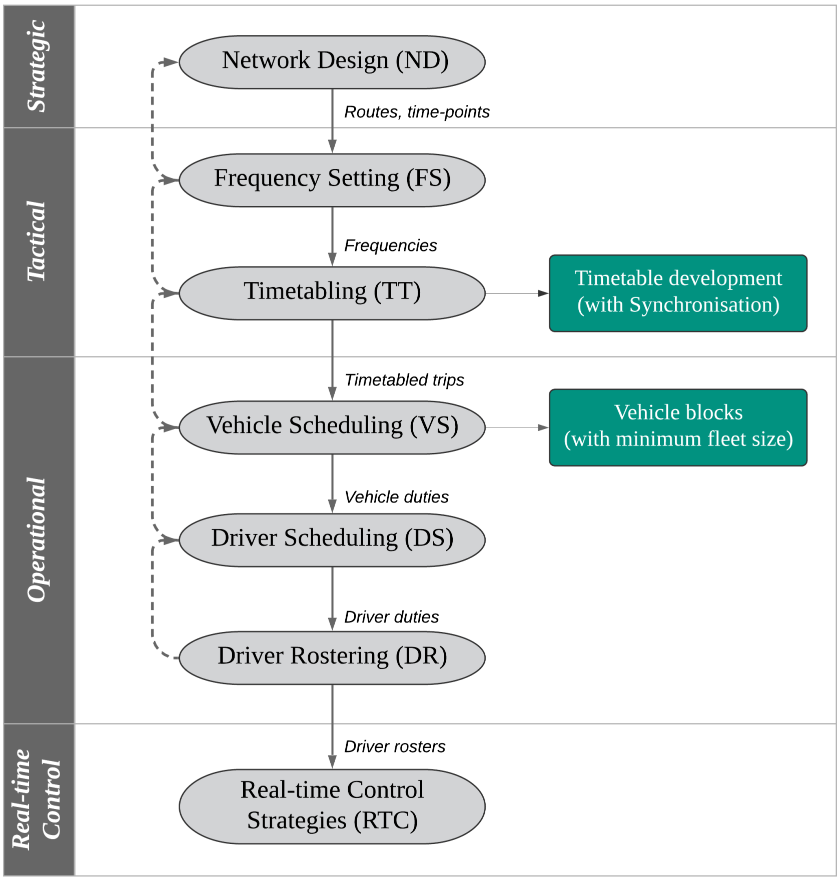


Figure 2.1: Stages and corresponding sub-problems of the public transport planning process (*Source: Author's adaptation from Ibarra-Rojas et al. (2015) and Desaulniers and Hickman (2007)*)

Advanced models and solution techniques have been researched for decades for all planning problems. An exhaustive review on all public transport planning processes is beyond the scope of this thesis. In the following sub-sections 2.2.1 and 2.2.2, we concentrate on surveying the *timetabling* and *vehicle scheduling* sub-problems respectively, that are core to this thesis. This review, hence, does not cover the upper-level problems of *network design* and *frequency setting*. Also, we leave aside the downstream problems of *driver scheduling and rostering* and *real-time control*. We refer Desaulniers and Hickman (2007) for a general review on the aforementioned transit planning stages. Guihaire and Hao (2008*a*) gives a focused review on the strategic and tactical planning stages. For a detailed and updated review on bus transit planning, operations and control with a special emphasis on timetabling and vehicle scheduling, we refer to Ibarra-Rojas et al. (2015).

2.2.1 Timetabling

The timetabling problem is a *tactical planning* component and the first decision made on converting desired service frequencies on a fixed route into schedules, i.e. the arrival and departure times for a set of trips in that route. It takes as input the network layout including the routes, time-points (stations or stops), running time (ideally including some layover time[1]) and frequencies set in the strategic and tactical planning stages prior. The objectives of timetabling arise from the need to satisfy various operational characteristics (Ibarra-Rojas et al., 2015) and are generally categorised as follows:

- satisfy time-of-day dependent passenger demand flows
- satisfy desired target headway bounds and maintain headway regularity
- minimise passenger waiting times and/or associated vehicle costs
- maximise synchronisation events between multiple services

Selection of dispatching times for a PT system appear as early as Bisbee et al. (1968) followed by Newell (1971) and Salzborn (1972) who focused on the single route timetabling problem with an objective to minimise passenger waiting time. Two types of timetabling problems exist in literature: *periodic* and *non-periodic*. In periodic timetabling, a given set of events or trips is scheduled in equally spaced intervals. Note that periodic timetables commonly appear in passenger railway services, predominantly in European countries; a common type of periodicity is introduced with clock-faced timetables, where the interval between trips is one hour. This class of problems is referred to as *Periodic Event Scheduling Problems* (PESP), which is a classic model introduced by Serafini and Ukovich (1989). The PESP aims to find a periodic timetable with minimum sum of passenger travel time without any track occupation conflicts. Although such timetables assist memorability of schedules and the regularity can eventually attract more passenger demand (Johnson et al., 2006), it does not take into account the imbalance between supply (trip dispatch) and demand (time dependent passenger arrivals). Also, with advanced technologies on real-time passenger information, passengers are less likely to remember the actual timetable. On the other hand, aperiodic timetabling disregards any periodicity in a timetable and has no repeating pattern of timetabled trips. Such timetables are more flexible in accommodating time-of-day dependent passenger demand and enables a certain degree of freedom in scheduling services. More recently, promising studies have emerged in combining the regularity of periodic timetables and the flexibility of aperiodic timetables (Robenek et al., 2017).

In planning cases where the headways are constant and demand flow is uniform through different blocks of time in a day, timetabling is a relatively easier problem to solve (Ceder and Wilson, 1986). The first vehicle departure is set (based on some clock time relative to each time period) and the consecutive vehicle departures are set as multipliers of the desired service headway. However, the need to coordinate schedules at a transfer location (a terminal or a time point where multiple services intersect) adds to the complexity of

[1] The time between multiple bus trips, usually to allow for some recovery time or driver meal breaks

timetabling. Desaulniers and Hickman (2007) observe that perhaps one of the most challenging components of timetabling is the synchronisation (or coordination) of timetabled trips such that the *transfers* within a given network are smooth and well-timed.

Several excellent general reviews exist on a wide range of objectives for timetabling (Ceder, 2016; Guihaire and Hao, 2008*a*; Ibarra-Rojas et al., 2015). However, a truly comprehensive review on all of those objectives is beyond the scope of this thesis. We are predominantly interested in the objective of minimising passenger waiting time while taking care of the judicious use of available operating resources. We dedicate Section 2.3 to introduce the concept of *timetable coordination* to meet this objective.

2.2.2 Vehicle Scheduling

Vehicle scheduling (often called the *Vehicle Scheduling Problem* or VSP) plays a vital role in PT planning as it is the first planning stage with a primary objective of minimising operator costs, such as vehicle acquisition and associated costs in dispatching those vehicles. It is an *operational planning* component where a chain of trips are created according to a given set of timetables. Each such trip chain is called a *vehicle schedule* and the entire chaining process is referred to as *vehicle blocking*. A block is a sequence of productive (active or live trip time) and unproductive trips (empty trip time: deadheads[2], pull-outs[3] and pull-ins[4]) performed with a bus. The cost structure of a VSP comprises the sum of costs incurred by all the trips in a vehicle schedule. In that context, the primary objective functions then aim to create vehicle schedules with:

- minimum unproductive trips and service time; and
- minimum vehicle fleet size

There is a large volume of literature on the VSP. Bunte and Kliewer (2009) provides a detailed overview on the modelling approaches for various vehicle scheduling problems and a synthesis of solution approaches. There are different variants of the VSP based on the number of depots/garages that the vehicles use. *Single Depot Vehicle Scheduling Problem* (SDVSP) is known to be the simplest version of the VSP, where a fleet of buses covering the trips pull-out of and pull-in to a single depot. The SDVSP is generally represented as assignment models and network flow models in the literature (Freling et al., 2001) and is popular with small-size transit agencies that rely on a single depot for operations. VSPs can also be categorised into the vehicle types in use (single or multiple). Bertossi et al. (1987) demonstrates that the simplest case of VSP, with a single vehicle type and a single depot is solvable in polynomial time and can be modelled as a minimum cost network flow problem. Surveys on the SDVSP and its extensions are covered by Desrosiers et al. (1995).

Although SDVSP can be solved easily, adding constraints that represent real-life scenario can make it intractable. Baita et al. (2000) provides an excellent outlook into the

[2]The time that a bus travels without any passengers; typically when buses need to travel inter-route

[3]The time the bus spends travelling from a depot to the start of an active trip

[4]The time the bus spends travelling from the end of an active trip back to the depot

approaches to solve the VSP under practical considerations. The main practical features include considerations to allow deadheading, vehicle fuelling at the end of a trip, and vehicle hold tactics to enable bus idle times before the start of a subsequent trip. They also consider the objective criteria of minimising fleet size, minimising unproductive trip times, and limiting the number of routes a bus is assigned to. To solve the problem, they compare a classic mathematical approach with compromised modelling capabilities against advanced approaches such as heuristics based on Logic Programming (LP) and Genetic Algorithms (GA). A wide range of good solutions are presented at competitive computational time

On the other hand, in cases where the vehicles use multiple depots, the *Multiple Depot Vehicle Scheduling Problem* (MDVSP) is common with medium to large size transit agencies. It aims to determine a feasible set of vehicle schedules emerging from each depot such that (i) each active trip is covered by exactly one vehicle schedule and; (ii) the number of buses dispatched from each depot is minimised. For more than 2 depots in scope, it can be formulated as a multi-commodity network flow problem. For large scale networks, this problem is known to be intractable (for proof, see Bertossi et al. (1987)). In spite of its complexity, solving real-life and large MDVSP instances has been attempted in the literature. Löbel (1998) presents a Column Generation (CG) based on Lagrangian relaxations for a large network instance and presented high quality feasible solutions at reasonable computational time. Kliewer et al. (2006) propose a novel time-space formulation that can limit the exponential growth of network size; they solved the MDVSP with upto 7000 trips using commercial solvers. Hadjar et al. (2006) (extending from Carpaneto et al. (1989)) solve the MDVSP with up to 750 trips and 6 depots to optimality using an exact Branch and Bound (BB) approach.

For those instances where finding sub-optimal (but good) solutions faster is a priority, heuristics have been proposed. Pepin et al. (2009) compares different heuristic approaches in solving the VSP. Instances are generated randomly with up to 1500 trips and 4 to 8 depots. This study argues that given ample computation time, the best performing approach to solving a MDVSP is truncated Column Generation. However, with limitations on computation time, the performance of *Large Neighbourhood Search* (LNS) meta-heuristics was found as the best alternative for finding good quality solutions. Ceder (2011) attempts an extension to the MDVSP, with even-load, even-headway timetables and multiple vehicle types. Formulated as a cost-flow network problem, a Deficit Function based heuristic is proposed. This study, in conclusion asserts on the need to understand the inter-dependencies between setting timetables and vehicle schedules. More recently, Guedes and Borenstein (2015) introduce a heuristic framework that combines time–space network formulation, truncated Column Generation and state space reduction, to solve large instances of the multiple-depot, multi-vehicle type VSP. The best solutions that simultaneously provided solution quality and efficiency, were obtained using the heuristics with state space reduction that decreased the problem complexity by using a manageable set of variables. Following this study, Guedes et al. (2019) propose a CG based algorithm to solve the multiple-depot, multi-vehicle type VSP considering fluctuation in bus passenger demand. Their approach promises to find a good compromise between operator costs and service quality to passengers, considering varying demand flow in a day.

For an overview on MDVSP and its extensions, we refer Desaulniers and Hickman (2007). The other variants of the VSP considering vehicle disruptions and punctuality include *Robust* VSP and *Dynamic* VSP. Since in this thesis, we do not consider stochasticity in bus operations, we refer the reader to Ibarra-Rojas et al. (2015) for a detailed review on these variants.

2.3 Timetable Coordination

In the age of rising private mode dependencies, it has been widely recognised that a transit network that offers city-wide integrated public transport services can attract ridership and improve the overall efficiency in its operations (Vuchic, 2005). To achieve multi-modal or inter-modal integration of different services (say buses and trains), it is necessary to enable convenient and seamless *transfers*. This can be achieved spatially and temporally; the spatial component optimally lays out a network with multiple services that can facilitate transfers, and the temporal component determines network schedules that are well-timed and coordinated. This concept of temporal coordination, widely known as *schedule synchronisation*, is the central theme of this research. We use a more focused term *timetable coordination* to refer to the same.

In the following sections, we give a brief context on the importance of transfers in a transit network and the role of timetable coordination in improving transfer efficiencies. We also explain why a network-wide timetable coordination is considered a complex combinatorial optimisation problem. We also outline the inherent complexity with finding optimally coordinated scheduling solutions and synthesise the various modelling and solving approaches studied so far in literature.

2.3.1 Importance of Transfers

Public transport journeys are often multi-modal, that is, comprising a combination of several modes or services. This involves passengers generally requiring to alight from one vehicle and board another; an action that is generally termed as performing *transfers* in a transit context. Evidence suggests that passengers dislike making transfers (Horowitz and A, 1994) as they interrupt travel and require them to physically orient themselves such that they can get to their destination. Therefore, it is sometimes perceived that transfers should be avoided whenever possible. Ideally, a simple, grid-based transit network with less transfers can be attractive for passengers. Transfers may be reduced by optimizing the transit network configuration, that is, optimally laying out transit routes such that the services are as direct as possible. However, there is limited evidence to support this concept. In fact, it is noted that only approximately 5% of all urban destinations can be accessed with direct, transfer-free services Currie and Loader (2010). Walker (2012) observes that although we could design direct networks with minimum number of transfers in a journey, enabling connections at a strategic point (say train stations) can ensure faster commute due to the higher frequencies it can offer. As also quoted by Vuchic (2005) on the necessity of transfers:

> *"The fact is...that transit networks with many transfer opportunities offer passengers much greater selection of travel paths than networks with a large number of integrated lines that involve little or no transferring."*
>
> Vuchic (2005, p. 497q)

For cities with circuitous transit networks like Melbourne, it is infeasible for all travel to offer direct services similar to car trips. Connecting all origins and destinations with direct routes would lead to a large number of bus lines running in parallel in all possible directions, in very low frequencies (Mees, 2000). An analysis by Currie and Loader (2010) on the relationship between network effects and transfers in Melbourne showed that transfer rates can be significantly improved for grid-networks with 10-min headway or better. But, it is not advisable to re-configure an existing network (which may or may not have an incentive for transfers) to a grid. Therefore, this study also cautions that the effect of network design on transfer rates, although an intriguing field of study, must in practice be dealt with more understanding of the real-world network characteristics. Thus, a well-planned transfer system must allow each transit line to be designed optimally for its physical characteristics and passenger demand flow, thereby improving the overall network efficiency.

While we establish that facilitating transfers is necessary, in cases where the frequencies of two given services do not harmonise, passengers experience delay in connecting from one service to the other. Such delays, called *transfer waiting time* are perhaps one of the most undesirable features of a transfer and contributes further to the dis-utility of public transport as compared to non-transfer alternatives (like cars) (Ceder, 2016; Guo, 2003). In the worst case scenario, missed transfers constitute the majority of reliability issues in PT services (Hadas, 2010). Analyses by Evans (2004) show that passengers tend to perceive the time they spend not travelling (called *out-vehicle travel time* or OVTT) including time spent on transfers as more inconvenient than that spent in-vehicle (called *in-vehicle travel time* or IVTT). When it comes to users' choice of mode, OVTT was considered 2-4 times as important as IVTT. Moreover, the transfer wait time component is often perceived to be more important than the initial wait for a service at the origin. This supports the fact that unlike initial wait time, transfer time (including walking and waiting) at an interchange is reliant on the efficiency of coordinated services and interchange facilities, and hence cannot be controlled by passengers (Iseki, 2009). Therefore, well-coordinated timetables play a vital role in improving the efficiency of passenger transfers.

2.3.2 Approaches

Academic literature on timetable coordination has shown problem specific modelling and solving approaches for a variety of objectives. The overarching aim of timetable coordination is to perform timetable adjustments such that it minimises the transfer waiting times for all related services in a multi-modal transit system. In this context, two main approaches for timetable coordination emerge: *timed transfer system* and *transfer optimisation* (Bookbinder and Desilets, 1992). These are described below:

Timed Transfer System (TTS)

In a timed transfer system (TTS), vehicles (say buses) from different lines are scheduled to arrive simultaneously at a transfer location and depart after a certain transfer window. This transfer window allows some buffer time for feasible connection in cases where the vehicles are running late. The main objective of this approach is to maximise the number of *synchronisation events* at a location such that the corresponding transfer waiting time is minimised (Vuchic, 2005). However, this objective eliminates a number of transfers outside the stipulated transfer window (Ceder et al., 2001) and is most effectively used on systems that operate with schedules that are less prone to uncertainty.

Transfer Optimisation (TO)

Considering the fact that transit systems that offer all-day and network-wide services rely heavily on passenger transfers, minimising the total transfer waiting time is an important timetabling objective. Transfer optimisation (TO) aims to achieve this objective and is widely discussed in literature as a popular timetable coordination approach (Guihaire and Hao, 2008*a*).

The total transfer waiting time is considered as the sum of all feasible transfer waiting time spent by passengers that transfer in a given planning period (De Cea et al., 1994). While TTS eliminates a large number of feasible connections and emphasises on achieving maximal synchronisation at a location, transfer optimisation considers all feasible transfers in all directions of connecting services, throughout the network (Daduna and Voß, 1995). A large number of research studies focus on timetable coordination as an optimisation problem with one or more objective functions and constraints. Apart from the main objective of minimising total passenger waiting time, studies also consider the following objectives (in conjunction with the main objective or otherwise):

- minimise the weighted sum of passenger costs and/or operational costs
- minimise the maximum waiting time at a transfer location of priority
- minimise the discrepancy in passenger demand flow
- minimise deviations from desired timetable standards
- minimise transfer dis-utility or penalty (qualitative objective)
- maximise the overall quality of transfers

Returning briefly to our main research argument, we explore the objective of minimising passenger transfer waiting time and required fleet size in a transit network and hence, the concept of transfer optimisation is the central theme of this thesis. In the following subsections, we explain why timetable coordination is considered a complex combinatorial optimisation problem. A review on the most relevant works on transfer optimisation and a synthesis on their limitations are presented later.

2.3.3 A Complex Optimisation Problem

Very rarely does a city have a comprehensive planning system that manages the operations of all public transport modes. In reality, several planning agencies interact with each other within the same regional area. Although having multiple operators is desirable to offer more frequent services, it is necessary to establish harmony in their operations as well. For cities with separate scheduling system for rail, bus and trams, timetable coordination is almost never achieved holistically as the scheduling constraints, objectives and techniques vary widely between operators. It is also important to note that schedule adjustments on a localised network is more feasible and easy to achieve than on a wider network. As Currie and Bromley (2005) noted, while it might be desirable to match a bus arrival to a train departure at a local level, it might be impractical to do so on a network wide basis. Any shift to the arrival times tend to cascade, forming uneven headways and affecting other coordinated arrival times along the route (positively or negatively). Also, a network-wide coordination problem can impose plenty of user and operator requirements, most often conflicting. A number of studies thus agree that transit timetable coordination in any form (timed transfers or transfer optimisation) qualify as a complex optimisation problem (Castelli et al., 2004; Ceder, 2001; Guihaire and Hao, 2008*a*; Shafahi and Khani, 2010; Shrivastava and Dhingra, 2002).

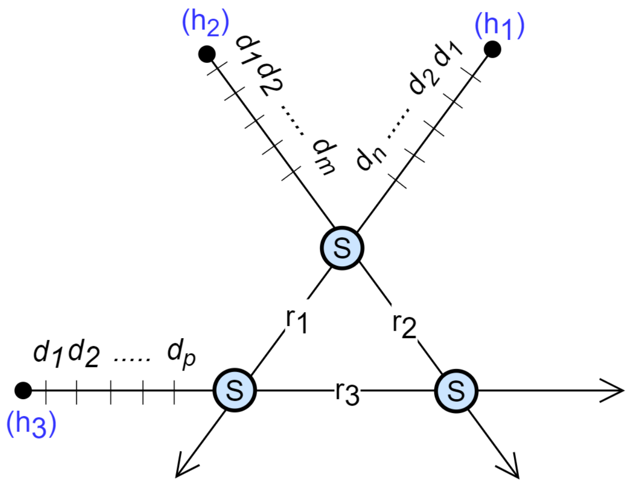


Figure 2.2: Illustration of a simple transit network with transfer locations

Consider a simple uni-directional network as shown in Figure 2.2 comprising three routes r_1, r_2, r_3 with different headway h_1, h_2, h_3 respectively, and intersecting at three transfer locations denoted as S. Let us set the departure time d from the first stop in each line as the decision variable by which we aim to minimise the transfer waiting time at the given locations. The departure time variable for each line can take any integer values within the given headways. Thus, there are $n.m.p$ possibilities for setting the timetables, considering all three lines. If we were to adopt equal headway h for each intersecting route r in the network, there will be h^r possibilities for first departure times. The complexity increases with the number of possible start-of-the-day times on each route. This further increases the complexity of the model exponentially when a network-wide scenario is considered, especially with variability in headway. Thus, the decisions on departure times will need to be made on a very large search space of size $\#buses \times \#trips \times \#horizon$.

In addition, when transfer passenger demand flow must also be considered, the objective includes a term that multiplies the number of transferring passengers by the transfer waiting time. Since both passenger demand and transfer waiting time depend upon the decisions on departure time for each vehicle trip, non-linearity is introduced in the objective. This makes timetable coordination a large and complex combinatorial optimisation problem, even for small transit networks (Schuele et al., 2009).

2.3.4 Models and Solution Techniques

As stated above, timetable coordination has been addressed using a variety of modelling and solving techniques. Similar to any other optimisation problem, it essentially has the following aspects:

1. selecting controllable parameter(s) as decision variable(s)
2. formulating the most relevant constraints and objective function(s)
3. devising an efficient solving technique to yield good quality solutions

For the timetabling component, vehicle departure time from the first stop is observed as a popular decision variable. Other decisions include determining the optimal headway, run-time and/or dwell time, terminal offset time (for TTS) and vehicle holding time (for real-time operations). Since expressed in minutes, these variables take only integer values. Additionally, the vehicle scheduling component decides the number of vehicles required and returns a specific assignment of each timetabled trip to a specific vehicle. This decision also takes integer values and are solvable when considered in isolation. However, the combination of decision variables from both timetabling and vehicle scheduling makes it hard to obtain optimal solutions.

In terms of mathematically formulating the timetable coordination problem, a number of proposals are made in literature, each characterised by its objectives, constraints and model parameters. Classically, the problem is represented as a Quadratic Semi-Assignment Problem (QSAP), Mixed Integer Program (MIP) or a Mixed-Integer Nonlinear Program (MINLP) (Ibarra-Rojas et al., 2015). A variety of optimisation strategies have been implemented to solve the model formulations. For cases where a problem can be formulated using known mathematical models, *exact* methods can be devised. However, in practice, exact methods are often limited to a set of simplifying assumptions to produce feasible solutions. As the problem size increases, these methods often suffer from a sharp rise in computational time and quickly become inefficient. Hence, a large number of reviewed papers on transit network planning use *approximation* algorithms to seek near-optimum solutions in a relatively short computational time. Solution methods based on heuristics and meta-heuristics are the most popular and Guihaire and Hao (2008*a*) classifies these into the following four categories:

1. Specific heuristics based on problem characteristics
2. Neighbourhood Search (e.g., Simulated Annealing (SA), Tabu Search (TS), Large Neighbourhood Search (LNS) etc.)
3. Evolutionary optimisation (e.g., Genetic Algorithm or GA)
4. Hybrid search (where multiple solution methods are combined)

We present past studies on formulating the timetable coordination problem and classify them according to the following major categories of solution techniques used: mathematical optimisation, heuristics, neighbourhood search, evolutionary algorithms and other methods. A brief outline on the use of commercial planning products is also given.

Mathematical Optimisation: Wong and Leung (2004) present a timetabling method for rail systems to maximise transfer synchronisation with minimum waiting time. A MIP model is proposed with running time, dwell time and departure time of each train as decision variables. This model relies on the assumptions of known and fixed transfer waiting times at a station, unlimited train capacity and exact adherence to planned schedule. Solving the problem using the CPLEX optimiser[5], their results demonstrate a reduction in passenger wait time by between 43% and 73% for minor timetable shifts and reasonable immediate cost. Presenting a quadratic semi-assignment model for timetable shifts, Schröder and Solchenbach (2006) address the objective of minimising transfer wait time considering the perceived quality of different transfer types. Solved again with CPLEX, global optimum solutions are obtained for a few instances in acceptable computational times. Bruno et al. (2009) aims to optimise transit timetable focusing in the trade-off between vehicle operating costs and transfer waiting time. Two mathematical models based on time-space network representation are presented and solved optimally using a commercial optimiser. Shafahi and Khani (2010) propose two MIP models for timetable coordination assuming uniform headway and deterministic running times. The first model determines the departure times of vehicles such that the total passenger waiting time is minimised. The decisions thus obtained are input into the second model, incorporating extra stopping time of vehicles at transfer stations as a new set of variables. Small and medium-sized network instances are solved quickly using CPLEX and GA is used for larger instances. Dou et al. (2015) propose a purely temporal model to minimise transfer failures from buses to last trains by offsetting bus schedules. Given no inputs on travel demand and transfer costs, their vital modelling criteria is that at least one of the serving bus routes must provide feasible last train connections. The formulated MINLP model is transformed to an equivalent MILP model to be solvable using CPLEX. Even though this model is used to coordinate buses with only last train services, when used in large network cases, long computational time is still a major challenge. Thus, the authors favourably point towards the use of heuristics.

There is a general agreement in the literature that while mathematical techniques can deal with traditional integer optimization problems with discrete or continuous variables, these can only be used to represent simple modelling scenarios. It is also acknowledged

[5]https://www.ibm.com/au-en/analytics/cplex-optimizer

that the size of an instance is a serious limitation to solve practical, real-world problems optimally.

Heuristics: Klemt and Stemme (1988) popularise the use of heuristics in transfer optimisation. They propose a Quadratic Semi-Assignment model to minimise transfer waiting time given a fixed number of trips as input. To solve this formulation, a constructive process is developed where trips are scheduled and synchronised individually. Domschke (1989) later improve these solutions further by designing algorithms based on Branch and Bound (BB), Local Search (LS) and Simulated Annealing (SA). These seminal works also give us a preliminary insight into the practical challenges associated with timetable coordination.

Castelli et al. (2004) present a Lagrangian based heuristic algorithm in which transit line schedules are optimised one at a time, partially correcting multiple decisions in the previous steps. Unlike frequency based line planning problems, the notable feature with this study is that the number of lines are determined by a fixed cost per line run. The quality of service is then evaluated using the total passenger transfer wait time. This study provides a rationale behind using the heuristics, acknowledging that their model is intractable for network-wide optimisation. Fleurent et al. (2004) present an innovative concept of *trip meets* that describes a potential connection between two given trips at a transfer location. Each transfer is characterised based a weight factor and a minimum, maximum, and ideal waiting time limits. The solution methods include solving network flow problems, Lagrangian relaxation and some heuristics, but there is limited information provided on these. This model is implemented in the commercially available software GIRO-HASTUS[6]. Wong et al. (2008) present a MIP model similar to that proposed by Klemt and Stemme (1988) for non-periodic train timetables that minimises the interchange waiting times of all passengers. The model incorporates bounds for headway and deterministic running times and is iteratively solved using piece-wise optimisation of the complete problem. Although the model is devised for rail systems, it is quite abstract to be of use for general public transit.

Neighbourhood Search: The use of meta-heuristic methods such as Simulated Annealing, Tabu Search, nearest neighbour algorithms etc. are quite popular in public transit timetabling. Voß (1992) present a coordination model to determine the vehicle departure times such that the passenger waiting time at selected transfer locations is minimised. In addition, a second problem is considered where different lines are allowed to use the same tracks partly such that the security distance constraints are satisfied. Both models are solved using Tabu Search. Similarly, Daduna and Voß (1995) propose a quadratic semi-assignment model to synchronize vehicle arrival times at transfer locations such that the corresponding waiting time is minimised. The authors also considered refining this objective further by including weights on the different kinds of transfers or on the maximum waiting time at a transfer location. Due to problem complexity, the initial solutions were

[6]https://www.giro.ca/en-ca/our-solutions/hastus-software/

computed using a variant of a regret heuristic; the remaining solutions are then found using Simulated Annealing and Tabu Search.

Jansen et al. (2002) propose a MINLP model to minimize the weighted sum of transfer waiting times given a route network with fixed headways. The weights use a general function for transfer passenger flow. Assuming constant and deterministic values for bus stopping and in-vehicle time, the problem is reduced to finding departure times for the first run of each line. The model is then solved using a Tabu Search heuristic algorithm, and tested on a large-scale case study from Copenhagen.

Zhao and Zeng (2008) acknowledge that the improvement in objective value can be approximately proportional to the CPU time, for large network instances. This is intensified when stochastic travel times are considered, impacting the probability of achieving well-timed transfers. Thus, Liebchen and Stiller (2012) develop this study further to add buffer times that generate delay resistant timetables while also minimising the price accrued for any delay. An optimisation problem is formulated for railways that minimise the potential weighted delay of trips with constraints on headway bounds and fleet size. A certain sampling approach is proposed to solve aperiodic delay resistant timetabling and two heuristics are proposed for the periodic problem.

Schuele et al. (2009) propose a novel mathematical model to classify transfers into convenient, risky and patience-requiring and aim to improve the overall quality of these transfer classifications. Minor adjustments to departure time are decided using meta-heuristics such that it minimises the overall transfer waiting time and maximises transfer quality. However, they also caution that this methodology is not holistic, but can offer guidance for planners by ranking transfers based on their relative relevance at a location.

A notable advantage with using heuristics and meta-heuristics is the fact that these are designed to accommodate many forms of mathematical models. These methods can hence define a general search framework for scheduling problems and is versatile enough to adapt to multiple, conflicting forms of constraints and objectives.

Evolutionary Optimisation: Evolutionary optimisation have features related to mathematical optimization, however it has emerged as an efficient technique for multi-objective, nonlinear problems. Much focus has gone into the application of Genetic Algorithms (GA) for the optimisation of timetables (Chakroborty et al., 1997, 1995; Deb and Chakroborty, 1998). Shrivastava and Dhingra (2002) present a model for minimising the bus-train transfer waiting time and vehicle operating costs. Penalties are applied if one or more constraints that reflect the passengers' satisfaction from the system are violated. Upon demonstrating that the problem is intractable using commercial solvers, they utilise the solving capabilities of GA that produce promising results. However, GA was also found to be very sensitive to the penalties added, resulting in wide variations in the results. Chakroborty (2003) further explains the efficiency of GA in comparison to traditional solving techniques but also cautions that some simplifying assumptions (such as unlimited vehicle capacity and fixed fleet size) are inevitable to make the problem tractable in large instances. Following this, Cevallos and Zhao (2006) proposes a GA based network-wide approach to minimise the total transfer waiting time by shifting existing timetables. Given existing timetables and known passenger volume at all transfer locations, their model constraints the headway to be strictly fixed, depriving the model of any flexibility but also making the problem more tractable in larger networks. This model hence filters infeasible transfers between multiple lines. Wu et al. (2015) presents a different take on minimising the passenger waiting time, while limiting the waiting time equitable over all transfer stations in a network. Decisions are made on adjusting the vehicle departure time, running time, dwell time and headway in all directions such that the worst transfer times are improved. GA is chosen as a suitable solving technique to represent the complete schedule using a series of binary variables. More recently, Li et al. (2019) propose a methodology to estimate potential passenger demand for last train transfers from a variety of connecting modes (bus, taxi etc.). Consequently, this study proposes models for demand-dependent last train coordination problem. A GA based solution algorithm is devised to maximise the last train connectivity. Results indicate that the proposed optimisation scheme is able to achieve improvements in passenger transfer demand, in comparison to the existing passenger flow.

Other methods: Bookbinder and Desilets (1992) propose a hybrid optimisation approach to minimise mean transfer dis-utility. The authors assume stochastic travel times and a unique headway value for each line; the first departure is the only decision variable while the rest of departures time are computed from it. The solution approach combines the flexibility of simulation and mathematical optimisation (similar to that presented by Klemt and Stemme (1988)). Dou et al. (2017) propose a MINLP model to minimise the weighted sum of transfer costs and vehicle operating costs for coordination between feeder buses and rail transit. A hybrid solution method combining heuristics and GA is proposed to determine terminal departure time for each feeder bus trip, and the related headway.

de Palma and Lindsey (2001) present an analytical model for the timetabling problem with known passenger demand. Considering a single transit line, first, passengers are assigned to certain line runs and then, an optimisation process is led to determine the departure times such that it minimises passengers' total schedule delay costs. Unfortunately, the oversimplification of this model makes it inadequate to be represented on a real-world setting.

As observed so far, timetabling problems comprise different objectives that can be conflicting. Some research thus attempt multi-objective optimisation such that the trade-offs between different objective functions are represented. For example, Kwan and Chang (2008) formulate a bi-objective model that aims to minimise both transfer costs and costs incurred by deviations from an initial timetable. They implement a state-of-the-art non-dominated sorting Genetic Algorithm II (called NSGA II) to obtain the relationships between the two objectives and use a combined hybrid solution methodology to produce good quality results. Hassold and Ceder (2012) also study a similar problem of minimising expected passenger waiting time and penalty for empty seats. They acknowledge the disadvantages of using even headway and even passenger loads in timetabling and suggest that the overall operational efficiency of transit can be improved with the prudent use of multiple vehicle types on the same line. First, with known departure time, multiple vehicle types are determined to cover all trips. A multi-objective label correcting algorithm then solves the problem. Results show more than 43% savings in passenger waiting time at an acceptable passenger load on all vehicles.

Attempts have been also made in literature to evaluate the efficiency of multi-objective solutions using fuzzy approaches. Tilahun and Ong (2012) formulate the timetabling problem as a fuzzy multi-objective model to minimise the waiting time for different types of transfers in single frequency routes. They implement GA to solve instances upto 10 intersecting lines.

Given the dynamic nature of the problem, several studies have also analysed the significance of transfer coordination on improving public transport timetables under uncertainty. Lee and Schonfeld (1991) formulate a total cost function that can determine optimal *"slack"* time between one bus route and one rail line. Their model assume stochastic vehicle conditions and conclude that there is no incentive for timetable coordination when there is significant variability between vehicle arrival times. Similarly, Knoppers and Muller (1995) evaluate the relevance of transfers in an intercity bus feeder system by considering the punctuality of bus arrivals. They observe that coordination is worth attempting when the standard deviation of bus arrival punctuality on the feeder line is less that 40% of the headway on the connecting line.

Table 2.1 presents a summary of the literature relating to transfer optimisation based timetable coordination reviewed so far, also outlining the practical considerations from each study.

Year	Author(s)	Objective	Decision Variables	Model	Solution Methods	Case	Network Size	Practical Considerations					
								Headway	RT_c	RT_v	PD_c	PD_v	Cap
(1988)	Klemt and Stemme	Min. transfer waiting time	Departure time	QSAP	Heu.	Example	Small	Ht_e	✓	×	✓	×	×
(1989)	Domschke	Min. transfer waiting time	Departure time	QSAP	BB, Heu.	Example	Small	H_e, Ht_e	✓	✓	×	×	×
(1992)	Bookbinder and Desilets	Min. mean transfer dis-utility	Stop offset time	IP	Hybrid	Example	Medium	H_e, Ht_e	×	✓	×	×	×
(1992)	Voß	Min. transfer waiting time	Departure time	IP	TS	Example	Small	H_e, Ht_e	✓	×	✓	×	×
(1995)	Daduna and Voß	Min. transfer waiting time	Departure time	QSAP	TS, SA	Example	Small	H_e, Ht_e	✓	×	✓	×	×
(1995)	Chakroborty et al.	Min. total transfer waiting time	Headway and stop time	MINLP	GA	Example	Small	H_u	✓	×	✓	×	×
(2001)	de Palma and Lindsey	Min. total delay cost	Departure time	-	Analytical	-	-	H_e	✓	×	✓	×	-
(2002)	Jansen et al.	Min. weighted sum of wait time	Departure time	MINLP	TS	Real	Large	H_e	✓	×	-	-	-
(2002)	Shrivastava and Dhingra	Min. waiting time & vehicle costs	Fleet size & frequency	MINLP	GA	Real	Medium	H_u	✓	×	×	✓	✓
(2004)	Fleurent et al.	Min. waiting time & vehicle costs	Route time shift	HASTUS	Heu.	Example	Small	H_e	✓	×	-	-	-
(2004)	Wong and Leung	Min. total transfer waiting time	Departure, running & stop time	MIP	CPLEX	Real	Large	H_e, Ht_e	✓	×	✓	×	×
(2004)	Castelli et al.	Min. user & operator costs	Multiple coordination variables	MINLP	Heu.	Test	Small	H_e, Ht_e	✓	×	✓	✓	-
(2006)	Cevallos and Zhao	Min. total transfer waiting time	Route time shifts	IP	GA	Real	Large	H_e	✓	×	✓	×	-
(2006)	Schröder and Solchenbach	Min. total delay costs	Line time shifts	IP	CPLEX	Real	Small	H_e	✓	×	×	×	-
(2008)	Wong et al.	Min. waiting time costs	Arrival, departure, travel time	MIP	Hue.	Real	Medium	Ht_e, $H_{u+tol.}$	×	✓	✓	×	×
(2008)	Kwan and Chang	Min. transfer & deviation costs	Frequency, travel time	MINLP	GA & Hybrid	Example	Small	-	✓	×	✓	×	✓
(2009)	Bruno et al.	Min. user & operator costs	Departure time, no. of lines	Network	Xpress-MP	Real	Small	Ht_e	✓	×	✓	×	✓
(2009)	Schuele et al.	Max. transfer quality	Deaprture time	QSAP	Meta-H.	Real	Medium	H_e	-	-	✓	×	-
(2010)	Shafahi and Khani	Min. transfer waiting time	Departure time, holding time	MIP	CPLEX, GA	Real	Large	H_e	✓	×	✓	×	×
(2012)	Hassold and Ceder	Min. wait time & empty seat penalty	Potential departure time	Network	Hue.	Test	Small	Ht_e	✓	×	✓	×	✓
(2012)	Liebchen and Stiller	Min. expected delay	Buffer time costs	MIP	Heu.	Example	Small	$H_{u+tol.}$	×	✓	-	-	-
(2012)	Tilahun and Ong	Min. waiting time for diff. transfers	Arrival ranking	Fuzzy	GA	Example	Small	H_e	✓	×	-	-	-
(2015)	Wu et al.	Min. and limit waiting time	Departure, travel time & headway	IP	GA + LS	Numerical	Small	Ht_e	✓	×	✓	×	-
(2015)	Dou et al.	Min. transfer failures	Departure time, stop offset time	MILP	CPLEX	Real	Large	Ht_e	✓	×	-	-	×
(2017)	Dou et al.	Min. weighted sum of costs	Departure time	MINLP	Heu.	Numerical	Small	Ht_e	✓	×	✓	×	✓

H_e: Homogeneous, even
Ht_e: Heterogeneous, even
H_u: Uneven; *tol.*: tolerance
RT_c: Constant run-time
RT_v: Variable run-time
PD_c: Uniform passenger demand
PD_v: Variable passenger demand
Cap: Vehicle capacity limits

QSAP: Quadratic Semi-Assignment Problem
IP: Integer Programming
MIP: Mixed Integer Programming
MINLP: Mixed Integer Non-Linear Programming
Network: Time-space network models
Fuzzy: Fuzzy problem formulation
HASTUS: Scheduling Software

BB: Branch and Bound
Heu.: Heuristic methods
TS: Tabu Search
GA: Genetic Algorithm
Meta-H.: Meta-heuristics
LS: Local Search
Decomp.: Decomposition methods
SA: Simulated Annealing
CPLEX, Xpress-MP : Commercial solvers
Hybrid.: Combination of methods

✓Considered in the model
× Not considered in the model
- Not enough information

Source: Author's synthesis of literature review on timetable coordination based on the studies cited within the table

Table 2.1: Summary of literature review on transfer optimisation based timetable coordination

2.3.5 Limitations with Timetable Coordination

As stated briefly in Section 2.2.1, coordinating timetables is perhaps the most challenging but vital problem in planning. A multitude of problem specific limitations have emerged from the studies reviewed so far in this chapter. Collectively, we can categorise these into limitations in model formulations and solving techniques.

Modelling Limitations: It is evident from our review that timetable coordination is never a single solution problem. The diversity with formulating this problem is reliant on specific objectives, modelling parameters, decision variables and constraints. In practice, the issue of coordination is critical as the network size and planning period increases. To get cohesive scheduling solutions that benefit both users and operators alike, practical considerations must be made while formulating the problem. As quoted by Ceder (2007):

> *"In order to gain a better approach to coordination at connecting points, there is a need to relax some of the rigorously defined timetable parameters."*
>
> Ceder (2007, p. 142)

However, being a complex optimisation problem in itself, considering practical features would intensify the complexity and almost render the problem not solvable using existing solving techniques. Most methods in literature tackle this limitation using a set of simplifying assumptions (Poorjafari and Yue, 2013; Schuele et al., 2009) that make the problem solvable. This however, deprives it of any flexibility and delivers solutions that are inefficient and practically non-implementable. Such assumptions to reduce model complexity include:

- Homogeneous or heterogeneous, even headway for multiple service lines
- Constant route run-time for a given time-of-day
- Discarding unproductive service times which impact PT cost
- Discarding time dependent transfer passenger volumes
- Not incorporating vehicle capacity limits
- Not integrating multiple vehicle types
- Discarding stochasticity in scheduling

Another vital challenge observed with defining the problem holistically is the mixed and conflicting nature of objectives and constraints associated with it in the real world (Ceder et al., 2001). While devising the constraints for the problem is less difficult, achieving a trade-off between these constraints is hard. For example, as Currie and Bromley (2005) notes, optimising passenger wait times often conflict with optimising fleet size and vehicle allocation since the former requires that vehicles spend time waiting whilst the latter requires faster speeds and shorter waiting to save operator costs.

Solving limitations: We have seen in subsection 2.3.3 that timetable coordination is a complex optimisation problem. Multiple vehicles plying on multiple routes offer a number of transfer possibilities. The above mentioned challenges intensify when the problem is faced with real world constraints. Incorporating variability into journey planning is necessary (Botea et al., 2013) but can lead to problems that are more complex to solve using traditional computational techniques (Nonner, 2012).

The intractability of the problem is because of the need to search for the optimum solution in a search space that grows exponentially as the problem size increases. As proven by Ibarra-Rojas and Rios-Solis (2012), an accurate mathematical formulation of a network-wide timetable timetable problem results in NP-hardness (i.e. the hard problem in non-deterministic polynomial problem class). Unless accompanied by some model flexibility and simplified assumptions, there is no discoverable algorithm that can guarantee optimal solutions to this problem. In other words, there is a degree of compromise that must be made between solution optimality and efficiency in order to solve this problem.

2.4 Constraint Based Optimisation

In view of defining real world constraints, Constraint Programming (CP) is a platform where relations between decision variables are stated in the form of a set of constraints. It reaps feasible solutions out of a very large set of candidates, where the problem can be modelled in terms of arbitrary constraints. The ability of CP in using a combination of variable and value-selection heuristics to guide the exploration of search space and its adaptability to almost any form of constraint and objective makes it a strong tool in solving difficult scheduling problems like timetable coordination (Belov et al., 2016).

From our research synthesis so far, we observe that a CP based solving approach is not used extensively for timetable coordination in transit. We are only aware of a few studies that use CP to solve transit network planning problems such as: Barra et al. (2007) who propose a constraint satisfaction model to be used in combination with a commercial constraint programming solver. There is no objective in his model but 11 types of essential and complementary constraints spanned across passengers demand, budget limits and level of service, define the problem. This paper represents a first work towards CP solution of the transit network design problem but did not produce any substantial results. The authors resort to reducing the instances' complexity since the chosen CP system demanded a huge computational effort. However, this study defined certain paths that suggest more efficient searches. CP is more popular in railway optimisation, such as the study proposed by Kroon et al. (2009) where a constraint programming based solutions approach called *CADANS* is used for a PESP formulated as a MIP. This study emphasises on the weakness of MIP in representing large instances and presents a rationale for using CP to find a feasible solution. *CADANS* provides solutions that satisfy all PESP constraints that they considered, in a few minutes. It also identifies when a feasible solution does not exist and the conflicting constraints that can be improved for feasibility. One of the limitations with this implementation is that it cannot directly optimise a timetable. Instead, when a feasible timetable is obtained, *CADANS* uses a post-optimisation approach to improve the timetable by adjusting the arrival and departure times of the trains. A similar research by Liebchen et al. (2008) compares different solution approaches to solve the PESP. A constraint programming algorithm creates feasible initial solutions that are at par with those from CPLEX and much superior to Local Search procedures. Interestingly, it is observed that CP is able to produce initial feasible solutions in less than half a second, even on the largest instance. CP suffered only on those instances where a specific value selection criteria was given.

2.5 Sequential Planning

2.5.1 State of the Art

A large and growing body of literature has investigated mathematical models and advanced solution techniques for each public transport planning problem (Ibarra-Rojas et al., 2015). However, solving these problems independently and sequentially does not always lead to holistic and practical solutions preferable for a transportation system. As noted by Ceder (2016), the output from top-level planning sub-problems need not be a good input for those downstream. For example, a well-coordinated timetable with minimum waiting time may not provide cost-efficient vehicle schedules. Sometimes minor timetable shifts can result in saving a complete vehicle. In which case, the timetabling problem must be re-solved with additional constraints that guarantee feasible and cost-efficient vehicle schedules. Planners usually have to identify conflicts from previous sub-problems to proceed with finding feasible solutions for the consecutive stages (Polinder, 2015).

Each sub-problem needs a specific objective against which it is optimised, (for example minimising the deadhead time when solving the VSP). By solving individual objective functions at each step we may be able to find locally "optimal" solutions for the sub-problem in question, however, there is no guarantee that the optimal solution to an earlier sub-problem is really part of a "globally" optimal solution to the complete problem (Bussieck et al., 1997). Indeed by changing the objective for the earlier sub-problem, the resulting changes to the solutions to the full problem could be made unpredictably worse or better.

It is also important to bear in mind that the objective functions used in each planning stage are often approximations on what a particular study focuses on (Schöbel, 2017). For example, timetabled trips are determined with some approximation on the fleet size and its associated cost while the real costs cannot be determined until the lower stream problems of vehicle and driver scheduling are solved. Thus, the wide array of objectives for each sub-problem reflect this fact.

2.5.2 State of the Practice

Operations Research based Decision Support Systems (DSS) play a crucial role in assisting transit agencies in creating cost-effective and efficient scheduling procedure. Different transit agencies worldwide use different tools to do so, based on their own level of expertise and the kind of problems they tackle. Hence, it is unlikely that any two given agencies (or operators) would use the same scheduling procedure. Even within the same transit agency, planners would sometimes need to adopt different scheduling practices for different groups of data. It is also important to address the fact that schedule adjustments on a localised network is more feasible and easy to achieve than on a wider network. As Currie and Bromley (2005) notes, any localised timetable shifts at a time-point tend to carry forward to other time-points, making a network-wide scheduling approach extremely complex. Most often, a scheduler would then need to put to use his or her scheduling expertise and some intuition to make faster decisions. As Ceder et al. (2001) observes about the schedule synchronisation sub-problem:

> *"Synchronization is the most difficult task of transit schedulers and is currently addressed intuitively"* Ceder et al. (2001, p. 914)

Current commercial tools can help trade-off the optimisation of one sub-problem against the other to an extent, but it is still a challenge to achieve network-wide practical solutions in a multiple route, multi-operator planning scenario like Melbourne. Given the fact that the major cost drivers for a transit agency are driver wages and vehicle acquisition costs, a number of mainstream commercial software on transit scheduling and analysis such as: GIRO-HASTUS[7], PTV[8], SYSTRA[9], TRAPEZE[10], etc. focus primarily on vehicle and crew scheduling (Ceder, 2016). However, such software are known to require some manual interference by the schedulers to yield holistic solutions. As observed by Torrance et al. (2009) in an extensive survey based review on the properties of the most widely-used automated scheduling tools in the industry:

> *"Due to the large number of possible blocks and shifts, it is not possible for any of these programs to successfully solve the optimization problem without resorting to decomposition of the original problem. This decomposition results in reducing the original problem to a smaller sample by removing any shifts observed to be inefficient. By removing these shifts, the solution is not guaranteed to be optimal since all possible shifts have not been considered."*
> Torrance et al. (2009, p. 9)

This asserts the fact that there is no universal outlook to scheduling and motivates the need to embed multiple scheduling options in a DSS (Ceder, 2007). The DSS must be able to perform a major part of decision making, coinciding with the scheduler's need to perform manual or semi-manual adjustments to make a solution better. This aids a scheduler in analysing and comparing the trade-off between passenger service quality and budgetary implications without having to rely heavily on intuition.

2.5.3 Re-ordering Sub-problems

While we outlined the disadvantages of sequential optimisation in planning, when it comes to tackling large-scale problems, there is also scope and novelty in thinking about the different ways we can decompose the entire planning process to yield feasible and good-quality

[7] https://www.giro.ca/en-ca/our-solutions/hastus-software/
[8] www.ptv.de
[9] www.systra.com
[10] www.trapezegroup.com.au

solutions than the traditional sequence. Michaelis and Schöbel (2009) identifies that a major drawback with following the classic planning procedure is that the main drivers of costs (fleet size, crew wage etc) are not considered until a later stage. They propose to re-order the sequential planning approach, by first designing the vehicle routes, splitting them into lines and calculating aperiodic timetables for these lines. In each of these three steps, costs are included and the objective is passenger-oriented. Using heuristics, they demonstrate promising results with reduction in number of vehicles and improvement in the attractiveness of the timetables. Expanding on this, Schöbel (2017) integrate line planning, timetabling and vehicle scheduling through an algorithmic scheme called the *eigenmodel*, which is a bi-objective model iterating between the three planning sub-problems; in the starting phase, the classical sequential approach is adopted to construct lines, timetables and vehicle schedules that are re-optimised in the iterations later, depending on the order the problems are solved (say iterating between line planning and timetabling; then between line planning and vehicle scheduling, so on and so forth). This continues until all sub-problems are combined. Pätzold et al. (2017) present three interesting "look-ahead" strategies to include vehicle scheduling aspects earlier in the planning stages: (i) line planning with a new cost structure on number of vehicle needed, (ii) selecting a pool of lines with good vehicle schedules (iii) vehicle scheduling before and after timetabling. Recently, Lübbecke et al. (2019) explore various decomposition approaches that are superior to the canonical decomposition into the planning processes. They present an integer linear programming framework for line planning, timetabling and vehicle scheduling with passenger routing.

Given the complexity with finding which decomposition methods are most suited for problem specific planning, the above mentioned studies acknowledge that there is a need for extensive computational study to understand the correlation between the sub-problems in scope. Decomposition can be a poor strategy if the sub-problems in hand are highly inter-dependent, but it can also yield high quality solutions provided the optimal solution to one sub-problem is compatible with that of the others. To the best of our knowledge, no decomposition strategy has been adopted to solve timetabling and vehicle scheduling together with an objective of transfer optimisation.

2.6 Integrated Planning

We present here an overview on the integration of various planning sub-problems. In particular, we give a detailed review on integrated timetabling and vehicle scheduling that is relevant to this thesis.

2.6.1 Overview

As already seen in Section 2.5, sequential approaches, although effective in producing locally optimal solutions for each sub-problem, do not give us cohesive, holistic solutions for the planning problem as a whole. This has motivated more recent attention to integrate two or more of the planning stages and solving those simultaneously. The importance of integrating multiple planning stages is recognised as early as Claessens et al. (1998) who attempt the combination of line planning and timetabling and Nachtigall (1998) who estimate the required fleet size in the timetabling stage. Ibarra-Rojas et al. (2015) reviews that integration is commonly attempted in the following ways:

i. partial integration approach: the characteristics of one sub-problem is considered mainly while taking decisions of other sub-problems, and/or an iterative sequential approach where the degrees of freedom of the integrated sub-problems is explored in iterations;
ii. complete integration approach: model formulations and/or solution approaches that determine decisions for the complete problem

For SDVSP instances, iterative sequential approaches are more common since it is easy to schedule vehicles after timetable modifications. However, a complete integration of timetabling and vehicle scheduling is extremely challenging, due to the inter-dependency between the two sub-problems. With trip departure times as decision variables, the model lacks a fixed network for the vehicles to be assigned into; instead it must rely on a set of potential timetabling decisions. Evidently, complete integration is the most complex of all, since it considers all degrees of freedom for each sub-problem. Hence, partial integration is usually observed in literature as an approach to combine the features of planning sub-problems (Ibarra-Rojas et al., 2015).

2.6.2 Integrated Timetabling and Vehicle Scheduling (I-TTVS)

More attention has been paid to the integration of timetabling and vehicle scheduling over the past decade, since these sub-problems primarily reflect the trade-off between user and operator requirements (and the conflict in decision making thereof). Timetabling and vehicle scheduling are extremely complicated processes by themselves in practice, and integrating them for the objective of simultaneous user and operator cost minimisation is naturally even more complicated. In the context of partial integration, Ceder (2001) is the first study that attempts to integrate the features of timetabling and vehicle scheduling for the SDVSP case, for timed transfers. This study considers a 4-step sequential approach, but with a feedback loop to adjust the initial timetables again. Following this, Chakroborty et al. (2001) propose a GA based algorithm to simultaneously optimise the fleet size and waiting time for passengers, although with the limitation that each bus gets to serve only one line. The authors also caution that the chosen genetic representation for this procedure will become cumbersome for large transit systems.

When developing an integrated model for TT and VS, it is important to manage the inter-dependencies between the contrasting objectives of maximising timetable quality and minimising operating costs. In this regard, we briefly describe below the strategies adopted in modelling the integrated problem. Due to the complexity of the problem, most studies rely on the capabilities of meta-heuristic or matheuristic algorithms[11] to obtain good quality solutions in reasonable computational time.

[11] combination of meta-heuristics and mathematical programming techniques; Boschetti et al. (2009)

Bi-level programming: In bi-level programming, optimisation decisions made on the upper-level (say, by the operators) is used by the lower-level decision makers (say, the users) in optimising their own set of objectives. For example, Liu and Shen (2007) integrate the timetabling problem proposed in Liu et al. (2007) and a MDVSP. The upper model determines optimised vehicle schedules minimising the bus fleet size and deadhead costs and the lower model selects a set of satisfied solutions from the upper model to derive minimised transfer waiting times. Implemented on a small network example, a Nesting Tabu-Search solves the problem systematically. A more complex bi-level IP model is proposed in Liu and Ceder (2017) where lower-level decisions also consider users' response with optimal route choice (passenger assignment). While the results are promising, this study did not consider time dependent passenger demand. More recently, Schiewe (2020) formulates the integrated timetabling and vehicle scheduling as a studies a bi-criteria problem where the weighted sum of passenger travel time and the operational costs. Discarding passenger route choices, this study combines periodic timetables with a more general case of aperiodic vehicle schedules in order to handle the variations in planing period.

Timetable Shifting: Characterised as the *Vehicle Scheduling with Time-Windows* (VSP-TW) problem, this strategy allows minor modifications or shifts to a given timetable, such that it derives cheaper vehicle schedules. Clearly, the objective of minimising operator costs in prioritised here, but with a condition that the timetable quality is not compromised. Van den Heuvel et al. (2008) expand on a seminal work on MDVSP by Kliewer et al. (2006) and modify an initial timetable to find better vehicle schedules using a Local-Search based algorithm. They report operating cost reductions of up to 8% when compared to the initial timetable. Similarly, Guihaire and Hao (2008*b*) implement an iterative sequential model for I-TTVS to optimise the fleet size, headway evenness, and the quantity and quality of transfers. An Iterated Local Search (ILS) method is used to solve a single depot problem, where each iteration implements trip-shifting and then, solves the VSP given the current timetable. Ceder (2016) studies the expansion of fixed trip schedules to variable trip schedules by shifting departures times within certain acceptable tolerances such that it enables the minimisation of fleet size. As a rule of thumb, shifting tolerances are decided based on the headway. This study proposes an SDT (shifting trip-departure time) algorithm based on a Deficit-Function model to enable fleet reduction involving trip departure shifting and DH trip insertion. Gkiotsalitis and Maslekar (2018) addresses the conflicting problem of bus service regularity and passenger waiting time minimisation. The bus coordination problem is formulated first to minimise the waiting time of passengers at transfer stations. Due to the computational complexity of this problem, a sequential hill-climbing heuristic is proposed such that near-optimal solutions are generated by re-scheduling trip departure times. Results from a small-scale network in Stockholm indicate improvement in transfer waiting time at minimum deterrence to the service regularity. Recently, Fonseca et al. (2018) and Carosi et al. (2019*a*) introduce matheuristic approaches for transfer optimisation with integrated TT and VS. The former use an integrated bi-objective Mixed Integer Programming (MIP) formulation that address selected sub-problems of I-TTVS in iterations, to minimise the weighted sum of transfer costs and operator costs. Their results indicate competitive solution quality in comparison to general purpose solvers, at a computation time of 1-5 hours. However, this model requires an initial set of timetables as input to work with. The latter presents the integrated TT and VS as a multi-commodity flow type problem, with weighted bi-objectives of service regularity and operator cost minimisation. This study mainly considers restricted operating environments and was tested only on single-line settings with reduced interlining.

Objective Weighting: Another popular approach in multi-objective optimisation in I-TTVS is weighting, where certain weights represent the priority of each objective against the other, and the global objective comprises the sum of these weighted objectives. In Guihaire and Hao (2010), the authors consider multiple weighted objectives that improve transfer quality and minimise vehicle operator costs. They consider a constraint that limits deviation from an initial input timetable, which also allow them define feasible shifting procedures. Tabu-Search is used to solve the proposed model. As a further development in HASTUS-*NetPlan*, Fleurent and Lessard (2009) implement the I-TTVS problem as the weighted sum of two objectives (viz., minimising timetable costs and vehicle costs). There is limited information on the heuristics they had adopted to solve the problem.

Petersen et al. (2013) focus on the simultaneous vehicle scheduling and passenger service problem (SVSPSP) on a multi-depot setting. They use LNS to initially obtain vehicle schedules without the timetable components. The solutions obtained in 12 hours are then input to solve the SVSPSP. In an aim to improve passenger service, they consider weighting alternative trips with departure times that could reduce waiting times. Agreeing to the findings by Pepin et al. (2009), the authors state that unlike other meta-heuristic methods, LNS has the capability to search (or sample) a neighbourhood of large size, a feature that is desirable for I-TTVS on a realistic setting. Similarly, Schmid and Ehmke (2015) model the problem as a *Vehicle Routing Problem with Time Windows* (VRP-TW). They use a Hybrid LNS approach to decompose and solve the problem in two phases: scheduling service trips and balancing their departure times such that the deadheads and headway deviations are minimised.

Pareto Front: Where selecting appropriate weights is difficult, it is feasible to try and produce a set of *Pareto-optimal* solutions, that is, a set of non-dominated solutions that are chosen if no objective can be improved without the quality of compromising another. Weiszer et al. (2010) adopt this strategy to propose a I-TTVS model with no interlining and solve it using multi-objective GA. Ibarra-Rojas et al. (2014) propose two IP models TT and VS separately and combine these in a bi-objective integrated model, solved repeatedly using a budgeting approach.

Remarks:

In table 2.2, we synthesise the most relevant literature on integrated planning approaches. Overall, these studies highlight the recent developments in public transport optimisation.

With the advent of advanced Operations Research methodologies, researchers and practitioners are becoming more aware of the models and algorithms that aid a planning process. However, a few limitations are noted: in most studies reviewed here, the model needs to start with an already input timetable (for example, Fonseca et al. (2018); Guihaire and Hao (2008*b*); Petersen et al. (2013) where sub-trips are formed from an already given original timetable) and work around shifting/modifying it based on some criteria. In addition, where weighting objectives is a strategy, it is observed that finding accurate weights for each objective is crucial. Also, the inclusion of detailed practical considerations still remain a challenge, especially on large scale networks.

Year	Author(s)	Objectives	VSP	Decision Variables	Model	Solution Methods	Case	Network Size	Practical Considerations					
									Headway	RT_c	RT_v	PD_c	PD_v	*Cap*
(2001)	Ceder	Min. transfer waiting time	SDVSP	Departure time	IP	Heu.	Real	Large	H_e, Ht_e	✓	×	×	✓	-
(2001)	Chakroborty et al.	Min. transfer waiting time	MDVSP	Departure time & fleet size	MINLP	GA	Example	Small	Ht_e	-	-	×	✓	-
(2007)	Liu and Shen	Min. user & operator costs	MDVSP	Departure time & fleet size	BLP	N-TS	Example	Small	Ht_e	✓	×	×	×	✓
(2008*b*)	Guihaire and Hao	Min. fleet size, headway deviation; Max. transfer quality	SDVSP	Starting time & vehicle pair	Network	ILS	Real	Large	$H_{u+tol.}$	×	✓	×	×	-
(2008)	Van den Heuvel et al.	Min. operating costs	MDVSP	Departure & arrival times	Network	LS	Real	Large	Ht_e	✓	×	✓	×	✓
(2009)	Fleurent and Lessard	Min. sum of timetable & vehicle costs	-	Departure or arrival time	IP	-	-	Example	Ht_e	✓	×	-	-	-
(2010)	Weiszer et al.	Min. transfer time, fleet size	MDVSP	Departure time	MINLP	GA	Test	Small	Ht_e	✓	×	✓	×	-
(2010)	Guihaire and Hao	Max. transfer quality + Min. vehicle costs	-	Starting time & vehicle pair	MINLP	TS	Real	Large	H_u	×	✓	×	×	×
(2013)	Petersen et al.	Min. sum of user & operator costs	MDVSP	Departure time, transfer choice	Network	Meta-H	Real	Medium	Ht_e	✓	×	×	✓	✓
(2014)	Ibarra-Rojas et al.	Max. level of service; Min. fleet size	SDVSP	Departure time	MIP	ϵ-const.	Real	Medium	H_u	✓	×	×	×	-
(2015)	Schmid and Ehmke	Min. deadheads & headway deviations	SDVSP	Departure time	MIP	Meta-H	Real	Small	Ht_e	✓	×	×	×	-
(2017)	Liu and Ceder	Min. in-vehicle travel hours; Min. initial waiting time	MDVSP	Route offset time	BLP	DF	Example	Small	H_e	✓	×	✓	×	✓
(2018)	Fonseca et al.	Min. user & operator cost	MDVSP	Departure time, dwell time	MIP	Math-H.	Real	Medium	H_u	×	✓	✓	×	✓
(2019*a*)	Carosi et al.	Min. deviation from desired headways	SDVSP	Timetable, vehicle assignment	Network	Math-H	Real	Medium	H_u	-	-	×	×	✓

H_e: Homogeneous, even
Ht_e: Heterogeneous, even
H_u: Uneven; *tol.*: tolerance
RT_c: Constant run-time
RT_v: Variable run-time
PD_c: Uniform passenger demand
PD_v: Variable passenger demand
Cap: Vehicle capacity limits

IP: Integer Programming
MIP: Mixed Integer Programming
MINLP: Mixed Integer Non-Linear Programming
BLP: Bi-Level Programming
Network: Time-space network models

Heu.: Heuristic methods
GA: Genetic Algorithm
N-TS: Nesting Tabu-Search
ILS: Iterated Local Search
LS: Local Search
TS: Tabu Search
Meta-H.: Meta-heuristics
Math-H.: Matheuristics
DF: Deficit-Function based algorithm
ϵ-const.: ϵ-constraint method

✓ Considered in the model
× Not considered in the model
\- Not enough information

Source: Author's synthesis of literature review on integrated timetabling and vehicle scheduling based on the studies cited within the table

Table 2.2: Author's synthesis of integrated timetabling and vehicle scheduling

2.7 Conclusion

This chapter aims to provide an overview on the stages of public transport planning and the need to integrate multiple sub-problems. Much emphasis is laid on the transfer optimisation component of timetabling, followed by a survey on the existing approaches, limitations and challenges in solving a comprehensive timetable coordination problem. We present the identified research gaps and opportunities from the literature review in Table 2.3.

In spite of the complexity associated with transfer optimisation, the majority of existing approaches often involve simplified instances that result in vulnerable models that do not represent real world uncertainties in transit systems. Many studies focus on theoretical problems, which are great foundations to understanding the problem complexity and the performance of the solutions proposed. However, current research tends to lack the flexibility required to match the passengers' needs with that of the operators. In this context, this thesis will help to close this gap by developing a holistic mathematical model that integrates more practical guidelines into planning, aimed to improve its overall efficiency and applicability (see Chapter 4).

While there is ample research on the traditional sequential approach to transit planning, the literature is relatively silent on the various ways a planning problem can be decomposed and/or re-ordered in order to get tractable solutions without compromising the benefits of problem integration, especially on large networks. This thesis recognises this gap and aims to address it by developing a sequential but re-ordered modelling approach for timetabling and vehicle scheduling, with emphasis on timetable coordination (see Chapter 5).

The integration of consecutive steps in planning is a predominant research path explored since the last decade or so. To date, the solutions to large-scale transit network problems that include a combination of sub-problems have been mostly reliant to the use of various heuristic, meta-heuristic and matheuristic approaches where the solution search schemes are based on a collection of design guidelines, criteria established from past experiences, and cost and feasibility constraints. However, the inclusion of detailed practical considerations still remain a challenge. Especially in planning large scale networks, compromises between tractability and problems integration are among the major points of interest that invite more investigation. This thesis will address this gap by developing an integrated model for the timetabling and vehicle scheduling sub-problems (I-TTVS) with emphasis on timetable coordination. Novel to this research, a Constraint Programming based meta-heuristic search scheme is also developed to solve the I-TTVS optimally (see Chapter 6). The next chapter of this thesis outlines the research approach for addressing the research gaps identified in Chapter 2.

Collectively, the studies summarised in Chapter 2 outline the critical need to tackle the inherent complexity of the BTCP and the limitations with current practices in solving them holistically. The most important conceptual difference with our work is that we address the BTCP as a constraint solving problem, using efficient modelling and solving

Location	***Key Research Gaps***	***Research opportunities***
Section 2.3: Timetable Coordination	Current models on timetable coordination lack the flexibility required to match user and operator needs, in a real-world setting. There is a need to consider more practical guidelines into planning to improve the overall problem efficiency and applicability	Propose an efficient and applicable mathematical model for timetable coordination that has the ability to incorporate multiple, conflicting objectives and real-world constraints (see Chapter 4).
Section 2.5: Sequential Planning	There is limited research and understanding on the various ways a planning problem can be decomposed or re-ordered than the traditional sequence	Develop a sequential but re-ordered timetabling and vehicle scheduling approach with timetable coordination as a main objective, to realise more accurate scheduling solutions on large networks (see Chapter 5).
Section 2.6: Integrated Planning	Current research on integrated timetabling and vehicle scheduling for timetable coordination suffer in terms of tractability, especially on large scale networks	Develop an integrated timetabling and vehicle scheduling (I-TTVS) framework for timetable coordination such that it incorporates multiple practical considerations; develop a Constraint-Programming based meta-heuristic approach solve the problem optimally (see Chapter 6).

Table 2.3: Research gaps and opportunities

techniques capable of deriving faster solutions without the requirement to iterate between TT and VS. It is important to note here that while automating the practice of scheduling helps us efficiently handle a wide variety of scenarios with least testing time, the most efficient schedule for a service is ultimately created combining the practical knowledge of a scheduler. It is within this context that this research was undertaken to present a versatile and holistic algorithm to optimize timetables wherein planning and operation agencies can realise accurate, realistic and cost-effective solutions.

Chapter 3

Research Methodology

3.1 Introduction

Chapter 2 provided a detailed literature review of public transport timetable coordination, with an emphasis on the modelling and solving limitations with existing approaches. It also identified a set of research gaps and opportunities which act as a foundation for the design of this thesis.

This chapter gives an overview of the research methodology adopted to address the research gaps and opportunities with regard to solving the *Bus-train Timetable Coordination Problem* (BTCP). Section 3.2 introduces the case study area: the *City of Wyndham*. In Section 3.3, we describe the overall research approach adopted for this thesis, comprising three phases. Section 3.4 briefly explains the modelling and solving procedures in *MiniZinc*. In Section 3.5, we explain the experimental design adopted for the research tasks. This chapter concludes with some final remarks in Section 3.6 that sets the foundation for the subsequent chapters.

3.2 Study Area: City of Wyndham

The public transport system in Melbourne include bus, tram and train services under multiple operators. The residents of Melbourne use public transport for 57% of all trips to the Central Business District (CBD). The overall share of public transport for work trips to CBD average at 15.6% with trains accounting for approximately 7.2% (Australian Bureau of Statistics, 2016).

Melbourne has a radial train network comprising 16 lines and 207 stations servicing Melbourne, Greater Melbourne and suburban regions. About 346 bus routes cover two thirds of its metropolitan area (Currie and Loader, 2010). Serving almost half of the daily PT trips in Melbourne, buses and trams play a pivotal role as feeder services to trains, enabling wider origin-to-destination connections. With around 250,000 bus passenger trips on a regular weekday, the bus services in Melbourne remain a popular choice of travel where access to other modes is limited. Due to the monocentric nature of the city, Melbourne

PT trips involve a significant number of transfers. While approximately 94,000 tram-train weekday passenger journeys are made, bus-train journeys approximate at a further 99,000, that constitutes 15% of total train journeys and 29% of total bus journeys (Public Transport Victoria, 2012). Bus-train interchanges are also mostly prominent during the morning and evening peak journeys.

In Australasia, bus, rail and tram schedules are generally implemented by different operators. There are around 16 bus operators in metropolitan Melbourne and 51 in regional Victoria (Public Transport Victoria, 2020). Transport agencies often use various scheduling systems ranging from manual efforts to sophisticated automated tools to identify solutions that can balance user and operator needs. In Melbourne, at least 10 alternative methods to scheduling are observed (Currie, 2005). While timetable coordination might be a simpler problem to tackle in cities with uniform bus and train frequencies, it is highly challenging to achieve network-wide practical solutions in a multiple route, multi-operator, dynamic planning scenario like in Melbourne (Currie and Bromley, 2005). The major contributor to operator costs in Melbourne is the bus fleet size which is dictated by the number of buses that ply on the road at the busiest time of the day and bus hours (including unproductive time like deadheading and layover).

In this thesis, we focus on the *City of Wyndham*, in south-western region of Melbourne, Australia where a subset of a larger public transport network is selected as our case-study area. Figure 3.1 is a geographic representation of this sub-network in Wyndham, comprising 24 bus routes (uni-directional) intersecting at 5 train stations, which includes 2 transfer stations on the Geelong V/Line (regional rail, also considering first class trains) and 3 transfer stations on the Werribee Metro Line. For this case study, our focus is on optimising the bus schedules to improve the temporal coordination with train services, which operate according to a known, fixed timetable. Serviced by a common bus operator (CDC Melbourne), most bus routes in Wyndham have an existing average bus frequency of 20-40 minutes during weekday AM and inter-peak, targeted to meet passenger demand for many different travel purposes. Train services in Wyndham (especially the Geelong line) exhibit wide variation in Weekday AM-peaks, and the inter-peak services mostly include 20 minute frequencies. With observed headway unevenness between buses and trains at different time of the day, type of the day, location and route direction, timetable coordination is identified as a challenging task for the Wyndham network.

Listed below are a few relevant challenges presented by the Wyndham network in terms of timetable coordination:

- Disharmony between bus and train service frequencies through different time-of-day and type of day;
- Bus routes coordinating mid-route (route 153 at Hoppers Crossing station), requiring connectivity for passengers travelling from both ends of the route
- Bus routes requiring to connect in both directions (e.g, route 180 with equal coordination priority at Tarneit and Werribee)
- Connection time requirements[1] specific to the transfer stations

[1]For example, 5-10 minute connection times per bus route and transfer station

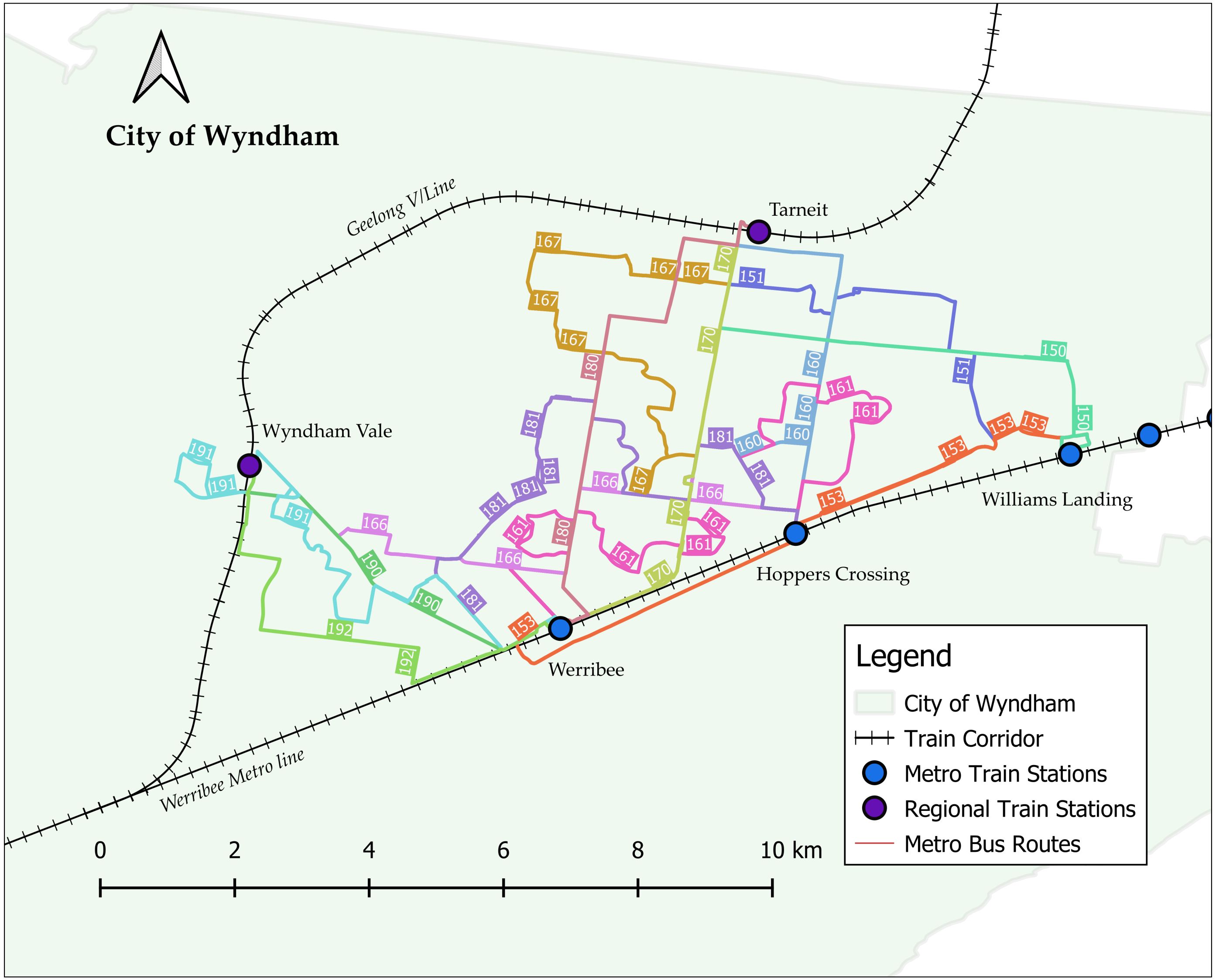


Figure 3.1: The complete network scope comprising 24 bus routes and 5 train stations in the City of Wyndham

3.2.1 Data Inputs

A range of primary and secondary data sources including bus service specifications and coordination requirements for this research are provided by the Department of Transport (DoT), Victoria (Public Transport Victoria, 2014). In addition, the data on PT networks, planned timetables, passenger volume and necessary service level adjustments were extracted at the initial stages of this research from various sources. Table 3.1 gives an overview of the main data sources, inputs and their characteristics.

Data Source	Data Input	Description
DoT Timetables	Runtime	Bus runtime on each route
	Bus Trips	Number of bus trips on each route
	Layover time	Taken as 10% of runtime as recovery time at the end of each trip
	Train timetable	Train departure time from each station
Google Maps	Deadhead time	Inter-route travel time time between stations with no passengers
	Walk time	Walking time between bus stop and train departure platform
Smartcard (Myki) data	Transfer passenger volume	Hourly bus-train transfer passenger volume at each transfer station
DoT Service Specifications	Target Headway	Planner defined target headways for each bus route at different time periods of the day
	Coordination priority	Planner defined priority transfer stations for coordination
	Waiting time range	Planner defined waiting time ranges at each transfer station

Table 3.1: Data setup and source

3.2.2 Problem Instances

We choose a schedule horizon from 7:00AM to 3:00PM (combining the AM-peak and inter-peak trips) for weekday CBD[2] bound bus trips from Wyndham suburbs, considering a time period when bus to train transfers are prioritised the most. A certain fleet of buses are scheduled over the given routes and directions. A minimum layover of 10% of route running time is allocated as the recovery time for each route, at the end of each of its trips. We allow a bus headway tolerance of ±20% from the specified target headway per route to enable the bus trips to be aligned more closely with the train departures. Note that some routes (for example, 150_A, 151_A, 153_A, 160_A and 190_A) in reality, have secondary coordination priority with CBD bound trains in the AM to inter-peak time period and are used to satisfy the headway constraints for all interlining routes. Melbourne uses a contact-less, integrated smart card ticketing system called *myki* for electronic fare payments on public transport services in Melbourne and regional Victoria. Keeping up with the period of service specifications, we use an aggregated data of hourly bus to train passenger transfers over the weekdays collected over the month of June 2017 (30 days). We study the case of two network instances in Wyndham with varying sizes:

i. *Small* (6 *X* 4): 6 bus routes (106 total bus trips) and 4 coordinating train stations;
ii. *Large* (24 *X* 5): 24 bus routes (368 total bus trips) and 5 coordinating stations

An overview of our network data inputs is presented in Table 3.2 with industry specific parameters and tolerances.

[2] Central Business District

Bus route	Route direction	Trips per route	Target headway AM & inter Peak (mins)	Variable headway AM-Peak Min-Max (mins)	Variable headway inter-Peak Min-Max (mins)	Coordinating station (No.)	Priority	Coordinating train line	Route layover time
150	In	15	20 & 40	16-24	32-48	Williams Landing (2)	1°	Werribee Metro	
150_A	Out	15	20 & 40	16-24	32-48	Tarneit (1)	2°	Geelong V/Line	
151	In	12	40 & 40	32-48	32-48	Williams Landing (2)	1°	Werribee Metro	
151_A	Out	12	40 & 40	32-48	32-48	Tarneit (1)	2°	Geelong V/Line	
153	In	12	40 & 40	32-48	32-48	Hoppers Crossing (3)	1°	Werribee Metro	
153_A	Out	12	40 & 40	32-48	32-48	Hoppers Crossing (3)	2°	Werribee Metro	
160	In	15	20 & 40	16-24	32-48	Hoppers Crossing (3)	1°	Werribee Metro	
160_A	Out	15	20 & 40	16-24	32-48	Tarneit (1)	2°	Geelong V/Line	
161	In	12	40 & 40	32-48	32-48	Hoppers Crossing (3)	1°	Werribee Metro	
161_A	Out	12	40 & 40	32-48	32-48	Werribee (4)	1°	Werribee Metro	
166	In	12	40 & 40	32-48	32-48	Hoppers Crossing (3)	1°	Werribee Metro	10% of route runtime
166_A	Out	12	40 & 40	32-48	32-48	Wyndham Vale (5)	2°	Geelong V/Line	
167	In	12	40 & 40	32-48	32-48	Hoppers Crossing (3)	1°	Werribee Metro	
167_A	Out	12	40 & 40	32-48	32-48	Tarneit (1)	1°	Geelong V/Line	
170	In	24	20 & 20	16-24	16-24	Werribee (4)	1°	Werribee Metro	
170_A	Out	24	20 & 20	16-24	16-24	Tarneit (1)	1°	Geelong V/Line	
180	In	24	20 & 20	16-24	16-24	Werribee (4)	1°	Werribee Metro	
180_A	Out	24	20 & 20	16-24	16-24	Tarneit (1)	1°	Geelong V/Line	
190	In	24	20 & 20	16-24	16-24	Werribee (4)	1°	Werribee Metro	
190_A	Out	24	20 & 20	16-24	16-24	Wyndham Vale (5)	2°	Geelong V/Line	
191	In	10	40 & 60	32-48	48-72	Werribee (4)	1°	Werribee Metro	
191_A	Out	10	40 & 60	32-48	48-72	Wyndham Vale (5)	2°	Geelong V/Line	
192	In	12	40 & 40	32-48	32-48	Werribee (4)	1°	Werribee Metro	
192_A	Out	12	40 & 40	32-48	32-48	Wyndham Vale (5)	2°	Geelong V/Line	

Table 3.2: Characteristics of the PT sub-network in Wyndham

3.3 Research Approach

Table 3.3 briefly re-states the research gaps and opportunities for this thesis and links it to the key research objectives as listed in Chapter 1. It also indicates the corresponding thesis chapters where the key contributions are presented. In addressing the research objectives, this research proceeds in three phases:

- Phase 1: Establishing the research background (RO_1)
- Phase 2: Model formulation and optimisation (RO_2, RO_3)
- Phase 3: Model testing, validation and evaluation (RO_4)

The research approach adopted for this thesis is shown in Figure 3.2, that creates a systematic modelling, solving and testing framework based on inferences from best practices on public transport timetable coordination. The major tasks and outcomes from each research phase are also shown in this figure and accordingly detailed in the following sections.

3.3.1 Phase 1: Establishing the Research Background

This phase primarily involved the following tasks to establish a sound research background that would thereby set the foundation for the rest of the thesis.

(a) Literature Review

As presented in Chapter 2, literature review was conducted to explore the limitations with conventional public transport planning processes. Since this thesis concentrates only on the timetabling and vehicle-scheduling components of planning, studies on the remaining planning stages (for example, network design, frequency setting, driver scheduling and rostering, real-time control) were excluded. Much emphasis was then laid on reviewing the existing compromises with modelling and solving the timetable coordination problem in a real-world setting. The most important findings from this review that form the key research gaps and opportunities are re-stated in Table 3.3.

(b) Industry Discussions

As stated in Chapter 1, this thesis results from a collaboration with transit agencies and has an active industry participation. The most essential modelling requirements and data inputs for this thesis were inferred from the service specifications provided by our industry partners at the Department of Transport (DoT)-Victoria, Melbourne. Following detailed discussions with the DoT, a range of coordination constraints that can render practically useful and operable scheduling solutions were identified. In this regard, we prioritised those constraints that have the most potential impact on decision making and are subject to change by the schedulers, adapting to the dynamic nature of transit demand over a period of time. The discussions also helped us understand the real-world industry practices on timetable coordination, and the challenges, limitations and opportunities for further improvement in transit planning and scheduling in general.

Research Gaps	***Research Opportunities***	***Research Objectives****	***Research Phases (Contributions)***	***Thesis Chapters***
Current models on the BTCP exhibit a lack of flexibility in balancing the user and operator needs in a real-world setting. There is a need to consider more practical guidelines into planning to improve the overall problem efficiency and applicability *(Section 2.3)*	• Explore the most essential real-world scheduling requirements that represent the problem practically and can render cost-efficient scheduling solutions • Propose a mathematical model for the BTCP that has the ability to incorporate multiple, conflicting user and operator objectives and real-world constraints that is applicable in a practical scenario	RO_1: To understand the existing challenges with BTCP RO_2: To mathematically formulate the BTCP with real-world requirements	Phase 1 Phase 2 (C_1)	Chapter 4
There is limited research and understanding on the various ways a planning problem can be decomposed or re-ordered than the traditional sequence *(Section 2.5)*	Develop a scalable optimisation approach to solve the BTCP with re-ordered timetabling (TT) and vehicle scheduling (VS) sub-problems	RO_2: To mathematically formulate the BTCP with real-world requirements RO_3: To solve the BTCP using state-of-the-art optimisation techniques	Phase 2 (C_2)	Chapter 5
Current research on integrated timetabling and vehicle scheduling (ITT-VS) for timetable coordination suffer in terms of tractability, especially on large scale networks *(Section 2.6)*	Develop an optimisation approach to solve the BTCP with ITT-VS such that it incorporates multiple practical considerations and is scalable to large PT networks		Phase 2 (C_3)	Chapter 6
	Validate and compare the optimised schedules using those generated by an existing commercial scheduling software	RO_4: To evaluate the quality of optimally coordinated timetable	Phase 3	Chapter 7

* *The research objectives framed in Chapter 1 are adapted in each phase in the context of the corresponding research gaps*

Table 3.3: Relationship between key research gaps, opportunities, associated research objectives, research phases and the corresponding chapters in this thesis

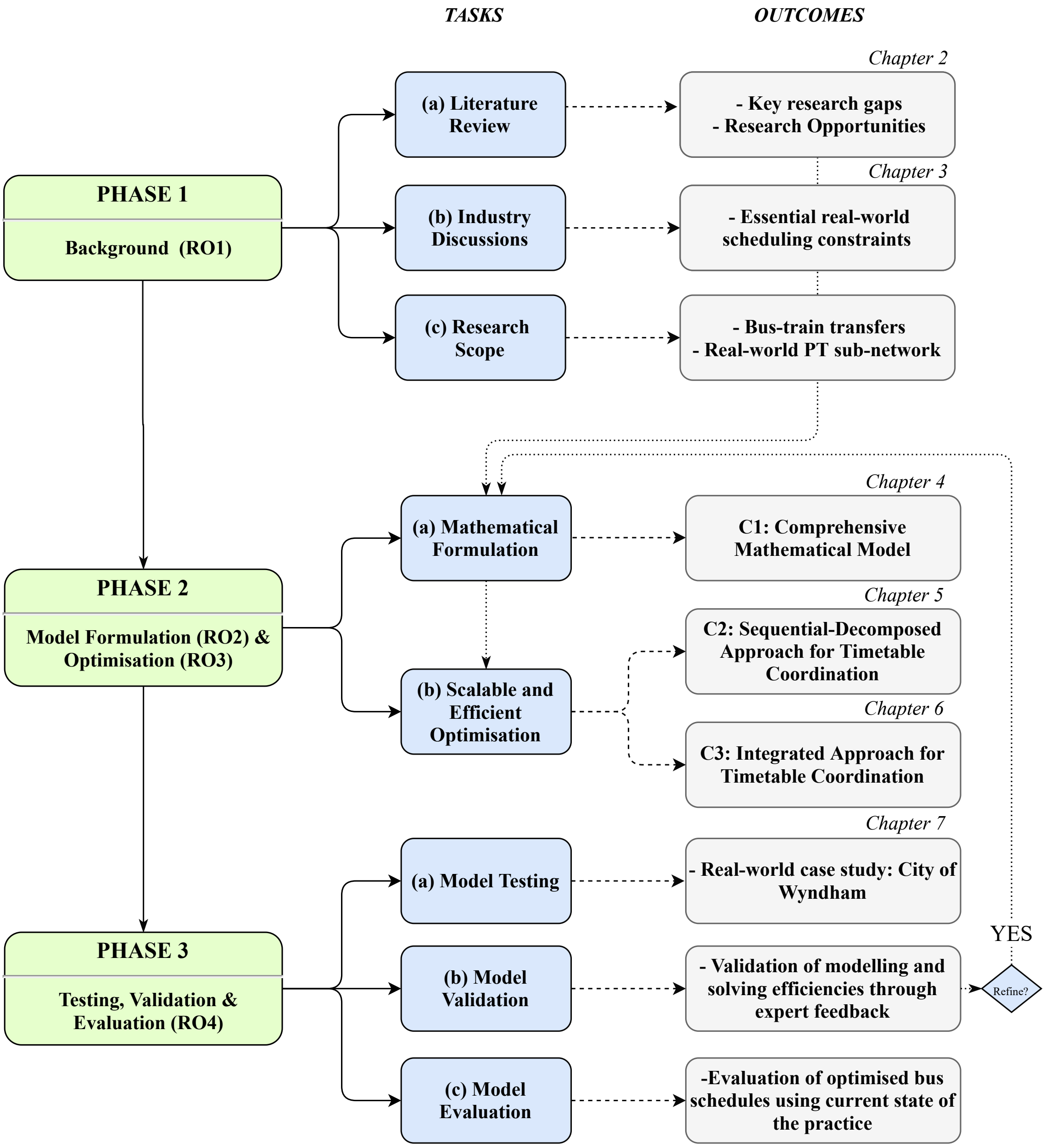


Figure 3.2: Research Approach

(c) Research Scope

The initial scope of this research looked at modelling timetable coordination between buses, trains and trams, but due to the vastness and complexity of the resulting problem, the scope was narrowed down to incorporate buses and trains alone. This was based on acknowledging the opportunity to improve the serviceability of buses as feeder systems for trains, supported by the fact that bus-train transfers are the most common type of PT transfers in Melbourne (Currie and Loader, 2010). Disharmony in bus and train frequencies at any given time of the day makes it highly challenging to achieve efficient bus-train coordination at a reasonable cost in the City of Wyndham, making it a promising candidate network to test our model's efficiency in rendering improved scheduling solutions.

Further details on the tasks, methods, data analysis and outcomes arising from Phase-1 are presented in Chapter 4, where we also elaborate the sub-set of scheduling constraints chosen to model the bus-train timetable coordination problem.

3.3.2 Phase 2: Model Formulation and Optimisation

In addressing RO_2 and RO_3, this phase primarily involved the formulation of real-world scheduling constraints inferred from Phase 1 into mathematical models for the BTCP and the design of scalable and efficient optimisation techniques to solve it, respectively. The contributions from this phase are three-fold:

(i) a comprehensive mathematical formulation for the BTCP such that it is practically applicable, incorporating a wide range of real-world constraints (C_1)

(ii) a two-stage optimisation approach that enalbes the re-ordering of timetabling and vehicle scheduling sub-problems (C_2) and

(iii) a meta-heuristic approach that enables the integration of timetabling and vehicle scheduling sub-problems (C_3)

(a) Mathematical formulation

In Section 2.3.3, we detail the rationale behind addressing the BTCP as a complex optimisation problem. In this regard, an abstract mathematical model was formulated characterised by the model parameters, objectives, decisions and constraints that are of prominence to this study. Much focus was laid on prioritising those scheduling constraints that are of core value to modelling the BTCP on a real-world setting. Note that these constraints were mainly explored in a context of bus-train transfers in Melbourne PT networks, but the developed mathematical model is abstract enough to allow applications by any mode and any city-related data.

(b) Constraint Optimisation

A *model* in the context of constraint optimisation problems comprises a set of decision variables, a set of parameters, an objective (which is an expression involving decision variables and parameters) and a set of constraints on the parameters and decision variables. A solution is an assignment of values to the decision variables that satisfy all the constraints, and yield a value for the objective. An *instance* to the problem is one where the input parameters are given and fixed. To summarise, consider the following building blocks of a constraint optimisation problem:

V : set of all variables representing decisions to the problem

C : set of all constraints over the variables V

D : the domain of a variable $v \in V$ such that D represents a set of possible values for v

f : an objective function to the problem

Thus, an instance to a constraint optimisation problem can be denoted as (V, C, D, F). A solution s to an optimisation problem is a feasible assignment of a value from domain D to each variable $v \in V$ belonging to the model such that s satisfies all constraints $c \in C$. For a minimisation problem, an optimal solution is s to the objective f such that $f(s)$ is minimal.

The main optimisation objective for this research is to ensure that the prescribed service levels on a PT network is delivered with minimum total passenger waiting time and minimum operator cost (that is governed by the fleet size, excessive layover etc). We developed a high-level constraint-based model of the BTCP in *MiniZinc*, and mapped it to various alternative solving approaches including integer-linear programming, constraint propagation and search, and learning while searching. The best of these approaches were capable of deriving solutions to variously sized network coordination problems in reasonable computation times. More details on *MiniZinc* modelling are provided in Section 3.4.

3.3.3 Phase 3: Model Testing, Evaluation and Validation

In order to answer RO_4, the outcomes from phases 1 and 2 were considered here and several criteria were laid down to evaluate the quality of these outcomes. It involved the following three tasks:

(a) Model Testing and Evaluation

We tested the developed models on our case study area in the City of Wyndham whilst increasing the network size incrementally. We examined the performance of the developed modelling and solving techniques in producing well-coordinated, cost-efficient scheduling solutions that are also scalable to larger network sizes.

(b) Model Validation

In this task, we analysed the extend to which the developed model is satisfactory in deriving well-coordinated and cost-efficient scheduling solutions, while also rendering itself usable in practical scenarios. Expert industry feedback from the DoT were sought to validate the quality of the optimised schedule from Phase 2. This enabled a deeper understanding of the relevance of the decision variables, constraints, objectives and implicit assumptions chosen in the model. Whenever necessary, any issues that interfered with representing the actual problem were identified and re-modelled in Phase 2.

(b) Model Evaluation

Finally, we investigated benefits of improved timetable coordination in terms of passenger service and bus resource requirement in the case study area. We also evaluated the quality of *MiniZinc* optimised solutions using *NetPlan*, a commercial scheduling tool. The purpose of this exercise is to indicate that the performance of our models and associated solvers in finding good-quality solutions for large-scale problems are at least comparable to the best available commercial solutions.

The numerical results and evaluation outcomes from this phase are explained in detail in Chapter 7.

3.4 Modelling in *MiniZinc*

Dedicated Constraint Modelling Languages have become popular to support problem specific modelling in a solver-independent way (Wallace, 2020). We choose *MiniZinc*[3], an expressive constraint modelling language to model and solve the BTCP due to its ability to achieve high level of problem abstraction and support for over 20 different solvers. *MiniZinc* is independent of underlying solvers and can interface easily to those by translating an input model and data file into solver specialised *FlatZinc* models (Nethercote et al., 2007). Additionally, *MiniZinc*'s ability to support a variety of global constraints makes it easy to define the complex relationship among multiple decision variables. *MiniZinc* provides a front-end to mathematical (integer-linear) solvers such as Gurobi[4]; for constraint programming solver such as Gecode[5]; a learning-while-search solver called Chuffed[6].

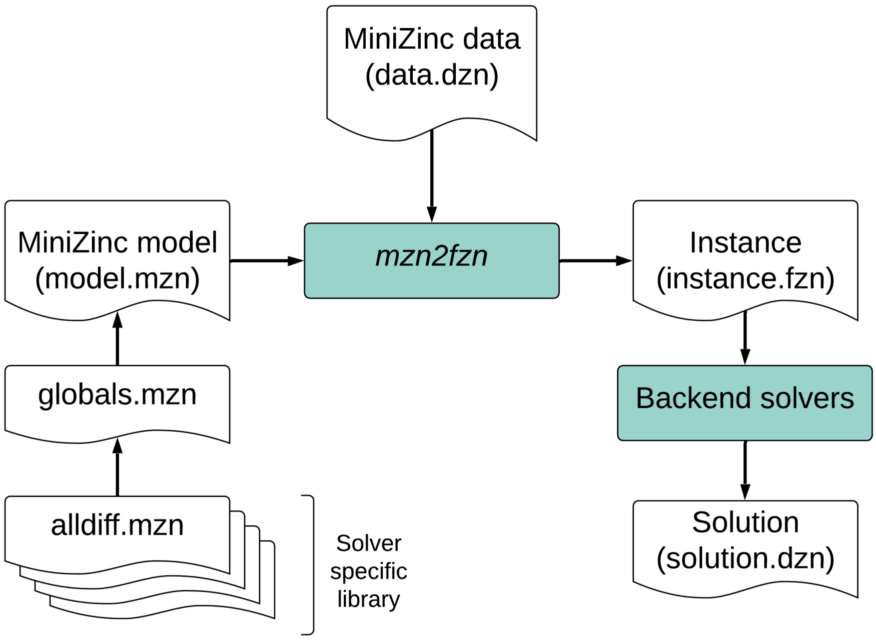


Figure 3.3: MiniZinc Compilation Process

Figure 3.3 shows a conceptual illustration of the compilation process in *MiniZinc*. A problem specified in *MiniZinc* has two parts: the *model* formalised by the user and the *data* that serves as the input data for the model. A conceptual pairing of a model with a particular data-set creates a *model instance* (or simply, an *instance*). The model and data

[3]https://www.minizinc.org/index.html
[4]https://www.gurobi.com/
[5]https://www.gecode.org/
[6]https://github.com/chuffed/chuffed

are often separated, to improve the re-usability of the model for different classes of similar problems. Note that there are two broad classes of problems in *MiniZinc*: satisfaction and optimisation (that is, maximize or minimize an objective function). A model can also include library specified *global constraints*, which define some complex relationships between multiple variables. Using information from the library, model parameters and the data file, the compiler then creates a solver specific *FlatZinc* instance, a process also called *flattening*. The MiniZinc-to-FlatZinc compiler (*mzn2fzn*) simplifies the model to a flat list of decision variables, simplified constraints and a solve item that can be comprehended by the solvers alone. A multitude of solver-specific FlatZinc instances can be created from a single *MiniZinc* model, making it a versatile and powerful tool.

3.5 Experimental Design

In this Section, we briefly explain the experimental design for the research tasks explored so far, with a primary focus on Phase 2. The developed optimisation model was enhanced and expanded incrementally, incorporating a wider range of planning and operational constraints as identified from Phase 1. Figure 3.4 shows the overall experimental setup.

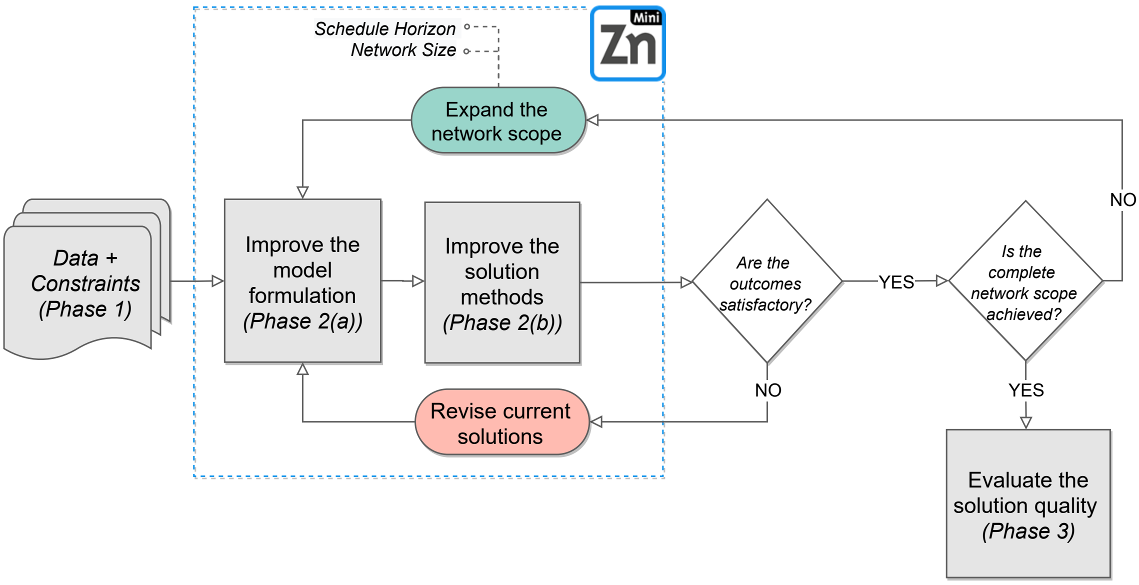


Figure 3.4: Experimental setup showing the incremental expansion of research scope

3.5.1 Pilot Optimisation Model with test data

As a pilot study, we formulated a small-scale optimisation model for timetable coordination in *MiniZinc* for a network comprising only one train station and one connecting bus route. The objective was to minimise the excessive waiting time between the arrival of a bus and the departure of the next train in the given network. The model characteristics are briefly summarised in Table 3.4. While the train departure time was used as an input, the scheduling constraints were limited to the following:

(i) 20 mins and 40 mins peak and off-peak fixed headways, respectively
(ii) Fixed time for the bus to complete a trip and return to a station
(iii) Every bus is constrained to meet with the next train
(iv) A waiting time range of 5-10 minutes; any value greater than 10 was considered excessive

Input Data	Bus fleet size, bus trips, cycle time, peak and non-peak headways; Number of trains; Walk time
Decision Variables	Bus arrival time and number of bus trips
Constraints	(i) Peak and non-peak headway constraints (ii) Bus turnaround constraint (iii) Next train constraint (iv) Bus-train meet constraint within a transfer time range
Objective Function	Minimise excessive wait time between bus arrival and next train departure

Table 3.4: Pilot optimisation model

The developed *MiniZinc* model satisfied the constraints and was able to check how well we can establish connectivity given a certain bus fleet size and fixed headway. While this exercise was an important start to designing our optimisation model, it omitted a number of constraints that represent the transit system in reality.

3.5.2 Expanded Scope

In the next step, we expanded the pilot scheduling model further to accommodate the bus-train coordination requirements from the City of Wyndham. The most important expansion was to have multiple routes and stations such that bus interlining is enabled. In reality, the BTCP is subject to a range of constraints that vary by time-of-day, day of week, route direction and location. We incrementally added the following constraints into our optimisation model to represent this:

– time-of-day dependent bus headway that harmonise with trains
– time-of-day dependent transfer passenger volume
– operator imposed constraints (deadhead time, layover time)
– bus route interlining

We explored a number of ways to revise and expand the model whilst not compromising with the most relevant scheduling requirements. As shown in Figure 3.4, we explored the expansion of our problem in the following temporal and spatial dimensions respectively:

i. Schedule Horizon: the schedule horizon for planning activities were expanded incrementally depending upon the modelling and solving capabilities to produce feasible solutions. We began with testing our models on a time period of 6:00AM to 9:00AM that represents the morning peak. This was then incrementally widened from 6:00AM to 3:00PM, representing the trips from morning and inter-peak periods.
ii. Network Size: we began with testing our model on a small-scale network with 6 bus routes and 4 transfer stations; this was later expanded to 12 bus routes and

5 transfer stations while also refining the model to be scalable. The expansion continued incrementally until the target network of 24 bus routes and 5 transfer stations was achieved.

At each iteration, the model was tested and validated with expert feedback for further improvements, until the complete research scope was met. Once feasible and satisfactory results were obtained, we proceeded to evaluate the quality of optimised scheduling solutions as briefly explained in Phase 3(c).

3.6 Conclusion

This chapter aims to describe the overall research approach adopted to address the gaps in knowledge and research opportunities identified in this thesis. In doing so, three research phases are outlined. The next Part II presents the *Optimisation Framework* to model and solve the BTCP and comprises three chapters: the mathematical formulation for the BTCP incorporating the most relevant constraints and objectives is presented in Chapter 4; detailed descriptions of multiple optimisation techniques adopted to solve the BTCP are provided in Chapters 5 and 6.

Part II

Optimisation Framework

Chapter 4

Comprehensive Mathematical Model

4.1 Introduction

As detailed in Chapter 2, the global problem of public transport (PT) planning is concerned with offering efficient level of service to the passengers while maintaining minimised operator costs. When addressed as a whole, this problem is often not solvable to optimality and is generally decomposed into various planning stages viz., strategic, tactical, operational and real-time control (Desaulniers and Hickman, 2007). With an overarching aim to solve the *Bus-Train Timetable Coordination Problem* (BTCP), we model the tactical and operational levels of public transport planning with multiple decisions concerned with determining cost efficient *timetabling* (TT) and *vehicle scheduling* (VS) solutions.

Inferring the major gaps in research from Chapter 2 this chapter collectively addresses research objectives RO_1 and RO_2, which are re-stated as follows:

RO_1 : To understand the existing challenges with bus-train timetable coordination

(1a) What are the limitations with existing approaches in solving the BTCP?

(1b) What modelling constraints represent the problem realistically?

RO_2 : To formulate the BTCP incorporating real-world scheduling constraints.

Contribution

This chapter presents contribution C_1, that is, a comprehensive mathematical model for the bus-train timetable coordination problem based on a set of real-world requirements. In particular, we investigate various constraints that can represent the timetable coordination problem realistically, including those scheduling requirements that are not yet considered in the literature before.

We briefly explore a generic model for public transit transfers and associated costs but the scope of our main model is limited to the modes of buses and trains alone. The remainder of this chapter is structured as follows: in Section 4.2 we provide a generic setting of the transfer coordination problem in a mode-independent scenario. Section 4.3

describes the main contribution from this chapter, a mathematical model to address the problem of bus-train timetable coordination (BTCP) with multiple decisions, variables and constraints. The overall objective of this model is to achieve a favourable trade-off between the contrasting objectives of achieving timetable coordination at minimum operator costs, subject to a variety of operational and practical requirements. Section 4.4 then demonstrates the complex nature of the model through direct solving with commercial solvers. Section 4.5 briefly outlines the various ways this model can be extended to accommodate additional operational scenarios. Finally, this chapter concludes in Section 4.6 with a discussion of model limitations.

4.2 Generic Model for Public Transport Transfers

In this section, we briefly describe the two main public transport planning stages of relevance to this study- *Timetabling* (TT) and *Vehicle Scheduling* (VS) and formulate the traditional sequential model for these sub-problems, with transfer optimisation as a major objective. A certain level of abstraction is used in the description of these sub-problems such that the model is generic enough to be applicable to any PT modes, independent of the specific constraints used. Note that the problems described below are assumed to be bounded. However, infeasibility may occur due to the inter-dependencies of multiple sub-problems.

4.2.1 Timetabling

Timetabling is a *tactical* planning problem that determines the specific time for certain events to occur. In the context of public transport planning, these events are the departure and arrival times of a vehicle at a time-point. The inputs to this process are decided at the previous stages of network design and frequency setting, that include the network structure comprising mainly transit service lines (bus routes, train lines etc.) and time-points (bus stops, train stations etc.), the desired line frequencies and running times between major time-points and any recovery time built into the schedule. The output from this stage, that is, the timetables are used to initiate the subsequent stages of vehicle scheduling and driver scheduling.

As seen in Chapter 2, many variants of the timetabling problems exist that rely mainly on the type of decisions and constraints considered to solve certain objective(s). A general objective of timetabling is to maximise the passenger service, which commonly include timetable coordination wherein the total passenger transfer waiting time between multiple scheduled services are minimised. In this thesis, we deal with the group of timetabling problems that consider passenger transfers and aim to minimise the total passenger transfer waiting time for a given scheduling period. Accordingly, the solutions developed from our models use known, hourly passenger transfer volume at a set of pre-defined transfer locations.

Given a public transport network (PTN) with a set of routes N_r and their desired frequencies N_f known and fixed, let x^{TT} be our decision variables concerned with the timetabling problem. Then, a generic formal description of the problem is as follows:

$$\begin{aligned}(N_rN_f \longrightarrow TT) \quad \min \quad & Z_{TT} = f_{TT}(x^{TT}) \\ \text{s.t.} \quad & x^{TT} \in F_{TT}(N_rN_f)\end{aligned} \tag{4.1}$$

where $(N_rN_f \longrightarrow TT)$ indicates that given the PTN inputs on routes N_r and their desired frequencies N_f, solve the desired timetabling objective f_{TT} and output an optimal timetable x^{TT} within the set of all feasible timetables F_{TT}. The timetabling variables x^{TT} denote the arrival and departure times of all vehicles at all stations in the given network.

4.2.2 Vehicle Scheduling

Vehicle scheduling (VS) is an *operational planning* problem where a chain of vehicle trips are created according to a given set of timetables. Each such trip chain is called a *vehicle schedule* and the entire chaining process is referred to as *vehicle blocking*; a block is a sequence of productive (active or live trip time) and unproductive trips (empty trip time: deadheads[1], pull-outs[2] and pull-ins[3]) performed with a bus. The primary objective of this sub-problem is to minimise resource costs, namely vehicle acquisition and associated operational costs in dispatching those vehicles.

Given a set of routes and their trip timetables known and fixed, let x^{VS} be our decision variables concerned with the vehicle scheduling problem. A generic formal description of the problem is then as follows:

$$\begin{aligned}(N_rx^{TT} \longrightarrow VS) \quad \min \quad & Z_{VS} = f_{VS}(x^{VS}) \\ \text{s.t.} \quad & x^{VS} \in F_{VS}(N_rx^{TT})\end{aligned} \tag{4.2}$$

where, $(N_rx^{TT} \longrightarrow VS)$ indicates that given the PTN inputs on routes N_r and their timetable solutions x^{TT}, solve the desired vehicle scheduling objective f_{VS} and output the set of all feasible vehicle schedules F_{VS}. The vehicle scheduling variables x^{VS} denote the sequence of all vehicle trips in the network (or *vehicle blocks*), according to the given set of timetables.

4.2.3 Timetable Coordination

As seen in Section 2.3, *timetable coordination* is identified as the most challenging process in timetable development (Desaulniers and Hickman, 2007). Well coordinated timetables ensure that the arrival time of a vehicle on one route is synchronised temporally with that on another route so that passengers can transfer between these routes seamlessly, at minimum waiting times. From an operational point of view, transit agencies wish to utilise the available resources judiciously, by minimising or restricting the required fleet size and associated vehicle and driver costs. Ideally, a planner (or an automated planning system) accomplishes this by performing modifications to the timetables during the vehicle and/or

[1]The time that a bus travels without any passengers; typically when buses need to travel inter-route
[2]The time the bus spends travelling from a depot to the start of an active trip
[3]The time the bus spends travelling from the end of an active trip back to the depot

driver scheduling stages. While such decomposition may render a large problem solvable, it often negates the dependence that each planning stage has on each other.

There are many cases in the literature where timetabling modifications are performed to an existing scheduled timetable (Chakroborty et al., 1997; Guihaire and Hao, 2010; Schröder and Solchenbach, 2006) to ensure that the optimised timetable do not vary drastically from the original. However, in this thesis, we generate new timetables based on a set of practical rules and do not consider existing schedules to initiate the optimisation.

A cost effective scheduling solution comprises a trade-off between user costs and operator costs. In transit, a demand-supply equilibrium can be expected when transit timetables are constructed or modified to accommodate daily varying passenger demand at minimum cost for the users, while the judicious use of operational resources is maintained. We define the user costs, also called "transfer" costs in this study as the sum of the total excess transfer waiting time for all passengers transferring between two given services in the network. The operator costs in this study are dictated by the minimum number of buses that are necessary to cover the set of timetabled trips in the given network.

Here, we provide a generic illustration of an inter-modal[4] transfer between two given transit routes R_i and R_j. Let these routes intersect at a transfer location S (Figure 4.1). Let us assume that the number of people seeking to transfer from R_i to R_j is known in any given hourly period. Within a certain schedule horizon, we consider first, the case where the timetable for one of the routes R_j is fixed and that of the other route R_i is to be optimised to minimise costs. In a generic setting, we aim to model the gap between these arrivals and departures that is, the total excessive minutes of waiting between the two routes incurred by the transferring passengers.

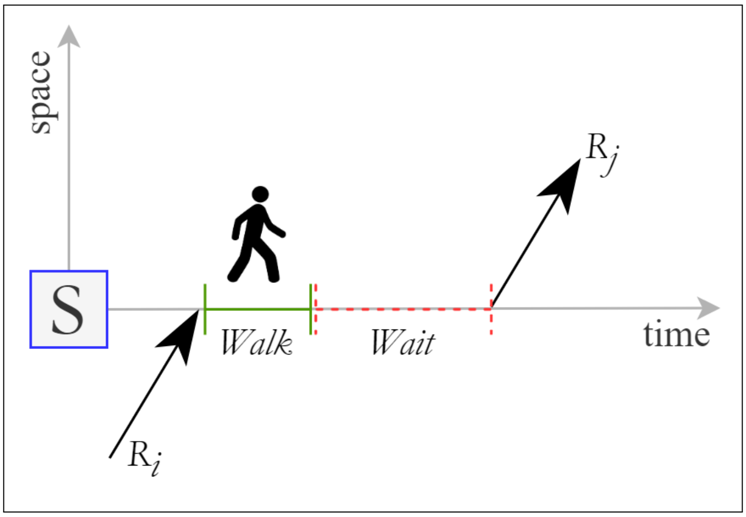


Figure 4.1: Schematic diagram depicting passenger transfer between two public transport lines

[4]Refers to the movement of passengers/cargo by several modes of transport where each of these modes have a different transport operator with individual contracts

Transfer Costs

Let $P_{i,j}$ be the number of passengers seeking to transfer between trips on routes R_i and R_j. As shown in figure 4.1, the minimum time required to make a successful transfer from R_i to R_j is the *walk* time. Any time gap greater than this walk time is considered excessive *wait* time, and added to our cost function $w_{i,j}$, as given below:

$$w_{i,j} = Dep_{R_j} - (Arr_{R_i} + walk_{i,j}) \tag{4.3}$$

The overall cost of a transfer can be computed by multiplying the wait time $w_{i,j}$ associated with each connection pair (R_i, R_j) with the number of passengers $P_{i,j}$ seeking to transfer, that is:

$$C^{tc}_{i,j} = w_{i,j}.P_{i,j} \tag{4.4}$$

Note that the calculations above hold true when the transfer flow is reversed, that is, from R_j to R_i (Equations 4.5 and 4.6).

$$w_{j,i} = Dep_{R_i} - (Arr_{R_j} + walk_{j,i}) \tag{4.5}$$

$$C^{tc}_{j,i} = w_{j,i}.P_{j,i} \tag{4.6}$$

Operator Costs

When it comes to scheduling vehicles, the optimum timetabling solutions developed for this objective determine the optimum vehicle fleet size and the allocation of a sequence of trips to each vehicle using known and fixed hourly passenger demand at a set of transfer locations. In this context, the primary objective that we consider with creating vehicle schedules is to minimise the number of vehicles to be used while considering unproductive service times.

4.3 Bus-Train Timetable Coordination Problem (BTCP)

In this section, we present the mode specific scenario for timetable coordination. The case of bus-train transfers is of core interest to this thesis, where the train timetables are fixed and bus timetables are optimised to minimise the transfer and operator costs. The ***Bus-Train Timetable Coordination Problem*** can be seen as a schedule optimisation problem to determine bus schedules that minimise the excessive passenger wait time when connecting with trains, considering minimum bus fleet size and efficient service regularity requirements in a given planning period.

Predominantly, the complete problem of timetable coordination between buses and trains involves the following timetabling and vehicle scheduling decisions-

i. the number of buses required
ii. the sequence of routes followed by a bus for each of its trips

iii. the specific train met by each bus trip
iv. the arrival time of each bus at the corresponding transfer station

and the following two main objectives-

a. Minimise total passenger wait time
b. Minimise the number of buses required

A favourable trade-off between the two main objectives is obtained by solving the problem for each bus fleet size (decision type (i)) separately. Thus, for each fleet size, a minimum excessive passenger waiting time is computed. However, the combination of decision types (ii), (iii) and (iv) which minimise the passenger waiting time is hard to compute. As shown in Equation (4.4), when transfer passenger demand flow must also be considered, the objective (a.) includes a term that multiplies the number of transferring passengers by the transfer waiting time. Since both passenger demand and transfer waiting time depend upon the decisions on departure time for each vehicle trip (decisions (ii) and (iii)), non-linearity is introduced in the objective. This makes timetable coordination a large and complex combinatorial optimisation problem, even for small transit networks.In addition, we model the VS with an aim to determine a feasible set of bus schedules where (i) each active trip is covered by exactly one vehicle schedule and; (ii) the number of buses required are kept at a minimum value.

To make the above decisions, the following problem inputs are necessary: (i) the *bus trips per route* including the industry specified *desired headway* during peak and off-peak periods; (ii) the *timetables* known and fixed for coordinating train lines; (iii) a bus *run-time* matrix comprising its journey time and deadhead[5] (iv) the *transfer passenger volume* between bus routes and corresponding trains (v) *transfer walk time* from alighting a bus to the transfer station; (vi) a minimum and maximum *bus layover*[6] *time* at the end of each route.

Based on the information presented above, the BTCP in a given schedule horizon *determines the allocation of each input number of buses to routes and trips such that it reaches its target coordinating train station in time to meet with a train, to minimise any excessive passenger waiting time maintaining a minimum bus fleet size and service regularity requirements in a given schedule horizon.* The cost function for BTCP thus comprises:

- cost of transfers, that is, the total excessive passenger wait time for all bus route-station pairs in the network (determined by x^{TT}).
- costs for the bus schedule, that is, the cost of using minimum number of buses to run the timetabled trips considering unproductive service time like layovers and deadheads (determined by x^{VS}).

[5]The time that a bus travels without any passengers, typically when it needs to travel inter-route
[6]The time between multiple bus trips, usually to allow for some recovery time or driver meal breaks

4.3.1 Constraints

We base the BTCP on a set of real-world scheduling requirements as follows that also frame the comprehensive mathematical model:

Headway policies

Headway, defined as the time between two consecutive vehicle trips on a route acts as a major factor influencing passenger transfer times and hence the overall commute time. For different time periods of the day (eg., peak, off-peak), and for different directions of bus travel (eg., peak, contra-peak[7]) transit agencies generally determine a target service frequency (or headway) to define bus operating schedules. In planning cases where the headways are constant and demand flow is uniform through different time periods in a day, timetabling is a relatively easier problem to solve (Ceder and Wilson, 1986). The first vehicle departure is set (based on some clock time relative to each time period, say 7:00am) and the consecutive vehicle departures are computed as increments of the desired service headway (say 7:20am, 7:40am, 8:00am etc for a 20 min headway). It is a common practice to calculate the waiting times as half the scheduled headway (Van Oort and van Nes, 2004). However, when it comes to integrating two services like the bus and train, this rule does not apply well when there is significant headway variability. If the arrival and departure times of connecting services at a transfer point are coordinated, there is potential to reduce the waiting time to much less than half the scheduled headway.

When trains are not in high frequency (especially during off peak), it is not always beneficial to define a perfectly even bus headway and the effort will be to specify headway for interlining bus routes that will be harmonised with the trains. Some tolerance could be allowed to ensure that maximum number of trains are met, while also allowing the headway to transition smoothly between peak and off-peak periods well. We keep headways within a desired tolerance Δh[8] from the target to enable better coordination and vehicle scheduling efficiencies that would not be possible with strict even headway.

Headway bounds must be selected with caution as we want minimum headway deviation from the target headway, so as to ensure service regularity. It must be noted that flexible headway can result in bus bunching[9] and we avoid this by specifying a minimum headway for the first bus (Li et al., 2020). In contrast, longer headway values could see long intervals in a planning period without an active trip arrival or departure. We avoid this by specifying an end time to our schedule horizon.

[7] Contra-peak refers to the PT services that run in the opposite direction to the direction of the highest passenger volumes

[8] The acceptable minimum and maximum bus headway values are inferred from operator defined target headway.

[9] The phenomenon when two or more buses on the same route which were scheduled with a certain headway, arrive simultaneously at a place

Coordination priority

The transfer coordination priority at each station is generally pre-determined by transit agencies based on historical demand trends, bus route direction and time of the day. In the AM-peak, most trips are work based that originate at suburbs and extend to the CBD (Central Business District). Hence we observe a high transfer demand from bus to train in the morning. With a flip time at around 3:00 PM, when most passengers are home-bound, the transfer demand is from train to bus. We model the AM and inter peak bus-train transfers from 7:00AM to 3:00PM (note that train to bus connections are not optimised due to lower priority during the morning peak). We consider one primary priority transfer station per bus route direction that may be at the end of a route or mid-route. For example, Figure 4.2 shows the bus to train and train to bus hourly transfer volume for route 150 connecting at Williams Landing station over a given weekday.

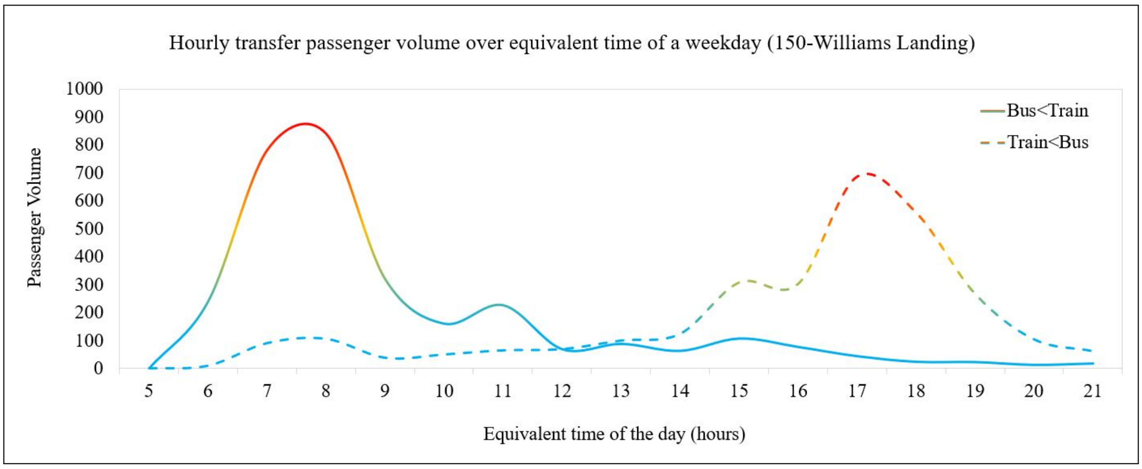


Figure 4.2: Transfer passenger volume from/to route 150 at Williams Landing station over equivalent time of a weekday

Time-of-day dependent transfer passenger volume

The number of passengers transferring between PT services differ significantly through peak and non-peak hours in a day. Incorporating time-of-day dependent passenger transfer volumes into the optimisation objective is an important aspect to realise schedules that cater to the demand needs in reality. Moreover, with variability in bus headway, the transfer waiting time incurred by different groups of passengers will be different. Hence, unlike the traditional approach of considering an average or aggregate value for transfer volume for a given time of the day, we calculate the proportion of transfer volume between two consecutive bus arrivals from a given hourly volume at each station. Figure 4.3 illustrates how we model transfer volumes by time of the day.

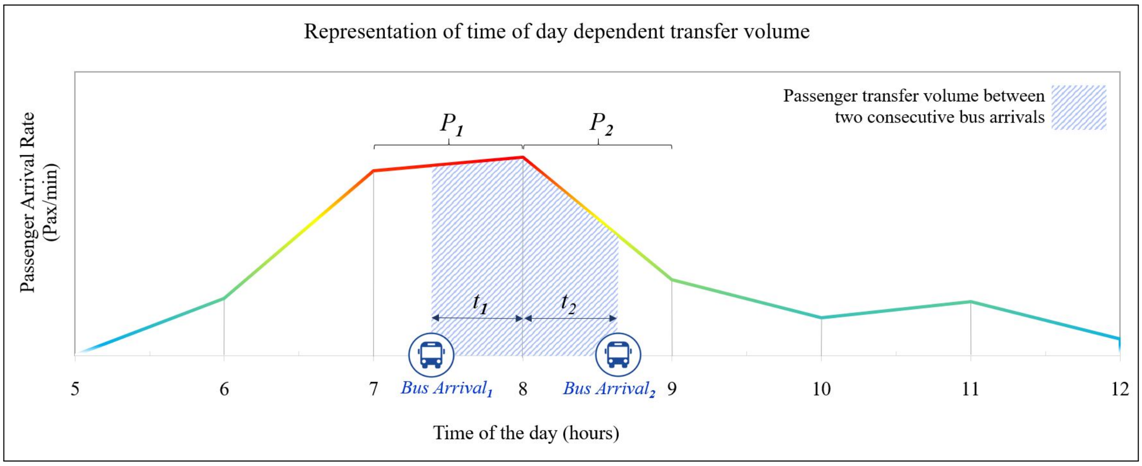


Figure 4.3: Representation of time-of-day dependent transfer volume

From the figure, the transfer volume P between consecutive bus arrivals 1 and 2 is given as:

$$P \geq (P_1 * t_1) + (P_2 * t_2) \tag{4.7}$$

where, P_1 and P_2 are the passenger arrival rates at hour 1 and hour 2 respectively (passengers/min) and $t_1 + t_2$ is the time interval between bus arrival 1 and 2, t_1 in the first hour and t_2 in the second, relative to the schedule start time.

Operational resource constraints

In general, bus scheduling is subject to resource limitations such as the requirement of fleet size, fixed bus dispatching times for the first and last trips, fixed number of trips per day etc. The major cost drivers among these are the fleet size, dictated by how many buses are needed to service a given network at the busiest time of the day; and bus service hours, that add to the cost of driver wages. In the case of large-scale PT networks, savings in operator cost can be much more apparent, since efficient optimisation has the potential to create an opportunity to run the entire network with at least one less bus than the existing fleet size.

In order to improve coordination and use a bus fleet over multiple routes efficiently, it is important to consider operator imposed unproductive time involved with a bus trip (Fleurent et al., 2004). Each bus route in the network has a specified running time between transfer stations. On completion of a trip on a given route, the bus then has a layover time before the start of its next route. This acts as a buffer to account for driver breaks or late bus arrivals (owing to a variety of external factors such as congestion, road conditions, delayed departures etc). In the simplest case, the next route starts at the location where the previous route ended. As an exception, however, the next route might start from a different location. This is called *"interlining"*. If an improved coordination efficiency can be gained by allocating a different location to the same bus, this would require some inter-route travel time, also called the *deadhead time*. We add deadhead time between each pair of routes as a conscious decision to support the optimisation model. Henceforth, the term *"run-time"* means the sum of bus route-running time and deadhead time between a pair of bus routes.

Consider a simple network segment as shown in Figure 4.4. Let's assume that a bus completes its first route A at station S_A and starts its next route B at station S_B. The bus finishes route A, deadheads from S_A to S_B, has a certain minimum layover at S_B before proceeding to perform its next route B (to station S_C and so on). The time between the start of route A and the start of route B is given as:

$$Run_{\mathrm{AB}} = (rt_{\mathrm{A}} + dh_{\mathrm{AB}}) + lay_{\mathrm{A}} \tag{4.8}$$

where,

rt_{A} is the running time for route A

dh_{AB} is the bus deadhead time between routes A and B

lay_{A} is the layover time at the end of route A

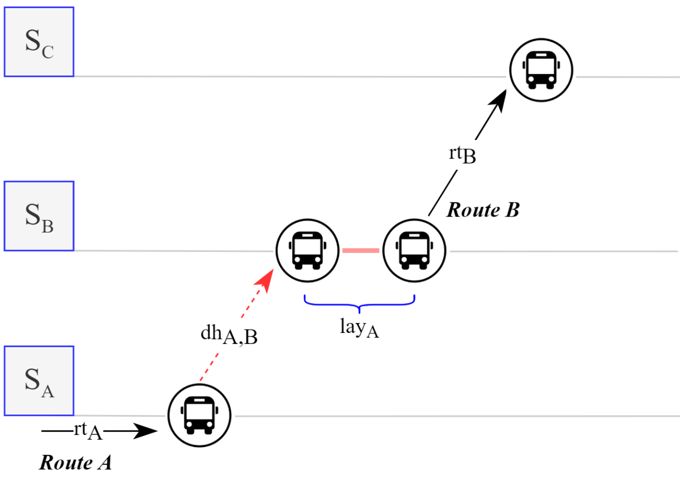


Figure 4.4: Duration Run_{AB} between the end of the route A and the end of the route B

It is a challenging task for schedulers to assign blocks of bus trips while ensuring good coordination, especially on low-frequency routes. We allow the buses to freely interline[10] that is, buses are not constrained to operate only on one route. Thus, a single bus could be used over a variety of routes maximising connections with trains, while the sequence of routes made by a bus is selected by the optimiser. Naturally, there is a constraint for a bus that the time required between completing one route and starting the next is greater than the sum of the layover time plus the deadhead time. If this time is indeed more than the layover plus deadhead time, then this difference in time is considered unproductive for both the bus and its driver, and is accordingly minimised by the optimiser. Nevertheless, it must be noted that due to additional constraints on bus fleet size, bus route headway and oordination requirements, the overall optimum solution will generally include unproductive time for some bus trips.

[10] A resource-efficient strategy where a certain fleet of buses are scheduled over a variety of routes; enables a single bus to perform multiple routes over a given period of time

4.3.2 Optimisation Goals

Considering the above mentioned scheduling requirements for the Wyndham bus network, let us briefly recall our main research question:

Can we generate timetables and vehicle schedules that optimise both timetable coordination and operator cost efficiency simultaneously?

To answer this question, we noted the following optimisation goals:

- Timetable Coordination:
  - Minimise the excessive passenger waiting time for every feasible bus-train meets.
  - Maximise the number of trains met by bus trips. With restrictions on headways at any given time of the day, there is limited opportunity to ensure that all trains are met in a schedule horizon. However, the maximum gap between connecting trains can be minimised such that there are less number of passengers disadvantaged with more waiting.
- Fleet size and operator costs:
  - Minimise the cost to deliver a defined service level and/or maximise service levels within a defined budget such that there is a reasonable trade-off between user and operator requirements. With cost minimisation as a major objective, the devised optimisation ensures that the prescribed service levels on a PT network is delivered with minimum number of buses (considering unproductive service time) at a given time-of-day.
- Service regularity:
  - Minimise excessive variations in bus headways compared to the desired service targets.

4.3.3 Model Formulation

In this section we present the proposed mathematical formulation for the BTCP. Consider a simplified illustration of a bus-train connected network $G = \{S, R\}$ as shown in Figure 4.5. Bus routes $\{R_1, R_2, R_3...\}$ with a separate route in each travel direction are defined between coordinating train (or transfer) stations $\{S_1, S_2, S_3...\}$. Passengers arriving on buses alight at the transfer station, walk to the platform and transfer to the next available train. We set the end of the operational schedule horizon such that there is always such a train. The minimum time required to make a successful transfer from a bus arrival to the next available train is the *transfer walk time*. Assuming that the passengers are captive with no alternative travel option, we aim to minimise the sum of all *transfer waiting time* that is greater than the given walk time between bus arrivals and train departures.

The most relevant notations used in formulating the model are as follows:

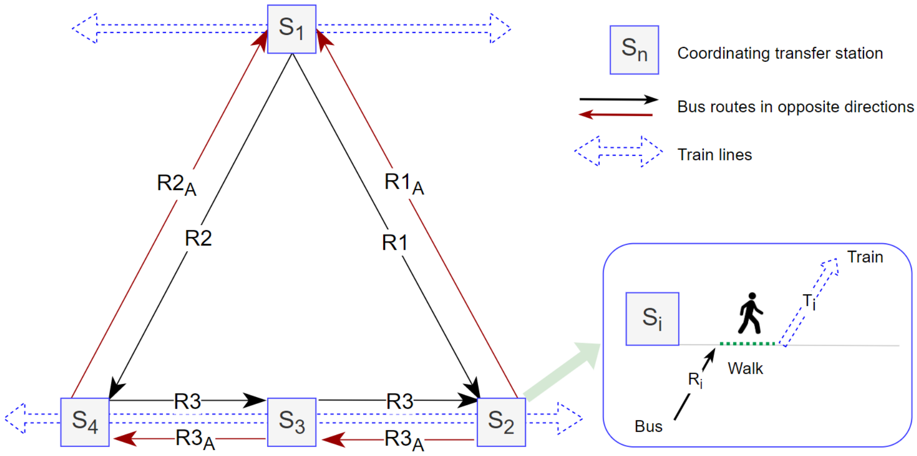


Figure 4.5: Illustration depicting a bus-train passenger transfer

Input Data:

T_0	:	the beginning of schedule horizon.
T_{max}	:	the end of schedule horizon.
R	:	set of all single-direction bus routes in the network, $R = \{r_1, r_2, ...\}$.
TR_r	:	set of all bus trips on each route r in the schedule horizon, $TR_r = \{t_1, t_2, ...t_{n_r}\}$.
S	:	set of all transfer stations in the network, $S = \{s_1, s_2, ...\}$.
B	:	set of all buses in the network, $B = \{1...b_{ct}\}$.
H	:	set of all hours in the schedule horizon, $H = \{h_1, h_2, ...\}$.
b_{ct}	:	the number of buses.
n_r	:	the number of bus trips for each route within $[T_0, T_{max}]$
f_r	:	the coordinating station served by each route r
Run_{r_1,r_2}	:	the run-time between start of the route r_1 and start of the route r_2.
hw_r^-	:	the minimum bus headway on route r, $r \in R$.
hw_r^+	:	the maximum bus headway on route r, $r \in R$.
min_r^{lay}	:	the minimum layover time for a trip on route r, $r \in R$.
max_r^{lay}	:	the maximum layover time for a trip on route r, $r \in R$.
$walk_r$	:	the walk time between bus arrival and connecting train, $r \in R$.
$T_{s,j}^{dep}$	:	the train departure times for all trains $j \in 1..pt$ from a station s, $s \in S$.
$P_{h,r}$	:	the hourly passenger transfer volume for route r, $h \in H$, $r \in R$.

Decision Variables:

The problem is formulated using the following decision variables and variables dependent on these:

(i) $flow_{r_1,t_1,r_2,t_2}$: binary variable indicating that a bus trip (r_1, t_1) immediately precedes (r_2, t_2); $(t_1 \in TR_{r_1}, t_2 \in TR_{r_2})$

(ii) $B_{r,t}^{start}$: integer variables indicating the start time of bus trip t on route r; $(r \in R, t \in TR_r)$

(iii) $B^{hour}_{r,t}$: integer variables indicating the hour within which each bus on route r and trip t arrives; $(r \in R,\, t \in TR_r)$

Dependent Variables:

(i) $Pax_{r,t}$: the number of passengers transferring from bus trip t on route r to a train (calculated between two consecutive bus hours), $(r \in R,\, t \in TR_r)$

(ii) $T^{met}_{r,t}$: the physical train that is met by a bus trip t on route r, $(r \in R,\, t \in TR_r)$

(iii) $wait_{r,t}$: the waiting time between a bus arrival and the next train departure, $(r \in R,\, t \in TR_r)$

(iv) $passwait_{r,t}$: the total transfer passenger waiting time for each route r and trip t, $(r \in R,\, t \in TR_r)$

Assumptions:

We consider the following model assumptions:

- transferring passengers from buses will always board the next available train at the coordinating station.
- the bus and train services run as per schedule and any service disturbances and uncertainty at the operational stage is disregarded.
- trains and buses have sufficient capacity to meet the transfer passenger demand
- the train dwell time at a transfer station is assumed to be negligibly small

Bus flow conservation constraints:

Consider a bus $b \in B$ traversing between route r_1 on trip t_1 and route r_2 and trip t_2. We define bus flows between routes such that all buses $b \in B$ have a starting point $1, 0$ and an ending point r, t, where $r \in R$ and $t \in TR_r$. Constraint (4.9) ensures that the total flow between routes and trips is less than or equal to the bus fleet size b_{ct} (that is, the input number of buses). Constraint (4.10) is the "possible" flow constraint where the flow between route-trip pairs r_1, t_1 and r_2, t_2 is set to 1 if and only if the same bus $b \in B$ performs both trips. For conservation of flows, constraint (4.11) to (4.15) ensures that all "impossible" bus flows are set to 0; trip 0 on all routes is a dummy trip and trip 0 on route 1 is the first "trip" performed by all buses. The number of flows from trip 0 on route 1 gives the minimum number of buses used in the schedule (given by Constraint (4.9)). No bus performs trip 0 on any other route, which is given by Constraint (4.11)). Bus flows to and from non-existent bus trips are given by Constraints (4.12) and (4.13) respectively. Constraint (4.14) indicate the flows that violate the trip order $t1$ to $t2$ and Constraint (4.15) shows the bus flows from a trip to itself.

$$flow_{1,0,r,t} \leq b_{ct} \qquad \forall r \in R, t \in TR_r \quad (4.9)$$

$$\sum_{r_2}^{R} \sum_{t_2}^{TR_r} flow_{r_1,t_1,r_2,t_2} = 1 \qquad \forall r_1 \in R, t_1 \in TR_r \quad (4.10)$$

$$flow_{r_1,0,r_2,t_2} = 0 \quad \text{where,} \quad r_1 > 1 \qquad \forall r_2 \in R, t_2 \in TR_r \quad (4.11)$$

$$flow_{r_1,t_1,r_2,t_2} = 0 \quad \text{where,} \quad t_2 > n_{r_2} \qquad \forall r_1, r_2 \in R, t_1, t_2 \in TR_r \tag{4.12}$$

$$flow_{r_1,t_1,r_2,t_2} = 0 \quad \text{where,} \quad t_1 > n_{r_1} \qquad \forall r_1, r_2 \in R, t_1, t_2 \in TR_r \tag{4.13}$$

$$flow_{r,t_1,r,t_2} = 0 \quad \text{where,} \quad t_2 \leq t_1 \qquad \forall r \in R, t_1, t_2 \in TR_r \tag{4.14}$$

$$flow_{r,t,r,t} = 0 \qquad \forall r \in R, t \in TR_r \tag{4.15}$$

Bus run-time and interlining constraints:

Constraint (4.16) ensures that for all possible flows, that is, if the same bus $b \in B$ does two consecutive start trips, there must be a minimum run-time $Run_{r1,r2}$ between the two, with the allocation of a minimum and maximum layover time $[min_{r2}^{lay}, max_{r2}^{lay}]$. Since our model allows bus trips to freely interline, we do not precisely identify which route must succeed or precede the other. Hence, a bus $b \in B$ can perform r_1, t_1 before r_2, t_2 or vice-versa.

$$flow_{r_1,t_1,r_2,t_2} = 1 \implies \begin{cases} B_{r1,t1}^{start} + Run_{r1,r2} + min_{r2}^{lay} \leq B_{r2,t2}^{start} \\ B_{r1,t1}^{start} + Run_{r1,r2} + max_{r2}^{lay} \geq B_{r2,t2}^{start} \end{cases} \quad \forall r_1, r_2 \in R, t_1, t_2 \in TR_r \tag{4.16}$$

First and last bus constraints:

Since headways are not fixed, the buses could become bunched around the time of greatest passenger demand, leaving no buses to meet the earliest and latest trains in the schedule. As a span of hours requirement, all bus arrivals on route r given by $B_{r,t}^{start}$ plus its run-time Run_r must fall between $[T_0, T_{max}]$, that represents the complete schedule horizon. Constraint (4.17) limits the first bus trip t on any route r to arrive at a time between the beginning of the schedule horizon T_0 and a certain maximum headway hw_r^+ since.

$$T_0 \leq B_{r,1}^{start} + Run_r < (T_0 + hw_r^+) \qquad \forall r \in R \tag{4.17}$$

Similarly, Constraint (4.18) limits the last trip t of a bus on route r to arrive at the station at within its maximum headway hw_r^+ from the end of the schedule horizon T_{max}.

$$B_{r,t}^{start} + Run_r >= T_{max} - hw_r^+ \qquad \forall r \in R, t \in TR_r \tag{4.18}$$

Headway constraint:

Constraint (4.19) ensures that the headway between consecutive bus trips on a given route route r is constrained to fall between a minimum and maximum, $[hw_r^-, hw_r^+]$.

$$hw_r^- \leq (B_{r,t+1}^{start} - B_{r,t}^{start}) \leq hw_r^+ \qquad \forall r \in R, t \in TR_r \tag{4.19}$$

Feasible connection constraints:

For successful meets with trains, Constraint (4.20) ensures that the next bus arrives only after the previous train has departed. Constraint (4.21) verifies that the last bus arrives before the last train departs. Similarly, Constraints (4.22) and (4.23) restrict earlier bus trips on a route to meet with earlier trains.

$$B^{start}_{r,t} + Run_r > T^{dep}_{s,j-1} \quad \text{where,} \quad s = f_r \qquad \forall r \in R, t \in TR_r, s \in S, j \in 1..pt \quad (4.20)$$

$$B^{start}_{r,t} + Run_r <= T^{dep}_{s,j_{last}} \quad \text{where,} \quad s = f_r \qquad \forall r \in R, t \in TR_r, s \in S \quad (4.21)$$

$$T^{met}_{r,t} < T^{met}_{r,t+1} \qquad \forall r \in R, t \in TR_r \quad (4.22)$$

$$T^{dep}_{s,j} \geq B^{start}_{r,t} + Run_r + walk_r \quad \text{where,} \quad s = f_r \qquad \forall r \in R, t \in TR_r \quad (4.23)$$

Passenger volume constraints:

The passenger volume constraints (4.24) to (4.26) compute the proportion of transferring passengers between consecutive bus arrivals at a station. As the first trip departure time is not fixed, $Pax_{r,t}$ is a variable dependent on bus arrival times and corresponding bus arrival hours. It is defined for three cases:

i. for the first bus trip within hour $B^{hour}_{r,1}$, we take the combined transfer volume proportionate to this hour and any hour h prior to it:

$$Pax_{r,1} = P * (B^{start}_{r,1} + Run_r - B^{hour}_{r,1}) + \sum_{h<B^{hour}_{r,1}} P_{h,r} \qquad \forall r \in R \quad (4.24)$$

ii. for all the remaining bus trips, the proportion of transfer passenger volume is computed based on the gap between two consecutive bus arrivals in current hour $B^{hour}_{r,t}$ and previous hour $B^{hour}_{r,t-1}$, respectively:

$$\text{if, } t = 2..n_r, \qquad \begin{aligned} Pax_{r,t} = [P * ((B^{hour}_{r,t-1} + 1) - B^{start}_{r,t-1} + Run_r) - \\ P * ((B^{hour}_{r,t} + 1) - B^{start}_{r,t} + Run_r)] \\ \forall r \in R; t \in TR_r; h \in H \end{aligned} \quad (4.25)$$

iii. 0, if the bus arrival is null that is, beyond the schedule horizon.

$$\text{if, } t > n_r, \quad Pax_{r,t} = 0 \qquad \forall r \in R; t \in TR_r \quad (4.26)$$

Wait time constraint:

Constraint (4.27) computes the waiting time as the interval between train departure and bus arrival with the inclusion of walk time. The passenger transfer waiting time is then calculated as the sum of all excessive extra waiting time incurred by transferring passengers (Constraint (4.28)).

$$wait_{r,t} = T^{met}_{r,t} - (B^{start}_{r,t} + Run_r + walk_r) \qquad \forall r \in R, t \in TR_r \quad (4.27)$$

$$passwait_{r,t} = Pax_{r,t} * wait_{r,t} \qquad \forall r \in R, t \in TR_r \quad (4.28)$$

Null constraint:

We add Constraints (4.29) and (4.30) to take care of the null events, that is, the events that fall beyond the schedule horizon T_{max}. N_{pt} is the dummy train that buses meet with and N_{time} indicates the dummy bus arrival or train departure time.

$$B^{start}_{r,t} = N_{time} \quad \text{where,} \quad t > n_r \qquad \forall r \in R, t \in TR_r \quad (4.29)$$

$$T^{met}_{r,t} = N_{pt} \quad \text{where,} \quad t > n_r \qquad \forall r \in R, t \in TR_r \quad (4.30)$$

Objective:

Considering the most relevant scheduling aspects of BTCP as listed above, the objective function (Equation (4.31)) to minimise the total passenger transfer waiting time across all routes and trips in the PT network whilst using a minimum bus fleet *bct* can be given as:

$$\min \quad Z = \sum_{r=1}^{R} \sum_{t=1}^{TR_r} passwait_{r,t} \qquad \forall r \in R, t \in TR_r \quad (4.31)$$

4.4 Direct Solving

To investigate the scalability of the complete (or integrated) BTCP model, we tested the *Small* network instance with 6 bus routes and 4 transfer stations using the Gurobi (MIP) and Chuffed solvers. All experiments were run on a personal computer with Intel Core i7-6700 @ 3.40GHz CPU and 16GB RAM. As reported in Table 4.1, the model is solvable within a maximum time-limit of 4 hours. Moreover, the proof of optimality is also obtained for higher number of buses (8,9,10). However, for the *Large* network instance, no feasible solutions were obtained even with higher fleet size when tested for a computational time of 24 hours. The tested solvers could also not prove infeasibility for this instance with fewer input number of buses.

Network Instance	No.of buses	*Chuffed*		*Gurobi*	
		Obj.	CPU time	Obj.	CPU time
Small	6	66,796*	4h00m	x	x
	7	8,856*	4h00m	10,662*	4h00m
	8	3,140	0h52m	3,140	1h15m
	9	2,764	0h01m	2,764	0h05m
	10	2,764	0h04m	2,764	0h07m

*: *Non-Optimal solutions*
x: *No solutions found*

Table 4.1: Comparison of objective values for the complete BTCP

This indicates that due to its combinatorial nature, the BTCP can be prohibitively large for standard optimisation techniques to find solutions quickly. The available solvers require computational effort that, in the worst case, grows exponentially with the size of the problem. Consequently, for models of larger problems it soon becomes impractical to complete a proof of optimality and it is unknown whether a solution returned by the solver is optimal or not. Therefore, the challenge here is to search for feasible and good quality solutions of such large problem instances in a reasonable time-frame.

4.5 Model extensions

The proposed mathematical model can be further extended and enhanced to solve a set of additional planning and operational scenarios such as the following:

(1) 24 hour schedule horizon: The model can be extended to a full-day first to last bus scenario, which includes pre-AM-peak, AM-peak, inter-peak, PM-peak and evening. For each time block, it is necessary to understand the coordination priority direction (bus to train or vice-versa) and the flip time between peak and inter-peak periods. This requires allowing ample flexibility in trip arrival window at the end and start times of a planning period to enable smooth trip transitions.

(2) Objective weights: The current model optimises the objective of wait times against bus resource constraints. This can be enhanced to a weighted-objective problem, where certain weights represent the priority of each objective against the other (say, prioritise minimising operator costs). The full objective would then comprise the sum of these weighted objectives.

(3) Capacity constraint: The current model assumes ample capacity for trains and buses to meet the passenger demand over the schedule horizon (7:00 AM to 3:00 PM). While in reality, it is quite common that passengers experience rejected boarding due to limited capacity in buses, especially during morning peak hours. The model can thus be expanded to accommodate the bus capacity constraint.

4.6 Conclusions

Current models on timetable coordination lack the flexibility required to match user and operator needs, in a real-world setting. There is a need to consider more practical guidelines into planning to improve the overall problem efficiency and applicability. In this chapter, we first explained a generic model to compute public transport transfer and operator costs that worked as a framework to define our research problem. We then proposed the main contribution (C_1), a *Bus-Train Timetable Coordination Problem* formulated as a mathematical model incorporating a set of scheduling requirements that represent the problem realistically.

It must be noted that there are resources, constraints and requirements that are not captured in our model yet. There are other scheduling requirements, such as the desire to cycle buses onto routes that run past depots, consider timetables that allow for efficient driver rostering, restrict certain fleet types to certain routes, operate certain school trips, etc. Hence, the model proposed for BTCP in this chapter can be looked at as a scenario based subset of a bigger problem.

Considering this mathematical model as a scaffold, in the upcoming Chapters 5 and 6, we introduce a *Sequential-Decomposed Timetabling & Vehicle Scheduling* (SD-TTVS) and an *Integrated Timetabling & Vehicle Scheduling* (I-TTVS), respectively to solve the complete problem of BTCP. The former approach is model-intensive where the complete problem of BTCP is decomposed into solvable and compatible sub-problems; the latter approach in solving-technique intensive where we introduce a meta-heuristic search scheme to improve the solutions further.

Chapter 5

Sequential-Decomposed Approach for Timetable Coordination

5.1 Introduction

In the previous Chapter 4, we developed a comprehensive mathematical model for the *Bus-train Timetable Coordination Problem* (BTCP) using a set of real-world constraints. We explored the difficulty with solving the complete problem of BTCP on a real-world case study area and proposed two approaches to obtain scalable and good-quality solutions to the BTCP.

This chapter presents one of the two solving approaches for the BTCP where we investigate the re-ordering of public transport (PT) planning problems namely, timetabling (TT) and vehicle scheduling (VS). Referred to as *Sequential-Decomposed Timetabling and Vehicle Scheduling*, (abbreviated as SD-TTVS), this approach involves a two-stage constraint based optimisation with compatible timetabling and vehicle scheduling sub-problems. The mathematical model from Chapter 4 is decomposed here such that simpler sub-objectives are solved in the first stage of optimisation, and parts of the solution thus obtained are input into the second stage to solve the full objective of BTCP. The decomposed problems are modelled and solved entirely in *MiniZinc*[1]. The developed optimisation models are tested for scalability on our case study area in the City of Wyndham, and the computational performance of the proposed approach in yielding good-quality solutions at a reasonable time is explored.

Contributions

We present multiple decomposition strategies for the BTCP that are solved sequentially. Such decomposition is motivated by the recognition that a problem can be decomposed in a variety of ways. Also, the final objective in the first stage can be approximated in different ways, which have a big influence on the performance of the decomposition.

This chapter presents contribution C_2 that is composed of the following components:

[1]https://www.minizinc.org/index.html

(1) we present a two-stage constraint optimisation models to solve the BTCP using a range of real-world scheduling requirements inferred from the industry; the complete problem is decomposed without losing the compatibility between timetabling and vehicle scheduling sub-problems.
(2) We illustrate the applicability and performance of our approach through a set of computational experiments on a real-world case study for the bus and train services in the City of Wyndham, Melbourne.

This chapter addresses research objectives RO_2 and RO_3 as stated below, which are tailored to suit the overall contributions from this study:

RO_2 : To formulate the BTCP incorporating real-world scheduling constraints.
RO_3 : To solve the BTCP using state-of-the-art optimisation techniques.

This chapter is structured as follows: in Section 5.2 we provide a brief context for traditional PT planning procedures and explain the rationale behind re-ordering planning sub-problems. Section 5.3 describes the proposed SD-TTVS approach and Section 5.4 presents the mathematical formulation for the BTCP such that it is tailored to be solvable using this approach. In this section, we also present the two-stage optimisation models for the SD-TTVS and the associated objectives in each stage. In Section 5.5 we detail the formal translation of the model into CP, written entirely in *MiniZinc*. Section 5.6 is dedicated to describe the computational experiments and case study results. This chapter concludes with a summary on this approach as well as directions for improvement in Section 5.7.

5.2 Background

From the review of past studies presented in Chapter 2, the global problem of PT planning is concerned with maximising passenger service quality while maintaining minimum operator costs. It involves multiple sub-problems, of which, timetabling (TT) and vehicle scheduling (VS) are relevant to this study. While it is desirable to address these sub-problems simultaneously and ensure compatibility, it is often cumbersome to do so. Especially in medium to large PT network instances, finding scalable and holistic solutions is computationally challenging due to the multiple, conflicting decisions and objectives concerned with each sub-problem (Ceder, 2007; Desaulniers and Hickman, 2007). Hence, traditionally, each sub-problem is treated separately and sequentially, with the outcome of one serving as an input to the other.

Decomposing the global planning problem sequentially may render it solvable, but it often negates the inter-dependence between each sub-problems. For example, timetables are traditionally constructed to a certain pattern to maximise service utility and cost-effective vehicle schedules are developed next from these timetables. Sometimes minor re-calculations in shifting timetables can result in saving a complete vehicle. In another case where the priority is achieving well-coordinated timetables at minimum transfer waiting time, additional vehicles may need to be deployed that thereby incur more operating costs. Thus, planners usually require to iterate between these sub-problems to identify and resolve conflicts, sometimes requiring manual adjustments (Polinder, 2015).

In multi-stage optimisation for transit problems, it is necessary to ensure the compatibility between associated sub-problems in each stage. Each sub-problem needs a specific objective against which it is optimised, (for example minimising the number of vehicles). By solving individual objective functions at each step we may be able to find *locally* optimal solutions for the sub-problem in question. However, there is no guarantee that the solution to an earlier sub-problem is actually part of a *globally* optimal solution to the complete problem. Indeed by changing the objective for the earlier sub-problem, the resulting changes to the solutions to the complete problem could be made unpredictably worse or better. It is also important to note that the objective functions used in each planning stage are often approximations of what a particular study focuses on (Schöbel, 2017). For example, conventionally, timetabled trips are determined with some approximation on the fleet size and its associated cost while the real costs cannot be determined until the lower stream problems of vehicle and driver scheduling are solved. We are aware of studies by Lübbecke et al. (2019); Michaelis and Schöbel (2009); Pätzold et al. (2017); Schöbel (2017) on multi-stage optimisation approach for PT planning although a problem specific decomposition of the TT and VS with an objective of transfer optimisation is not observed in the literature.

Given the limitations with traditional planning procedures in finding compatible solutions, there is a need for an extensive computational study to understand the correlation between the sub-problems in scope. When it comes to tackling large-scale problems, there is scope and novelty in exploring the different ways to decompose the complete problem to yield feasible and good-quality solutions.

5.3 The SD-TTVS Approach

In this section, we detail the proposed *Sequential-Decomposed Timetabling and Vehicle Scheduling Approach* (SD-TTVS) to solve the BTCP. Decomposing the complete problem of BTCP into solvable sub-problems can be a poor strategy if the sub-problems at hand are highly integrated. However, it can also yield good quality solutions if an (locally) optimal solution to one decomposed component is compatible with that of the others.

In this context, the SD-TTVS approach presents a two-stage optimisation framework with solvable sub-problems for timetabling and vehicle scheduling. The overall structure of this approach is motivated by the observation that following the traditional approach of solving planning sub-problems independently and sequentially need not always produce feasible solutions. We propose multiple problem-specific scheduling options to compare the trade-off between passenger service and operator cost requirements. Novel to this study, to obtain holistic and good-quality solutions for the BTCP, the complete problem is divided into two inter-related stages. We ensure compatibility between timetabling and vehicle scheduling by using the same model for both stages. The decomposition we propose are based on the approximation of choice of trains to be met by buses such that excessive bus-train waiting time are minimised. Using these choices as inputs, the computational complexity of the complete problem can be reduced significantly such that it renders it solvable to optimality (or near-optimality for larger problems) using direct CP or MIP[2] solvers. Moreover, by ensuring compatibility between both stages, we avoid the need to iterate between sub-problems, like the studies proposed by Guihaire and Hao (2008*b*); Schöbel (2017). The SD-TTVS makes it possible to model and solve the sub-problems and their respective objectives at minimal manual intervention and least computational efforts.

5.4 Mathematical Formulation for the SD-TTVS

In this section, we present the mathematical formulation for the SD-TTVS. Concretely here, the comprehensive mathematical model from Chapter 4 is tailored in such a way that we solve the timetabling and vehicle scheduling sub-problems in two optimisation stages that are compatible with each other.

Let an inter-modal PT network be defined by a directed graph $G = \{S, L\}$ where S is the set of all transfer (train) stations and L is the set of travelling links that represent bus routes in the network. T_0 and T_{max} are defined as the beginning and end of the schedule horizon, in minutes. The set B has $1..b_{ct}$ input number of buses and all bus departures are set within the discrete time interval of $[T_0, T_{max}]$.

H is the set of all hours h within the schedule horizon. There are R set of single-direction bus routes in the network with $r \in R$. Every bus route r has an associated coordinating station s mapped by the function f_r, comprising a maximum of pt number of physical trains within the schedule horizon. The departure times $T^{dep}_{s,j}$ of all trains $j \in 1..pt$ from a station s is known and input to the problem. The minimum time required to make a successful transfer from a bus arrival to the connecting train is the walk time, $walk_r$. $P_{h,r}$ are the number of passengers seeking to transfer from each bus trip on route r on hour h. There are n_r number of trips per route r to be covered and TR_r set of all scheduled bus trips within the interval $[T_0, T_{max}]$. Run_r is the running time for a bus on route r from its origin location to the coordinating station, which in majority of cases is where the next route starts. An inter-route *deadheading* time is added to Run_r when a bus needs to travel to a different location to start the next route. We also consider a minimum and maximum bus *layover* time $[min^{lay}_r, max^{lay}_r]$ per route. Note that the actual bus journey time is deterministic, where we consider the average service time for a given route r within the schedule horizon. The minimum and maximum headways between two consecutive bus arrivals on route r are given by hw^-_r and hw^+_r respectively.

5.4.1 Key Decision Variables

We define the following key decision variables for the BTCP:

i. $B^{start}_{r,t}$: integer variables indicating the start time of bus trip t on route r, forall $r \in R$, $t \in TR_r$

[2] Mixed Integer Programming

ii. $flow_{r1,t1,r2,t2}$: binary variable indicating that a bus trip (r_1, t_1) immediately precedes (r_2, t_2), forall $t_1 \in TR_{r_1}, t_2 \in TR_{r_2}$

iii. $B^{hour}_{r,t}$: integer variables indicating the hour within which each bus on route r and trip t arrives (used for predicting the number of passengers transferring from that trip), forall $r \in R$, $t \in TR_r$

iv. $T^{met}_{r,t}$: non-negative integer variables indicating the physical train that is met by a bus trip t on route r, forall $r \in R$, $t \in TR_r$

v. $Pax_{r,t}$: the number of passengers transferring from bus trip t on route r (depending on the bus hour and the time since the previous bus), forall $r \in R$, $t \in TR_r$

vi. $wait_{r,t}$: the transfer waiting time between a bus arrival $(B^{start}_{r,t} + Run_r)$ plus a minimum walk time, $walk_r$ and the next train departure $T^{met}_{r,t}$, forall $r \in R$, $t \in TR_r$

vii. $passwait_{r,t}$: the transfer passenger waiting time for each route r and trip t, forall $r \in R$, $t \in TR_r$

The variables (i) and (iii) are related to bus scheduling decisions while the variables (iv) to (vii) are dependent on bus schedules for the timetable coordination decisions. The bus scheduling decisions of type (i) to (iii) are made in Stage-1 to generate the choice of trains to be met (decision type (iv)). In Stage-2, the bus scheduling decisions are re-computed using the choice of meeting trains known from Stage-1. Additionally, the timetable coordination decisions of types (v) to (vii) are made in Stage-2 and the full BTCP objective is thus minimised in this stage.

5.4.2 Constraints

The following constraints are of relevance to this study:

$$flow_{1,0,r,t} \leq b_{ct} \qquad \forall r \in R, t \in TR_r \quad (5.1)$$

$$\sum_{r_2}^{R} \sum_{t_2}^{TR_r} flow_{r_1,t_1,r_2,t_2} = 1 \qquad \forall r_1 \in R, t_1 \in TR_r \quad (5.2)$$

$$flow_{r_1,0,r_2,t_2} = 0 \quad \text{where,} \quad r_1 > 1 \qquad \forall r_2 \in R, t_2 \in TR_r \quad (5.3)$$

$$flow_{r_1,t_1,r_2,t_2} = 0 \quad \text{where,} \quad t_2 > n_{r_2} \qquad \forall r_1, r_2 \in R, t_1, t_2 \in TR_r \quad (5.4)$$

$$flow_{r_1,t_1,r_2,t_2} = 0 \quad \text{where,} \quad t_1 > n_{r_1} \qquad \forall r_1, r_2 \in R, t_1, t_2 \in TR_r \quad (5.5)$$

$$flow_{r,t_1,r,t_2} = 0 \quad \text{where,} \quad t_2 \leq t_1 \qquad \forall r \in R, t_1, t_2 \in TR_r \quad (5.6)$$

$$flow_{r,t,r,t} = 0 \qquad \forall r \in R, t \in TR_r \quad (5.7)$$

$$T_0 \leq B^{start}_{r,1} + Run_r < (T_0 + hw^+_r) \qquad \forall r \in R \quad (5.8)$$

$$B^{start}_{r,t} + Run_r > T^{dep}_{s,j-1} \quad \text{where,} \quad s = f_r \qquad \forall r \in R, t \in TR_r, s \in S, j \in 1..pt \quad (5.9)$$

$$B^{start}_{r,t} + Run_r >= T_{max} - hw^+_r \qquad \forall r \in R, t \in TR_r (5.10)$$

$$B^{start}_{r,t} + Run_r <= T^{dep}_{s,j_{last}} \quad \text{where,} \quad s = f_r \qquad \forall r \in R, t \in TR_r, s \in S (5.11)$$

$$hw^-_r \leq (B^{start}_{r,t+1} - B^{start}_{r,t}) \leq hw^+_r \qquad \forall r \in R, t \in TR_r (5.12)$$

Together, Constraints (5.1) to (5.7) ensure the conservation of bus flows to and from feasible trips and determine the sequence that a bus $b_{ct} \in B$ follows through multiple routes: Constraint (5.1) restricts the bus fleet size to an input value b_{ct}. Constraint (5.2) is concerned with the binary decision of setting all the "possible" bus flows to 1, representing an active bus sequence. By adding Constraints (5.3) to (5.7), we enable conservation of bus flows by setting all the "impossible" bus flows to 0. Constraint (5.8) ensures that the first bus on a route r starts within its maximum headway hw_r^+. We add a Constraint (5.9) so that the next bus arrives after the previous train has departed; this ensures that each bus meets with a different train. Constraints (5.10) ensures that all the bus arrivals fall within a time-frame of maximum headway from the end of schedule horizon and Constraint (5.11) limits the last bus to arrive before the last train departs from the related station s. Constraint (5.12) ensures that the headway between consecutive bus trips on a route r is constrained to fall between a minimum, hw_r^- and maximum, hw_r^+.

$$flow_{r_1,t_1,r_2,t_2} = 1 \implies \begin{cases} B^{start}_{r1,t1} + Run_{r1,r2} + min^{lay}_{r2} \leq B^{start}_{r2,t2} \\ B^{start}_{r1,t1} + Run_{r1,r2} + max^{lay}_{r2} \geq B^{start}_{r2,t2} \end{cases} \quad \forall r_1, r_2 \in R, t_1, t_2 \in TR_r \tag{5.13}$$

Using Constraint (5.13), we ensure two consecutive start trips t_1 and t_2 of a bus b_{ct} on routes r_1 and r_2 to be separated by running time $Run_{r1,r2}$ between these routes (which includes inter-route deadhead time for interlining) plus a range of layover time [min^{lay}_{r2}, max^{lay}_{r2}].

$$T^{met}_{r,t} < T^{met}_{r,t+1} \quad \forall r \in R, t \in TR_r \tag{5.14}$$

$$T^{dep}_{s,j} \geq B^{start}_{r,t} + Run_r + walk_r \quad \text{where,} \quad s = f_r \quad \forall r \in R, t \in TR_r, s \in S \tag{5.15}$$

$$B^{start}_{r,t} = N_{time} \quad \text{where,} \quad t > n_r \quad \forall r \in R, t \in TR_r \tag{5.16}$$

$$T^{met}_{r,t} = N_{pt} \quad \text{where,} \quad t > n_r \quad \forall r \in R, t \in TR_r \tag{5.17}$$

Constraint (5.14) is concerned with computing the choice of trains to be met by buses and restricts earlier bus trips on a route to meet with earlier trains at the corresponding transfer station. Constraint (5.15) is the transfer requirement ensuring each bus trip to arrive in time to meet its connecting train, leaving enough transfer walking time. Constraints (5.16) and (5.17) take care of the null events, that is, the events that fall beyond the schedule horizon T_{max}; the parameter N_{pt} is the dummy train that buses meet with and N_{time} indicates the dummy bus arrival or train departure time.

5.4.3 Two-Stage Decomposition and Objectives

In this section, we define two-stage optimisation models for the BTCP where variables, constraints and objectives are defined for compatible elements within the mathematical model for BTCP. Figure 5.1 shows a flowchart for the proposed SD-TTVS approach. We present five decomposition strategies z_1 to z_5, each with simplified objectives as follows:

z_1: Maximising the number of passengers on met trains
z_2: Minimising the maximum number of transfer passengers
z_3: Minimising the bus headway deviation
z_4: Minimising wait per passenger
z_5: Fixing the bus headways

These sub-objectives are individually solved in Stage-1 and the choices of trains met (T_{m1} to T_{m5}) from solving each of these objectives are recorded and fed as inputs in Stage-2. For an input number of buses, the full BTCP objective (Z) is then solved for each decomposition alternative, yielding a solution set for bus assignment and corresponding bus start times that are well coordinated with the trains. We explain in detail the two-stage optimisation models and objectives in the following sub-sections.

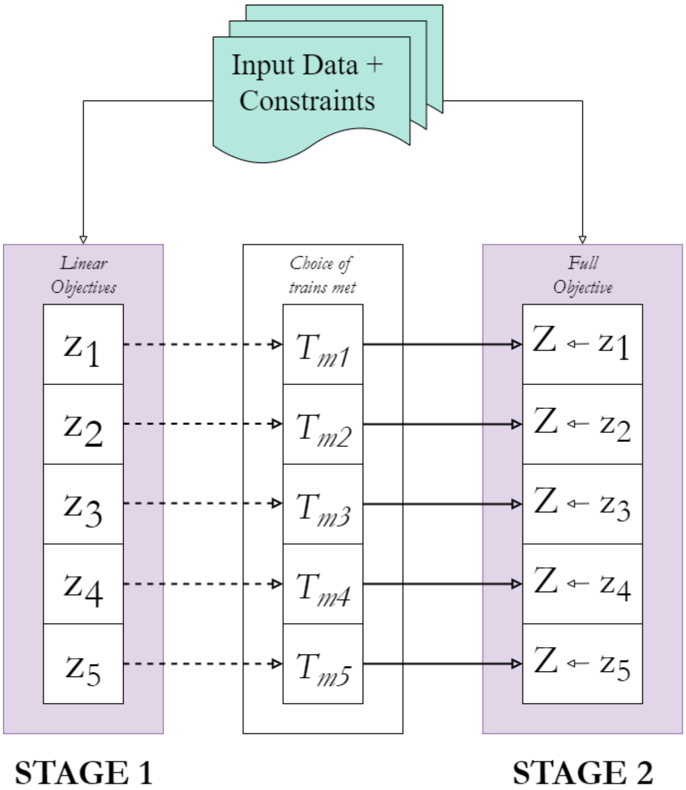


Figure 5.1: Flow chart depicting the SD-TTVS approach to solve BTCP

STAGE-1 Sub-Objectives (z_1 to z_5):

Stage-1 returns the choice of trains met by each bus on each of its route and trip. Accordingly, this stage returns a specific assignment of each bus trip to a specific train. Note that in order to optimise the trains to meet we also need to determine the optimal sequence of bus routes and associated bus start times initially, but these particular decision values are discarded and not passed to the second stage of optimisation. The trains to meet, bus allocations and corresponding bus arrival times are chosen to optimise a variety of sub-objectives that are described as follows:

(z_1) *Maximise the number of passengers on met trains*

The objective z_1 ensures that the trains met by bus trips are those with the most passengers-based on the number of passengers arriving between successive train departures in a station. That is, the number of transfer passengers from buses to trains are computed based on the desirability of a particular train to be chosen.

We compute the number of passengers in a met train, given by $Pax^{met}_{r,t}$ by considering the time gap between a given pair of train departures. Equation (5.18) presents the computation of $Pax^{met}_{r,t}$ for two cases where, $ht \in H$ is the hour of current train departure and $hpt \in H$ is the hour of previous train departure: (i) for the first train departure ($j = 1$) within hour ht, we take the combined volume proportionate to hour ht and any hour h prior to it; (ii) for all the remaining train departures, the proportion of passenger volume is computed based on the gap between two consecutive train departures in hour ht and previous hour hpt, respectively.

$$Pax^{met}_{r,t} = \begin{cases} P_{ht,r} * (T^{dep}_{s,1} - ht) + \sum_{h<ht} P_{h,r}, & \text{if, } j = 1 \\ [P_{hpt,r} * (hpt - T^{dep}_{s,j-1}) - P_{ht,r} * (ht - T^{dep}_{s,j})] & \\ \qquad + \sum_{hpt<h<=ht} P_{h,r}, & \text{else.} \end{cases}$$
$$\forall r \in R; s \in S; j \in 1..pt; h, ht, hpt \in H \qquad (5.18)$$

By choosing the trains with the most desired passengers, we minimise the number of passengers whose desired train is not met by a bus, thereby enabling the minimisation of any excessive waiting time. Objective z_1 across all routes and trips can thus be formulated as:

$$\max \quad z_1 = \sum_{r=1}^{R} \sum_{t=1}^{TR_r} Pax^{met}_{r,t} \qquad \forall r \in R, t \in TR_r \quad (5.19)$$

subject to: Constraints (5.1) – (5.17)

(z_2) *Minimise the maximum number of transfer passengers*

The objective z_2 ensures that more bus trips occur at peak periods by distributing the met trains more uniformly in the given schedule horizon. We introduce an integer variable $Pax^{train}_{r,t}$ that computes the maximum number of passengers on a met train; $t_{curr} = T^{met}_{r,t}$ is a non-negative integer variable indicating the current train to be met; $t_{prev} = T^{met}_{r,t-1}$ is a non-negative integer variable indicating the previous train met; ht and hpt are non-negative integer variables for the hour of the current and previous train departures, respectively. Equation (5.20) presents the computation of $Pax^{train}_{r,t}$ for three cases: (i) 0, if the bus arrival is null that is, beyond the schedule horizon; (ii) for the first bus arrival within hour ht, we take the combined transfer volume proportionate to the train in hour ht and any hour h prior to it; (iii) for all the remaining bus arrivals, the proportion of transfer passenger volume is computed based on the gap between two consecutive met trains (t_{curr} and t_{prev}) in hour ht and previous hour hpt, respectively.

$$Pax^{train}_{r,t} = \begin{cases} 0, & \text{if, } t > n_r \\ P_{ht,r} * (T^{dep}_{s,t_{curr}} - ht) + \sum_{h<ht} P_{h,r}, & \text{else if, } t = 1 \\ [P_{hpt,r} * ((hpt+1) - T^{dep}_{s,t_{prev}}) - & \\ P_{ht,r} * ((ht+1) - T^{dep}_{s,t_{curr}})] + \sum_{hpt<h<=ht} P_{h,r}, & \text{else.} \end{cases}$$
$$\forall r \in R; s \in S; h, ht, hpt \in H \qquad (5.20)$$

Objective z_2 then minimises the maximum number of passengers on a met train, which is given as follows:

$$\min \quad z_2 = \max(Pax^{train}_{r,t}) \qquad \forall r \in R, t \in TR_r \quad (5.21)$$

subject to: Constraints (5.1) – (5.17)

(z_3) *Minimising the headway deviation* (z_3)

The objective, z_3 minimises the deviation between optimised bus headways (given by $hw^{opt}_{r,t}$) and desired target headways (given by $hw^{target}_{r,t}$). This objective is motivated by the importance of maintaining bus service regularity throughout the schedule horizon. Note that the constraints (5.9) and (5.11) still enforce that successive bus trips must meet distinct trains.

We add a variable $hw^{diff}_{r,t}$ here, which computes the absolute difference between $hw^{opt}_{r,t}$ and $hw^{target}_{r,t}$ on a given route r and its corresponding trips (Equation (5.23)):

$$hw^{opt}_{r,t} = B^{start}_{r,t+1} - B^{start}_{r,t} \qquad \forall r \in R, t \in TR_r \quad (5.22)$$
$$hw^{diff}_{r,t} = \left| hw^{opt}_{r,t} - hw^{target}_{r,t} \right| \qquad \forall r \in R, t \in TR_r \quad (5.23)$$

Objective z_3 then minimises $hw^{diff}_{r,t}$ across all routes and trips, which formulated as follows:

$$\min \quad z_3 = \sum_{r=1}^{R} \sum_{t=1}^{TR_r} hw^{diff}_{r,t} \qquad \forall r \in R, t \in TR_r \quad (5.24)$$

subject to: Constraints (5.1) – (5.17)

(z_4) *Minimising wait per passenger*

The objective z_4 minimises the total excessive wait time (in mins), between a given bus arrival $B^{start}_{r,t} + Run_r$ and its corresponding train departure $T^{met}_{r,t}$, ignoring the transfer passenger volume. Since both the number of transfer passengers and excess wait time are variables dependent on bus times, solving an objective containing both is hard and requires significant computational efforts. With this decomposition, the complete problem can be broken down to compute the choice of trains to be met with minimised waiting times per passenger first, which can then work as an input to infer the minimised excessive waiting times for all passengers in the next stage.

Objective z_4 can thus be formulated as follows:

$$\min \quad z_4 = \sum_{r=1}^{R} \sum_{t=1}^{TR_r} T^{met}_{r,t} - (B^{start}_{r,t} + Run_r + walk_r) \qquad \forall r \in R, t \in TR_r \quad (5.25)$$

subject to: Constraints (5.1) – (5.17)

(z_5) Fixing the bus headways

Using this objective, z_5, we use fixed (or regular) bus headways to determine the choice of trains met by buses such that the sum of objective cost per bus route is minimised. An advantage of this strategy is that fixing the arrival of the first bus on a route also fixes all the consecutive bus arrival times on that route and therefore, the full objective cost for that route can be calculated easily. Consequently, we adapted our model such that this objective was computed a-priori for every possible first bus arrival time on each route. The remaining requirement is to find bus tours with a given number of buses that can achieve the arrival time of each trip on each route, which becomes computationally challenging when there are fewer buses. Essentially, this objective can be looked at as a subset of the full objective of BTCP.

Given a starting time $D \in [St_{min}, St_{max}]$ for each bus on route r (where $[St_{min}, St_{max}]$ is the range of earliest and latest start times a bus can take) and a running time, Run_r from the start of a route to its coordinating transfer station, Equation (5.26) computes the bus arrival times on route r (given by $B^{arr}_{r,t,D}$) , based on its fixed headway $hw^{target}_{r,t}$.

$$B^{arr}_{r,t,D} = \begin{cases} D + Run_r & \text{if, } t = 1 \\ D + Run_r + \sum hw^{target}_{r,t} & \text{else if } t \leq n_r \\ N_{time} & \text{else.} \end{cases} \qquad \forall r \in R; t \in TR_r \quad (5.26)$$

To complete the definition of objective cost, we also need to know the proportion of transfer passenger volume between two consecutive bus arrivals on a given hour- $B^{hour}_{r,t,D}$ and the hour prior to it- $B^{hour}_{r,t-1,D}$, respectively. We compute this using Equation (5.28) for three cases: (i) 0, if the bus arrival is null that is, beyond the schedule horizon; (ii) when $t = 1$ or the first bus arrival within hour $B^{hour}_{r,1,D}$, we take the combined transfer volume proportionate to this and any hour h prior to it; (iii) for all the remaining bus arrivals, the proportion of transfer passenger volume is computed based on the gap between two consecutive bus arrivals ($B^{arr}_{r,t,D}$ and $B^{arr}_{r,t-1,D}$) in hour $B^{hour}_{r,t,D}$ and previous hour $B^{hour}_{r,t,D}$, respectively.

$$B^{arr}_{r,1} = D \qquad \forall r \in R, D \in [St_{min}, St_{max}] \quad (5.27)$$

$$Pax^{bus}_{r,t,D} = \begin{cases} 0, & \text{if, } t > n_r \\ P * (D - B^{hour}_{r,1,D}) + \sum_{h < B^{hour}_{r,1,D}} P_{h,r} & \text{else if, } t = 1 \\ [P * (B^{hour}_{r,t-1,D} - B^{arr}_{r,t-1,D}) - P * (B^{hour}_{r,t,D} - B^{arr}_{r,t,D})] & \\ \qquad + \sum_{B^{hour}_{r,t-1,D} < h <= B^{hour}_{r,t,D}} P_{h,r} & \text{else.} \end{cases}$$
$$\forall r \in R; t \in TR_r; D \in [St_{min}, St_{max}] \quad (5.28)$$

We define an additional variable C^{route}_r here, which is the objective cost or the passenger waiting time (in passenger-mins) per bus route (Equation (5.29)).

$$C^{route}_r = \sum_{t=1}^{TR_r} Pax^{bus}_{r,t,D} * (T^{met}_{r,t} - (B^{arr}_{r,t,D} + walk_r))$$
$$\forall r \in R, t \in TR_r, D \in [St_{min}, St_{max}] \quad (5.29)$$

Objective z_5 thus minimises the sum of these objective costs across all routes:

$$\min \quad z_5 = \sum_r C^{route}_r \qquad \forall r \in R \quad (5.30)$$

subject to: Constraints (5.1) – (5.17)

STAGE-2: Full BTCP Objective (Z):

Stage-2 of the SD-TTVS approach is concerned with optimising the full objective (Z) of the BTCP, subject to the choice of met trains known and passed from solving each Stage-1 sub-objective. Note that the *only* values passed from Stage-1 to Stage-2 are the choice of trains to be met that is, $T^{met}_{r,t} : r \in R, t \in TR_r$ (results from decision type (iv)), keeping the interaction between the two optimisation stages as narrow as possible. In this stage, decisions of type (i) bus start time, (ii) bus flow sequence and (iii) bus hours and are re-computed such that the full objective of minimising excessive passenger waiting time is met. Furthermore, the timetable coordination decisions (type (v) to (vii)) are made that represent the complete problem of BTCP.

Since only the met train values T^{met}_r are passed on, Stage-2 also needs to find feasible values for the bus flow variables (Constraints (5.1) to (5.7)), the next bus and last bus values (Constraints (5.9) and (5.10) respectively), the bus headways (Constraint (5.12)), bus running times and arrival times (Constraint (5.13)) and transfers (Constraint (5.15)). Consequently, these constraints are kept common between Stage 1 and Stage 2 models. In addition to these, the following new constraints related to timetable coordination are also formulated:

$$Pax_{r,t} = \begin{cases} 0, & \text{if, } t > n_r \\ P * (B^{start}_{r,1} + Run_r - B^{hour}_{r,1}) + \sum_{h<B^{hour}_{r,1}} P_{h,r} & \text{else if, } t = 1 \\ [P * ((B^{hour}_{r,t-1} + 1) - B^{start}_{r,t-1} + Run_r)- & \\ P * ((B^{hour}_{r,t} + 1) - B^{start}_{r,t} + Run_r)] & \\ \qquad + \sum_{B^{hour}_{r,t-1}<h<=B^{hour}_{r,t}} P_{h,r} & \text{else.} \end{cases}$$

$$\forall r \in R; t \in TR_r; h \in H \qquad (5.31)$$

The passenger volume constraint (5.31) computes the number of transferring passengers between consecutive bus arrivals at a station. Note that the primary difference between $Pax^{bus}_{r,t,D}$ (Equation 5.28) and $Pax_{r,t}$ is that for the latter, the first trip departure time D is not fixed and all bus arrivals are variables and unknown. $Pax_{r,t}$ is thus a variable dependent on bus arrival times and corresponding bus arrival hours. It is defined for three cases: (i) 0, if the bus arrival is null that is, beyond the schedule horizon; (ii) for the first bus arrival within hour $B^{hour}_{r,1}$, we take the combined transfer volume proportionate to this hour and any hour h prior to it; (iii) for all the remaining bus arrivals, the proportion of transfer passenger volume is computed based on the gap between two consecutive bus arrivals in current hour $B^{hour}_{r,t}$ and previous hour $B^{hour}_{r,t-1}$, respectively.

$$wait_{r,t} = T^{met}_{r,t} - (B^{start}_{r,t} + Run_r + walk_r) \qquad \forall r \in R, t \in TR_r \quad (5.32)$$

$$passwait_{r,t} = Pax_{r,t} * wait_{r,t} \qquad \forall r \in R, t \in TR_r \quad (5.33)$$

Constraint (5.32) then computes the excessive waiting time between a given bus-train connection pair (where T^{met}_r indicates met trains known from Stage-1) and Constraint (5.33) produces the total passenger waiting time, which is minimised as the main objective. The trade-off between passenger transfer waiting time and operator costs is calculated by solving the objective for each input bus fleet size $b_{ct} \in B$. The full objective (Z) of BTCP is thus formulated as:

$$\min \quad Z = \sum_{r=1}^{R} \sum_{t=1}^{TR_r} passwait_{r,t} \qquad \forall r \in R, t \in TR_r \quad (5.34)$$

subject to:

Constraints (5.1) − (5.7);

Constraints (5.9), (5.10), (5.12), (5.13), (5.15), (5.16)

5.5 Modelling in MiniZinc

In this section, we describe the translation of the BTCP into a Constraint Programming (CP) based optimisation framework. Mirroring the mathematical formulations described in Section 5.4, we present a simplified version of the two-stage optimisation models, written entirely in *MiniZinc*. The *MiniZinc* models for both stages of optimisation and the most relevant data inputs are presented in Appendices A.1 and A.3, respectively.

Input Data:

- a set of bus uni-directional routes, `ROUTES`; `route` ∈ `ROUTES`
- a set of bus trips, `TRIPS`; `trips` ∈ `TRIPS`
- a set of transfer train stations, `STN`; `rsct` ∈ `STN`
- a set of physical buses, `BUSES`; `bct` ∈ `BUSES`
- a set of physical trains, `TRAINS`; `pt` ∈ `TRAINS`
- a set of all time values in the planning horizon, `MAXTIME`
- a minimum walk time, `walk` between a bus arrival and connected train departure
- a bus running times, `RT` from the start of a route to the start of the next route; `RTCoord` is a subset of `RT` comprising the running time from the start of a route to its coordinating transfer station to account for those cases where a bus must coordinate with a transfer station mid-route
- the end time of the schedule horizon, `tmax`
- the train departure time from each transfer station, `ttime`
- a minimum and maximum bus headway per route, `minHW` and `maxHW`, respectively
- the hourly transfer passenger volume per station, `pass`

```
set of int: ROUTES = 1..route;  % set of all routes in the network
set of int: TRIPS = 1..max(trips);  % set of all bus trips
set of int: BUSES = 1..bct ;  % set of all buses
set of int: TRAINS = 1..pt;  % set of all physical trains
set of int: MAXTIME = 0..Ntime+max(walk); % set of all time
set of int: HOURS = 1..hours; % set of all hours
```

Listing 5.1: Set declarations

Listing 5.1 shows the set declarations. We measure all time values in minutes, although time granularity can be easily changed in the model. All active bus routes are represented as `route` and their corresponding active trips as `trips`. Coordinating transfer stations occur at the end of each route. We also consider those exceptions where a bus route must coordinate with a station mid-route. In addition, the *null* time parameter `Ntime` which indicates the dummy bus arrival or train departure time.

Decision Variables:

- `BStart` denotes the start trip time of each bus route
- `flow` decides the physical buses assigned to each route and trip
- `BHour`, is the hour within which each bus on its route and trip arrives
- `trainmeets` decides the physical train assigned to each matching bus route and its trip

- `passBT` is the bus-train ransfer passenger volume between two consecutive bus arrivals
- `waitBT` returns the waiting time between a bus arrival and matching train departure, measured in minutes
- `passwaitBT` returns the waiting time computed for each transferring passenger in a bus-train connection pair, measured in passenger-minutes

5.5.1 Stage-1 modelling

Listing 5.2 shows the declaration of variable arrays and their corresponding domains in *MiniZinc* for Stage-1 of the SD-TTVS.

```
% bus start times
array [ROUTES,TRIPS] of var MAXTIME: BStart;
% bus flow sequence
array [ROUTES,0..max(trips),ROUTES,1..final_trip] of var 0..1: flow;
% choice of trains met
array [ROUTES,TRIPS] of var TRAINS: trainmeets;
```

Listing 5.2: Decision variable declarations for Stage-1 of the SD-TTVS

Bus flow and run-time constraints: We restrict all active bus flows to be performed by an input number of buses, `bct`. We require two consecutive start trips of a bus on routes `r1` and `r2` to be separated by a minimum running time `RT[r1,r2]` and layover time between these trips. Before route `r2` commences, we impose a minimum bus layover (`minlay`) at 10% route running time and a maximum layover (`maxlay`) at an additional 15 minutes. Since our model allows the bus trips to freely interline, we do not precisely identify what trip must succeed or precede the other (Listing 5.3).

```
% bus fleet size constraint
sum([flow[1,0,r2,t2]| r2 in ROUTES, t2 in 1..trips[r2]]) <= bct;

forall(r1,r2 in ROUTES, t1 in 1..trips[r1], t2 in 1..trips[r2])(
(flow[r1,t1,r2,t2]=1 -> % for an active bus flow
    BStart[r2,t2] >= BStart[r1,t1] + RT[r1,r2] + minlay[r2]) /\
(flow[r1,t1,r2,t2]=1 ->  % for an active bus flow
    BStart[r2,t2] <= BStart[r1,t1] + RT[r1,r2] + maxlay[r2]));
```

Listing 5.3: Bus flow and run-time constraints

First and last bus constraints: As a span of hours requirement, all bus arrivals at a transfer station must fall within the schedule horizon of [0, `tmax`]. We thus constraint the first bus trip arrival of a route within its maximum headway `maxHW`. Similarly, the second bus constraint restricts the second bus arrival after the first train on the coordinating station has departed; this ensures a successful first train connection (Listing 5.4).

```
forall(r in ROUTES)
(BStart[r,1]+RTCoord[r] <= maxHW[r]);  % first bus constraint

forall(r in ROUTES,t in 2..trips[r])
(BStart[r,t] > ttime[s,t-1]-RTCoord[r]) ;  % second bus constraint
```

Listing 5.4: First bus constraints

Similarly, the last bus trip on any route must arrive at the station a time between [`tmax`, `maxHW`]. The second last bus constraint ensures that last but one bus arrives before last but one train. This is to enable a successful connection for the buses and trains at the end of `tmax` (Listing 5.5).

```
% last bus constraint
forall(r in ROUTES)
(BStart[r,trips[r]]+RTCoord[r] >= tmax-maxHW) ;
% second last bus constraint
forall(r in ROUTES,t in 0..(trips[r]-1))
(BStart[r,trips[r]-t] <=
 ttime[r_sct[r],last_train[r]-t] - (RTCoord[r]+walk[r]));
```

Listing 5.5: Last bus constraints

Bus headway constraints: The headway between consecutive bus trips on any route is constrained to fall between a minimum and maximum, given by `minHW` and `maxHW`, respectively (Listing 5.6).

```
forall(r in ROUTES, t in 1..trips[r]-1) % bus headway constraint
(BStart[r,t+1] >= BStart[r,t] + minHW[r] /\
 BStart[r,t+1] <= BStart[r,t] + maxHW[r]);
```

Listing 5.6: Bus headway constraints

Transfer constraint: Every bus trip on a route must arrive in time to meet its connecting train, leaving enough transfer `walk` time (Listing 5.7).

```
array [ROUTES,TRIPS] of var 0..max(max_hdwy): waitBT  =
    array2d(ROUTES,TRIPS,
    [ttime[r_sct[r],trainmeets[r,t]]-(BStart[r,t]+RTCoord[r]+walk[r])
    | r in ROUTES, t in TRIPS]);
```

Listing 5.7: Transfer constraints

Sub-Objectives(s):

Maximising the number of passengers on met trains: We translate the Objective 5.19 here to maximise the number of passengers on a met train; the integer array `onetrainpass` considers the passenger volume from buses dependent on the time gap between two consecutive train departures. Its sum across routes and trips is then maximised.

```
array[ROUTES,1..pt] of HOURS: trainhour  = % train departure hours
    array2d(ROUTES,1..pt,
    [max([h * bool2int(ttime[r_sct[r],k] > hours[h])
    | h in HOURS])|r in ROUTES, k in 1..pt]);

array[ROUTES, TRAINS] of int: onetrainpass = % pax in desired trains
array2d(ROUTES,TRAINS,  [let {int:ht = trainhour[r,1]} in
    if j=1 then % first train
        Pass[ht,r]*(ttime[r_sct[r],1]-hours[ht])
      + sum(h in HOURS where h < ht )(60*Pass[h,r])
    else let {int: hpt = trainhour[r,j-1]} in % remaining trains
              (Pass[hpt,r]*(hours[hpt]+60-ttime[r_sct[r],j-1])
             - Pass[ht,r]*(hours[ht]+60-ttime[r_sct[r],j]) )
             + sum(h in HOURS where h>hpt /\ h  <= ht )(60*Pass[h,r])
    endif | r in ROUTES, j in TRAINS ]);
% objective function
var int: objz1 = sum([onetrainpass[r,trainmeets[r,t]]
               | r in ROUTES, t in 1..trips[r]]);   % objective z1
solve maximise objz1;
```

Minimise the maximum number of transfer passengers: We translate the Objective 5.21 here to minimise the maximum number of transfer passengers on a met train, such that the met trains are spread more uniformly across the schedule horizon. The variable array `trainpass` computes the maximum number of passengers on a met train by considering the time gap from the previous met train. Its sum across routes and trips is then minimised.

```
array[ROUTES, TRIPS] of var int: trainpass = % pax in met trains
array2d(ROUTES,TRIPS,
[if      t>trips[r] then 0  % null bus arrival
elseif   t=1 then           % first bus arrival
    let {var TRAINS:train = trainmeets[r,1],
         var HOURS:ht = trainhour[r,train]} in
        (Pass[ht,r]*(ttime[r_sct[r],train]-hours[ht])
         + sum(h in HOURS where h  < ht )(60*Pass[h,r])   )
else let {var TRAINS:prevtrain = trainmeets[r,t-1], % remaining buses
          var TRAINS:train = trainmeets[r,t],
          var HOURS:hpt=trainhour[r,prevtrain],
          var HOURS:ht=trainhour[r,train]} in
        ((Pass[hpt,r]*(hours[hpt+1]-ttime[r_sct[r],prevtrain])
        - Pass[ht,r]*(hours[ht+1]-ttime[r_sct[r],train]) )
        + sum(h in HOURS where h>hpt /\ h  <= ht )(60*Pass[h,r]))
endif  | r in ROUTES, t in TRIPS]) ;
% objective function
var int: objz2 = max([trainpass[r,t]
|r in ROUTES, t in 1..trips[r]]);   % objective z2
solve minimize objz2;
```

Minimising the headway deviation: We translate the Objective 5.24 here to minimise the headway deviation; the function `headwaydiff` is defined as the absolute difference between the optimised and target bus headways on a given route, r.

```
function var int: headwaydiff(ROUTES:r, TRIPS:t) =
% optimised headway - target headway
abs((BStart[r,t+1]-BStart[r,t])-targetHW[r,t]) ;
% objective z3
var int: objz3 = sum(r in ROUTES,t in 1..trips[r]-1)(headwaydiff(r,t));
solve minimize objz3;
```

Minimising wait per passenger: We translate the Objective 5.25 here to minimise the total excessive wait time `waitBT` (as defined in Listing 5.7), discarding the transfer passenger volume;

```
var 0..sum([max_hdwy[r]*trips[r]|r in ROUTES]): objz4 = % objective z4
sum(r in ROUTES,t in 1..trips[r])(waitBT[r,t]); % wait pr. passenger
solve minimize objz4;
```

Fixing the bus headways: We translate the Objective 5.30 here to minimise the sum of objective costs per bus route. Given a bus starting time `s` *in* `[minstart, maxstart]` and fixed bus headways, array `allcosts` computes the total passenger waiting time (given by `pass(r,t,s)*waitBT(r,t,s)` for every possible bus arrival time on each route . Sum of these costs across all routes is then minimised using the function `rcost`.

```
array [ROUTES, minstart..maxstart] of int: allcosts =
      array2d(ROUTES,minstart..maxstart,
      [sum(t in 1..trips[r])(pass(r,t,s)*waitBT(r,t,s)) % cost per route
       | r in ROUTES,s in minstart..maxstart ]);
function var 1..max(maxcost): rcost(ROUTES:r) =
         allcosts[r,BStart[r]];
var 0..sum(maxcost):objz5 = sum([rcost(r)|r in ROUTES]); % objective z5
solve minimize objz5;
```

5.5.2 Stage-2 modelling

To maintain compatibility between the two optimisation stages, Constraints listed from (5.3) to (5.7) are common between Stages 1 and 2. The following constraints are expressed newly in Stage-2 to solve the full objective:

Transfer passenger volume constraint: We compute the transfer passenger volume `passBT` between two consecutive bus arrivals (Listing 5.8). The proportion of transfer volume between bus arrivals at bus hours `ht0`, `ht1` $\in$ `BHour`, respectively, are recorded here.

```
% hour of bus start
array [ROUTES,TRIPS] of var HOURS: bushour ;
% transfer pax volume from bus to train
array [ROUTES,TRIPS] of var 0..max(sumPass) : passBT;
constraint forall(r in ROUTES,t in TRIPS)(passBT[r,t]>=0) ;
constraint forall(r in ROUTES, t in 1..trips[r])
(passBT[r,t] =  let {var HOURS:ht1 = bushour[r,t] } in
  if       t=1 then              % first trip
           (Pass_data[ht1,r]* ((BStart[r,1]+RTCoord[r])-hours[ht1])
          + sum(hs in HOURS)(bool2int(hs<ht1)*Pass_data[hs,r]*60))
  elseif   t>trips[r] then 0    % null trips
  else     let {var HOURS:ht0 = bushour[r,t-1] } in
           ((Pass_data[ht0,r]*(hours[ht0+1]- (BStart[r,t-1]+RTCoord[r]))
           - Pass_data[ht1,r]*(hours[ht1+1]- (BStart[r,t]+RTCoord[r])))
           + sum(hs in HOURS)(bool2int(hs>ht0/\hs<=ht1)*60*Pass_data[hs,r
   ]))
  endif) ;
```

Listing 5.8: Transfer passenger volume constraint

Full Objective: The full BTCP objective is the product of `waitBT` and `passBT` across all routes and trips. Listing 5.9 shows this objective expression which is minimised against input number of buses, `bct`.

```
var 0..sum([sumPass[r]*max_hdwy[r]|r in ROUTES]): passwaitBT;
constraint passwaitBT = sum(r in ROUTES,t in 1..trips[r])
    (waitBT[r,t]*passBT[r,t])  ;
% full BTCP objective Z
solve minimize passwaitBT;
```

Listing 5.9: Full objective expression

5.6 Computational Experiments and Case Study

In this section, we detail the generation of solutions using the SD-TTVS approach through a set of computational experiments. All experiments are run on a personal computer with Intel Core i7-7600 @ 2.9GHz CPU and 16GB RAM. We test the proposed SD-TTVS approach on our case study area in the City of Wyndham, Melbourne for problem instances with varying PT network sizes namely, *Small* and *Large*. All instances in this study are solved using *Gurobi* at a time-limit of 30 minutes per test run. It is important to note that due to the efficient solver-interfacing capability of *MiniZinc*, it linearises the CP model devised in Section 5.4 such that it is compatible to be solved using *Gurobi*. In some cases, we also use *Chuffed* to prove solution infeasibility.

Table 5.1 shows the numerical results from using the different decomposition strategies in our case study area. The first column indicates the network instance type and size; the second column shows the input number of buses required to produce feasible solutions from each instance. For every input number of buses, the Stage-1 and Stage-2 objective values obtained for each decomposition strategy are reported, along with the corresponding CPU time in seconds. Note that we consider a set of solutions to be the "best" when there is no solution improvement within the given time-limit of 30 mins or when the *optimality gap*[3] returned by objectives in both stages is 0.00%.

Instance: *Small* (6 *X* 4)

Solving the *Small* PT network instance comprising 6 bus routes and 4 transfer stations using the *Chuffed* solver, bus fleet size < 6 is infeasible for all decomposition alternatives, Using *Gurobi*, the search for solutions timed-out (crossed the 30 min CPU time limit) with a bus fleet size of 6 but reported feasible Stage-1 objective values with 7 buses. Incrementally increasing the number of buses up to 10, feasible trains met by each trip were recorded for each of the proposed decomposition strategies and thereby fed into Stage-2. Note that in two cases, it was not possible to prove optimality for Stage-1 solutions. However, all Stage-2 solutions were proven optimal in a few seconds.

Comparing the performance of different decomposition strategies, the "best" solution for the *Small* instance is obtained using $Z \leftarrow z_4$ that is, by decomposing the complete problem to minimise the waiting time per passenger (in mins) first. The best objective value thus obtained in Stage-2 is 2,764 passenger-mins with 9 buses, at a CPU time of 14s. The second best decomposition strategy is observed to be $Z \leftarrow z_5$ that is, decomposing the complete problem such that the objective cost per route is minimised first with the simplifying constraint (5.26) that fixes bus headways. This decomposition reported a Stage-2 objective value of 5,542 passenger-mins with 9 buses, at a CPU time of 11s. Considering z_3, clearly, if there exists a feasible solution with fixed headways then the minimum headway deviation is zero; so if $Z \leftarrow z_5$ is solvable then the objective $Z \leftarrow z_3$ is obsolete. It can also be observed from Table 5.1 that sub-objectives z_1 and z_2 did not fare well in producing good quality Stage-2 solutions.

Instance: *Large* (24 *X* 5)

The pure integer-linear Stage-1 and Stage-2 FlatZinc models generated by *MiniZinc* for the *Large* instance is 11.18 and 11.20 times larger than that for the *Small* instance, respectively. The growth in each model instance comprises the increase in input data, the increased number of disjunctions, implications and negations that are translated by *MiniZinc*.

We know from Chapter 4, Section 4.4 that the *Large* network instance, we have no access to a true global optimum objective value. In such a case, the proposed decomposition could be a beneficial solving approach. Observing the best decomposition strategies from the *Small* instance to be $Z \leftarrow z_4$ (that is, *minimising wait per passenger*) and $Z \leftarrow z_5$ (that is, *fixing the bus headways*),the *Large* instance too is tested against these objectives.

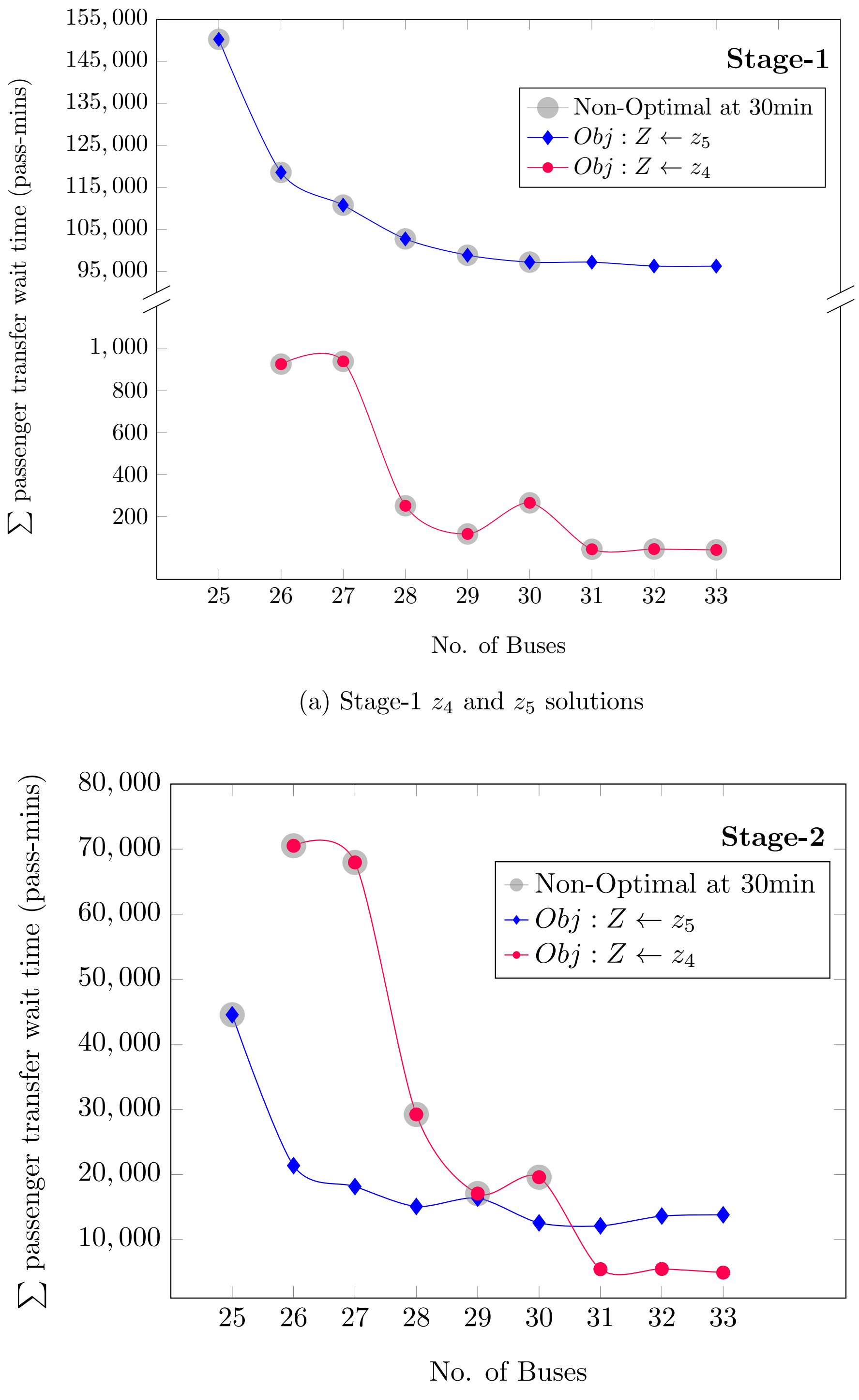
[3]In Mixed Integer Programming (MIP), the optimality gap is the magnitude of the difference between the best incumbent solution and the best known lower bound of a problem

		STAGE-1		STAGE-2			
Network Instances	No. of buses	$Obj(s) : z$	CPU time (s)	$Obj : Z$	CPU time (s)	O^{bb}_{BTCP}	Gap (%)
Small (6 *X* 4)	6	*TO*	-	*TO*	-	66,796$^{(*)}$	-
	7	z_1= 137,543	528s	33,571	11s	8,856$^{(*)}$	73.62%
	7	*TO*	-	*TO*	-	8,856$^{(*)}$	-
	7	z_3= 0	17s	13,772	12s	8,856$^{(*)}$	35.70%
	7	z_4= 49$^{(*)}$	1800s	9,262	14s	8,856$^{(*)}$	4.38%
	7	z_5= 43,909	901s	15,175	14s	8,856$^{(*)}$	41.64%
	8	z_1= 136,573	96s	32,367	11s	3,140	90.30%
	8	z_2= 300	47s	23,989	11s	3,140	86.91%
	8	z_3= 0	16s	11,192	11s	3,140	71.94%
	8	z_4= 26$^{(*)}$	1800s	3,149	14s	3,140	0.29%
	8	z_5= 31,210	901s	6,442	12s	3,140	51.26%
	9	z_1= 136,620	16s	35,095	11s	2,764	92.12%
	9	z_2= 300	42s	27,444	11s	2,764	89.93%
	9	z_3= 0	17s	8,456	11s	2,764	67.31%
	9	z_4= 18	88s	2,764	14s	2,764	0.000%
	9	z_5= 29,624	11s	5,542	11s	2,764	50.13%
Large (24 *X* 5)	25	z_4 : *TO*	-	*TO*	-	-	-
	25	z_5= 150230$^{(*)}$	1800s	44544$^{(*)}$	1800s	-	-
	26	z_4= 924$^{(*)}$	1800s	70,515$^{(*)}$	1800s	-	-
	26	z_5= 118578$^{(*)}$	1800s	21354	1415s	-	-
	27	z_4= 937$^{(*)}$	1800s	67,957$^{(*)}$	1800s	-	-
	27	z_5= 110781$^{(*)}$	1800s	18144	274s	-	-
	28	z_4= 250$^{(*)}$	1800s	29,227$^{(*)}$	1800s	-	-
	28	z_5= 102755$^{(*)}$	1800s	15084	241s	-	-
	29	z_4= 116$^{(*)}$	1800s	17,069$^{(*)}$	1800s	-	-
	29	z_5= 98873$^{(*)}$	1800s	16352	238s	-	-
	30	z_4= 264$^{(*)}$	1800s	19,582$^{(*)}$	1800s	-	-
	30	z_5= 97225	233s	12562	204s	-	-
	31	z_4= 43$^{(*)}$	1800s	5,446	177s	-	-
	31	z_5= 97214	194s	12101	191s	-	-
	32	z_4= 44	1800s	5,483	177s	-	-
	32	z_5= 96297	166s	13610	181s	-	-
	33	z_4= 40$^{(*)}$	1800s	4,922	1721s	-	-
	33	z_5= 96278	188s	13793	203s	-	-

$^{(*)}$: Non-optimal solutions obtained at 30 minutes CPU time
TO : Search has timed-out (or, no feasible solutions found within 30 minutes)

Table 5.1: Objective value comparison for all network instances with the SD-TTVS approach

Solving the objective $Z \leftarrow z_5$ that is, using fixed headway to compute cost of each route and then solving the complete problem is observed to be surprisingly successful with as few as 25 buses. This decomposition approach enables a fast convergence to high quality solutions for *Large* instance with varying bus fleet sizes. For a sufficiently large bus fleet of 33 buses, the best full objective value yielded with this objective is 13,793 passenger-mins in 203s (Table 5.1). $Z \leftarrow z_4$ fares well as the second best decomposition strategy with solutions obtained with buses as few as 26. Figure 5.2 compares the trade-off between the input number of required buses and the corresponding total passenger transfer waiting time (pass-mins) obtained using these two decomposition strategies on the *Large* network instance.



(a) Stage-1 z_4 and z_5 solutions

(b) Stage-2 z_4 and z_5 solutions

Figure 5.2: Best objective value comparison from (a) Stage-1 and (b) Stage-2 models for the *Large* network instance

5.6.1 SD-TTVS Performance Measurement

Although we get optimal solutions in Stage-1 and corresponding optimal solutions in Stage-2 based on the inputs from Stage-1, it is not guaranteed that the final solution is globally optimal. As seen in Table 5.1, solving individual objectives at each stage provides us locally optimal solutions concerned with that stage, however, it cannot be concluded whether these solutions are in fact, part of a globally optimal solution of the complete problem. The advantage of using integer-linear solvers like *Gurobi* is that they can produce not only problem solutions, but also bounds on the optimum. Thus, even if a solution cannot be proven optimal, the solver can return the *optimality gap* which is a percentage figure saying how far it could be from optimality. In case the gap is 1%, this means the solution may be optimal, and it is certainly within 1% of the optimal. Frustratingly, if the best bound is unknown, this tells us nothing about the quality of the solution.

To understand how far the generated SD-TTVS solutions are from true global optimum, we compare these with the solutions obtained from solving the original complete problem. We express the computational performances in terms of the *optimality gap*, as shown in Equation (5.35), where $O^{inc}_{SD-TTVS}$ indicates the best incumbent solution from SD-TTVS and O^{bb}_{BTCP} indicates the true global optimum (or the best lower bound where the true global optimum could not be found) for the complete problem of BTCP, solved for each bus fleet size.

$$\text{Optimality Gap (\%)} = \frac{O^{inc}_{SD-TTVS} - O^{bb}_{BTCP}}{O^{inc}_{SD-TTVS}} \tag{5.35}$$

Recall from Chapter 4 (Table 4.1) that the complete problem of bus-train timetable coordination is solvable to optimality for the *Small* network instance (as shown under the column O^{bb}_{BTCP} in Table 5.1). With 9 buses, the objective value 2,764 pass-mins has proven to be the true global optimal. With 8 buses, the true global optimum obtained is 3,140 pass-mins. With 7 buses, the best lower bound observed at the end of 4 hours is 8,856 pass-mins. With 6 buses, *Gurobi* failed to find feasible solutions for the complete problem. However, with *Chuffed*, the solution returned at the end of 4 hours is recorded as 66,796 pass-mins. We cannot infer the optimality gap for the 6 bus instance since there are no feasible solutions available from the decomposition strategies.

The final column in Table 5.1 shows the optimality gap (%) between the locally optimal Stage-2 solutions and the corresponding global optimal solutions for the *Small* instance. Evidently, the SD-TTVS solutions from objective z_4 are the closest to the true optimum values, falling within an optimality gap of 4.38% to 0.00% with increasing fleet size of 7 to 9, respectively. However, in the case of *Large* network instance, as explained in the previous section, we are unable to infer the optimality gap due to the lack of feasible solutions from the full objective for comparison.

5.7 Conclusions

In this study, we explore the various ways the BTCP can be decomposed and re-ordered compared to the traditional sequence, to render it solvable on large network instances. We design a two-stage optimisation approach called the SD-TTVS and formulate the problem with multiple decisions and objectives concerned with determining cost efficient timetabling (TT) and vehicle scheduling (VS).

Using the proposed SD-TTVS approach, we divide the complete problem of BTCP into two inter-related optimisation stages with different sub-objectives, such that the characteristics of the comprehensive mathematical model are retained in both stages. Different from the existing studies on sequential solving of planning problems, we model the SD-TTVS in such a way that there is no iteration between the two sequential stages timetabling and vehicle scheduling. The model accommodates a variety of real-world constraints that makes the BTCP a complex optimisation problem to solve on larger networks at reasonable computation efforts.

Numerical results from testing this approach on a real-world PT network in the City of Wyndham, Melbourne indicate that efficient decomposition strategies to divide the complete problem into solvable and compatible sub-problems can yield good quality scheduling solutions at low computational effort. Notably, feasible solutions for the BTCP are obtained with decomposition, where direct solving with commercial solvers failed to find any. As an extension to this study, alternate approximations can be investigated with other problem variables, such as the bus flow sequence.

One of the key limitations with this study however, is the difficulty in assessing the quality of solutions produced especially for larger network instances; each sub-objective of the decomposition strategy produces an optimum solution against itself and there is no guarantee that the overall objective derived from these are globally optimum. In this case, it would be interesting to investigate the role of heuristics in yielding better results. We explore this methodology in the next Chapter 6 by introducing an *Integrated Timetabling & Vehicle Scheduling* (I-TTVS) approach, which is a constraint based meta-heuristic optimisation technique to solve the complete problem of BTCP.

Chapter 6

Integrated Approach for Timetable Coordination

6.1 Introduction

In the previous Chapter 5, we investigated the re-ordering of public transport planning sub-problems namely, timetabling (TT) and vehicle scheduling (VS) to solve the Bus-Train Timetable Coordination Problem ($BTCP$). Through a novel two-stage optimisation framework, we introduced multiple decomposition strategies to generate cohesive and scalable scheduling solutions such that the objectives of minimising passenger transfer waiting time and operator costs are met simultaneously. However, one of the main limitations of this study was the difficulty with assessing the quality of decomposed solutions in terms of closeness to the true optimum solutions of the BTCP. This led to the necessity to investigate the role of heuristics in yielding better solutions for the BTCP.

In this chapter, we present an *Integrated Timetabling and Vehicle Scheduling* approach (abbreviated as I-TTVS) to find better quality solutions to the BTCP. This approach models the timetabling and vehicle scheduling sub-problems of planning simultaneously such that a favourable trade-off between the contrasting objectives of improving timetable coordination and reducing operator costs is obtained. To tackle large network instances of the problem, we define an optimisation framework based on the *Large Neighbourhood Search* (LNS) meta-heuristic. Tested on a subset of a real-world public transit (PT) network in the City of Wyndham, Melbourne, we then present results that demonstrate the capability of this approach in yielding good-quality scheduling solutions within acceptable computation time.

Contributions

This chapter presents contribution C_3, that is, a *Large Neighbourhood Search* meta-heuristic approach for the BTCP through integrated timetabling and vehicle scheduling. This contribution is composed of the following components:

i. We first present the mathematical model for the I-TTVS comprising a wide range of real-world scheduling requirements that are gathered from the industry. This model integrates the timetabling and vehicle scheduling sub-problems with the objectives to minimise total transfer passenger wait time using fixed bus fleet size, while also maintaining bus service regularity.
ii. We then propose a *Large Neighbourhood Search* meta-heuristic, that generates improved scheduling solutions for large-scale, real-world PT network instances faster in comparison to general purpose solvers and problem decomposition.
iii. We illustrate the applicability and performance of our approach through a set of computational experiments on a real-world case study for the bus and train services in the City of Wyndham, Melbourne.

This chapter, which is the third chapter in Part II: *Optimisation Framework*, addresses research objectives RO_2 and RO_3 as stated below, which are tailored to suit the overall contributions from this study:

RO_2 : To formulate the BTCP incorporating real-world constraints.
RO_3 : To solve the BTCP using state-of-the-art optimisation techniques.

This chapter is structured as follows: in Section 6.2, we provide a brief background on the latest developments in integrated timetabling and vehicle scheduling with an emphasis on transfer optimisation. We then describe the I-TTVS approach in Section 6.3 and formulate it mathematically in Section 6.4, considering the most relevant scheduling constraints. Section 6.5 proposes the solutions approach based on the *Large Neighbourhood Search (LNS)* meta-heuristic to solve the I-TTVS. Section 6.6 demonstrates the performance of the proposed approach through a set of computational experiments. A brief discussion on the observed results is then presented. The chapter concludes with a discussion of the major findings and future research in Section 6.7.

6.2 Background

Current research indicates that integrating two or more public transport planning sub-problems suffer in terms of tractability, especially on large scale networks (Ceder, 2007). As previously noted in Chapter 2, conventional sequential approaches for planning, although effective in producing locally optimal solutions for each sub-problem, do not give us cohesive solutions for the complete planning problem. This has motivated more recent attention to integrate two or more of the planning stages and solving these simultaneously (Ibarra-Rojas et al., 2015).

This study focuses on the integration of timetabling (TT) and vehicle scheduling (VS) sub-problems, with transfer optimisation as the main objective. Optimising transfers is observed to be a popular objective in existing studies on integrated TT and VS (Ceder, 2001; Chakroborty et al., 2001; Liu and Ceder, 2017; Liu and Shen, 2007; Weiszer et al., 2010), the only exceptions observed being the research conducted by Carosi et al. (2019*b*); Schmid and Ehmke (2015), where deadheads and headway deviations are minimised and

Van den Heuvel et al. (2008), where the costs associated with assigning bus service trips to multiple vehicle types are minimised under the condition that the corresponding timetable remains periodic.

When developing an integrated model for TT and VS, it is important to manage the inter-dependencies between the contrasting objectives of maximising timetable quality and minimising operating costs. Due to the complexity of the complete problem, most studies rely on the capabilities of meta-heuristic or matheuristic[1] algorithms (Carosi et al., 2019*a*; Fonseca et al., 2018; Guihaire and Hao, 2008*b*, 2010; Liu and Ceder, 2017) to obtain good quality solutions in reasonable computational time. With regards to the use of *Large Neighbourhood Search* (LNS) meta-heuristics in PT planning, we are aware of the works of Schmid and Ehmke (2015) who model the integrated TT and VS as a vehicle routing problem with time windows and balanced departure times. A hybrid meta-heuristic approach is proposed which decomposed the problem into scheduling and balancing phases. The scheduling component is solved using LNS considering the minimisation of deadheads. In addition, Petersen et al. (2013) focuses on the simultaneous vehicle scheduling and passenger service problem (SVSPSP) on a multi-depot setting. They used LNS to initially obtain vehicle schedules without the timetable components. The solutions obtained in 12 hours are then input to solve the passenger service problem. In an aim to improve passenger service, they considered the selection of alternative trips with departure times that could reduce waiting times. In contrast to our study, they relax the requirement to cover every vehicle trip. There is also limited information on the scalability of the algorithm to broader scheduling horizons and larger networks.

Collectively, these studies outline the critical need to tackle the inherent complexity of PT timetable coordination and the limitations with existing practices in solving the problem holistically. One of the fundamental difference between current research and our study is the observation that the devised models need an existing input timetable (for example, Fonseca et al. (2018); Guihaire and Hao (2008*b*); Petersen et al. (2013)) where sub-trips are formed from an already given original timetable and work around shifting or modifying it based on some criteria that favours coordination and cost minimisation. Also, the inclusion of detailed practical considerations still remain a challenge, especially on large scale networks.

In summary, our contribution in comparison to the current research on integrated TT and VS is that instead of relying on already known timetables as inputs, we generate the solutions by satisfying a wider range of real-world scheduling constraints. The motivation to propose a meta-heuristic algorithm for the comprehensive problem of timetable coordination thus arises here, where the meta-heuristic takes care of improving the generated solutions further. In addition, we compare the efficiency of the proposed algorithm in producing good-quality solutions at acceptable computational time.

[1] combination of meta-heuristics and mathematical programming techniques (Boschetti et al., 2009)

6.3 The I-TTVS Approach

The I-TTVS (*Integrated Timetabling and Vehicle Scheduling*) approach is based on the concept of simultaneously solving the *tactical* planning problem- *Timetabling* (TT) (finding bus schedules that are well-coordinated with the train schedules and at minimum deviation from the target service headway) and the subsequent *operational* planning step- *Vehicle Scheduling* (VS) (finding cost-efficient bus schedule to serve the entire network). Figure 6.1 shows the I-TTVS approach in relation to the traditional planning sequence. This approach is motivated by the limitations observed with finding good quality and holistic solutions for the complete problem of BTCP, using the traditional sequential planning approaches.

The primary objective of the I-TTVS is to minimise the excessive transfer waiting time for passengers transferring between two given services such that it incurs minimum operational costs, simultaneously. The main inputs for the I-TTVS is a (i) pre-defined public transport network (PTN) with bus routes and associated desired frequencies defined across a given schedule horizon; (ii) the *bus trips per route* including the specified *desired headway* in the AM-peak and Inter-peak periods; (iii) the *fixed timetables* for coordinating train lines; (iv) a *Run-Time* matrix comprising the bus running time and deadhead time required to travel between coordinating stations while operating an in-service trip on a particular route (v) the *transfer passenger volume* between bus routes and corresponding transfer stations (vi) *walking time* between a given bus route and transfer station pair; (vii) *bus layover time* at the end of each route.

By definition, integrating the TT and VS sub-problems has the potential to explore a larger feasible solutions space, which may ideally increase the scope of finding better quality solutions. Such integration is commonly attempted partially or completely (Ibarra-Rojas et al., 2015); the former approach considers the characteristics of one sub-problem mainly while taking decisions of other sub-problems, and/or iteratively, where the degrees of freedom of the sub-problems is explored in iterations. The latter approach deals with model formulations and/or solution approaches that determine decisions for the complete problem. Evidently, the complete integration of planning sub-problems is extremely challenging, due to the inter-dependency between the two sub-problems. Searching a large solution space requires mathematical formulations that are cohesive and hence, complex and necessitates solution techniques that have the ability to diversify the search on a large space at reasonable computation times.

6.4 Mathematical Formulation for the I-TTVS

In this section, we revisit the most relevant constraints for the I-TTVS and present a simplified version of the comprehensive mathematical model for the BTCP (the complete model for the BTCP is given in Section 4.3, Chapter 4).

The integrated problem of timetabling and vehicle scheduling consists of the following decision variables that are solved simultaneously: (a) bus flow variables $flow_{r1,t1,r2,t2}$

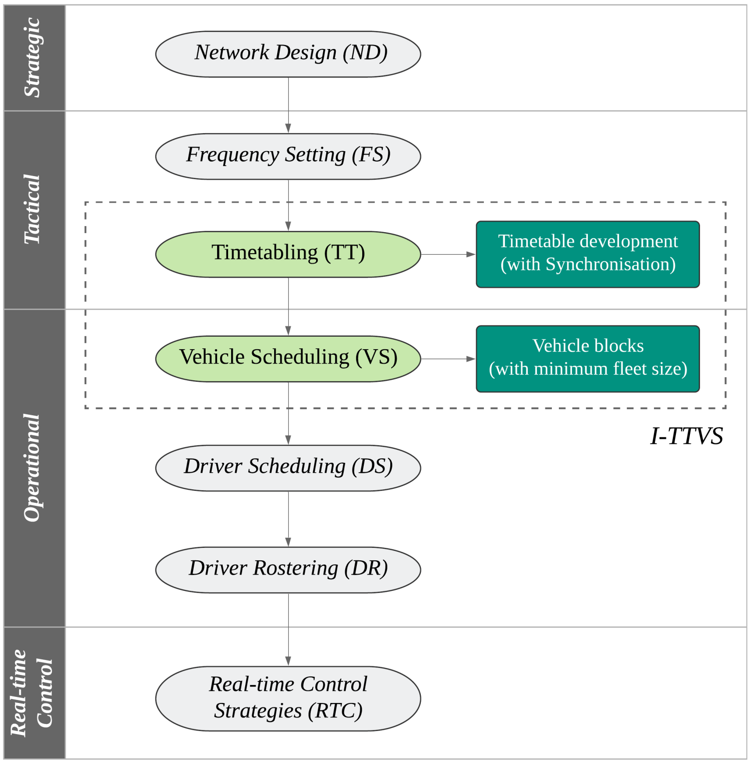


Figure 6.1: I-TTVS shown in relation to the traditional sequential planning approach

representing the sequence of bus trips performed by each bus; (b) bus start time variable $B^{start}_{r,t}$ indicating the start time of bus trip t on route r; (c) bus arrival hour $B^{hour}_{r,t}$ variable indicating the hour within which each bus on route r and trip t arrives; (d) train meets variable $T^{met}_{r,t}$ indicating the physical train that is met by a bus trip t on route r; (e) passenger volume variable $Pax_{r,t}$ representing the number of passengers transferring from bus trip t on route r; (f) waiting time variable $wait_{r,t}$ indicating the transfer waiting time between a bus arrival ($B^{start}_{r,t} + Run_r$) and the corresponding train to meet $T^{met}_{r,t}$, leaving enough time to walk, $walk_r$. Variables (a) to (c) are related to bus scheduling decisions while the variables (d) to (f) are dependent on bus schedules for timetable coordination decisions.

We minimise the excessive transfer passenger waiting time (Equation (6.1)) between two transit services (bus and train in this study) subject to the following constraints:

i. **Bus flow constraints:** Assigns binary values to all “possible” and “impossible” sequence of locations visited by a bus, $b_{ct} \in B$ (Constraints 6.2 to 6.8);
ii. **First bus and last bus constraints:** The first bus on route $r \in R$ must start within its maximum headway hw^+_r (Constraint 6.9) and the last bus trips must arrive at a station within its maximum headway from the end of the schedule horizon that is, $T_{max} - hw^+_r$ (Constraint 6.10)
iii. **Feasible connection constraints:** For a particular route, each bus must meet with a different train (Constraints 6.11 and 6.12)

iv. **Bus headway constraints:** The headway between two consecutive bus trips on a route $r \in R$ must fall within a minimum and maximum value hw_r^-, hw_r^+, respectively (Constraint 6.13);

v. **Run-time constraints:** Two consecutive start trips $t_1, t_2 \in TR_r$ of a bus $b_{ct} \in B$ on routes $r_1, r_2 \in R$ must be separated by a running time $Run_{r1,r2}$ between these routes (which includes inter-route deadhead time for interlining) plus a range of layover time $[min_{r2}^{lay}, max_{r2}^{lay}]$ (Constraint 6.14);

vi. **Train meets constraints:** Restrict earlier bus trips on a route to meet with earlier trains at the corresponding transfer station (Constraints 6.15 and 6.16);

vii. **Passenger volume constraint:** The number of transferring passengers are calculated between consecutive bus arrivals at a station (Constraint 6.17);

viii. **Transfer constraints:** The excessive passenger waiting time between a given bus-train connection pair is computed as the gap in time between a bus arrival and train departure, leaving a transfer walking time, $walk_r$ (Constraints 6.18, 6.19);

ix. **Null constraints:** The events that fall beyond the schedule horizon T_{max} are considered null; the parameter N_{pt} is the dummy train that buses meet with and N_{time} indicates the dummy bus arrival or train departure time (Constraints (6.20) and (6.21))

$$\min \quad Z = \sum_{r=1}^{R}\sum_{t=1}^{TR_r} passwait_{r,t} \qquad \forall r \in R, t \in TR_r \tag{6.1}$$

subject to:

$$flow_{1,0,r,t} \leq b_{ct} \qquad \forall r \in R, t \in TR_r \tag{6.2}$$

$$\sum_{r_2}^{R}\sum_{t_2}^{TR_r} flow_{r_1,t_1,r_2,t_2} = 1 \qquad \forall r_1 \in R, t_1 \in TR_r \tag{6.3}$$

$$flow_{r_1,0,r_2,t_2} = 0 \quad \text{where,} \quad r_1 > 1 \qquad \forall r_1, r_2 \in R, t_2 \in TR_r \tag{6.4}$$

$$flow_{r_1,t_1,r_2,t_2} = 0 \quad \text{where,} \quad t_2 > n_{r_2} \qquad \forall r_1, r_2 \in R, t_1, t_2 \in TR_r \tag{6.5}$$

$$flow_{r_1,t_1,r_2,t_2} = 0 \quad \text{where,} \quad t_1 > n_{r_1} \qquad \forall r_1, r_2 \in R, t_1, t_2 \in TR_r \tag{6.6}$$

$$flow_{r,t_1,r,t_2} = 0 \quad \text{where,} \quad t_2 \leq t_1 \qquad \forall r \in R, t_1, t_2 \in TR_r \tag{6.7}$$

$$flow_{r,t,r,t} = 0 \qquad \forall r \in R, t \in TR_r \tag{6.8}$$

$$T_0 \leq B_{r,1}^{start} + Run_r < (T_0 + hw_r^+) \qquad \forall r \in R \tag{6.9}$$

$$B_{r,t}^{start} + Run_r >= T_{max} - hw_r^+ \qquad \forall r \in R, t \in TR_r \tag{6.10}$$

$$B_{r,t}^{start} + Run_r > T_{s,j-1}^{dep} \quad \text{where,} \quad s = f_r \qquad \forall r \in R, t \in TR_r, s \in S, j \in 1..pt \tag{6.11}$$

$$B_{r,t}^{start} + Run_r <= T_{s,j_{last}}^{dep} \quad \text{where,} \quad s = f_r \qquad \forall r \in R, t \in TR_r, s \in S \tag{6.12}$$

$$hw_r^- \leq (B_{r,t+1}^{start} - B_{r,t}^{start}) \leq hw_r^+ \qquad \forall r \in R, t \in TR_r \tag{6.13}$$

$$flow_{r_1,t_1,r_2,t_2} = 1 \implies \begin{cases} B^{start}_{r1,t1} + Run_{r1,r2} + min^{lay}_{r2} \leq B^{start}_{r2,t2} \\ B^{start}_{r1,t1} + Run_{r1,r2} + max^{lay}_{r2} \geq B^{start}_{r2,t2} \end{cases} \quad \forall r_1, r_2 \in R, t_1, t_2 \in TR_r \tag{6.14}$$

$$T^{met}_{r,t} < T^{met}_{r,t+1} \quad \forall r \in R, t \in TR_r \tag{6.15}$$

$$T^{dep}_{s,j} \geq B^{start}_{r,t} + Run_r + walk_r \quad \forall r \in R, t \in TR_r \tag{6.16}$$

$$Pax_{r,t} = \begin{cases} 0, & \text{if, } t > n_r \\ P * (B^{start}_{r,1} + Run_r - B^{hour}_{r,1}) + \sum_{h < B^{hour}_{r,1}} P_{h,r} & \text{else if, } t = 1 \\ [P * ((B^{hour}_{r,t-1} + 1) - B^{start}_{r,t-1} + Run_r) - & \\ P * ((B^{hour}_{r,t} + 1) - B^{start}_{r,t} + Run_r)] & \\ \quad + \sum_{B^{hour}_{r,t-1} < h <= B^{hour}_{r,t}} P_{h,r} & \text{else.} \end{cases}$$

$$\forall r \in R, t \in TR_r \tag{6.17}$$

$$wait_{r,t} = T^{met}_{r,t} - (B^{start}_{r,t} + Run_r + walk_r) \quad \forall r \in R, t \in TR_r \tag{6.18}$$

$$passwait_{r,t} = Pax_{r,t} * wait_{r,t} \quad \forall r \in R, t \in TR_r \tag{6.19}$$

$$B^{start}_{r,t} = N_{time} \quad \text{where,} \quad t > n_r \quad \forall r \in R, t \in TR_r \tag{6.20}$$

$$T^{met}_{r,t} = N_{pt} \quad \text{where,} \quad t > n_r \quad \forall r \in R, t \in TR_r \tag{6.21}$$

The CP translation of the I-TTVS model in *MiniZinc* integrates the variables, constraints and objective as presented in Section 5.5 (Chapter 5) into a single model. The complete *MiniZinc* model for I-TTVS in presented in Appendix A.2 and the model-independent data in Appendix A.3, respectively.

6.5 Solutions Approach

As seen in Chapter 3, the formulated model for the BTCP can be solved directly in a small network instance with fewer decisions to make. But, in a larger PT network the BTCP is generally intractable owing to the large number of candidate solutions and conflicting constraints. Moreover, with multiple bus routes interlining and intersecting at train stations, a number of transfer opportunities are created that increases the search space exponentially in the number of routes. Given the hardness of the problem and the expanding network sizes in scope, heuristic optimisation methods seem promising to seek high quality solutions within acceptable computation times.

In this section, we propose a solution approach that combines the two-stage optimisation described in Chapter 4 with the *Large Neighbourhood Search* (LNS) meta-heuristic to solve the integrated problem of BTCP. We present a generic description of LNS in

Section 6.5.1, followed by the proposed meta-heuristic framework and its problem specific operators for the BTCP in Section 6.5.2.

6.5.1 Large Neighbourhood Search (LNS)

The LNS was proposed by Shaw (1998) and similar to other meta-heuristics, uses the principle of locating feasible solutions in the neighbourhood of an existing solution. LNS is considered a successful search method for large-scale problems due to its ability to search (or sample) in larger neighbourhoods. We are aware of the following studies that use LNS to solve transit planning sub-problems: Pepin et al. (2009) shows the competitiveness of LNS against other heuristic methods in solving the *Multi-Depot Vehicle Scheduling Problem* (MDVSP). Similarly, Bent and Van Hentenryck (2004) and Pisinger and Ropke (2007) use LNS to solve vehicle routing problems with time windows. We are yet to come across a study that uses LNS to solve the integrated problem of TT and VS with transfer coordination as a major objective.

For a problem instance I with a set of S finite feasible solutions, a *neighbourhood* of a solution $s \in S$ can be defined as $N(s) \subseteq S$, that is, the function N maps a solution s to the set of solutions S. LNS is based on the concept of finding improving solutions in the neighbourhood of an existing solution (Pisinger and Ropke, 2010). For example, in a minimisation problem, we want to find a solution $s' \in S$ such that the $cost(s') \leq cost(s)$, where the "$cost()$" function denotes the cost of a solution.

A LNS heuristic needs to start with an initial solution produced by any solver. It then runs an improvement step repeatedly, until a stopping criterion is met which can be a user-defined limit on the number of nodes visited, number of consecutive iterations, total computation time or when a solution improvement is not observed. The solution improvement step in LNS has two parts, both of which can be done in many ways:

- *destruction* of part of the solution: this operation deletes or removes certain parts of a solution, giving partial solutions.
- *reconstruction* of a complete solution: the partial solution is the reconstructed or repaired to a feasible solution.

The destroy and re-construct functions replace a former neighbourhood search by destroying (or removing) a part of the solution and re-constructing it to render feasible.

The generic schema of LNS is shown in Algorithm 1 inferring from Pisinger and Ropke (2010). The LNS heuristic takes an initial solution s to the problem instance I as input and makes it the current best solution s_{min} (line 1). The main body of the algorithm are constituted by lines 2 to 7; while the stopping criterion is not met, the algorithm iterates the process of *destroying* and *reconstructing* the current solution s, resulting in a new solution s' (line 3). In lines 4 to 7, the *destruction* and *re-construction* process is iterated using an acceptance criterion (a simple hill-climbing meta-heuristic) that is, the function $accept(s, s')$ accepts the new solution s' if it is at least as good as the current best solution s, that is, $cost(s) \leq cost(s')$. The best known solution is assigned to s_{min} and returned.

Algorithm 1: Large Neighbourhood Search (from Pisinger and Ropke (2010))

Input: problem instance I

create an initial solution $s_{min} = s \in S(I)$
while *stopping criteria not met* **do**
  $s' = r(d(s))$
  if *accept* (s, s') **then**
    $s = s'$
    if $c(s) < c(s_{min})$ **then**
      $s_{min} = s$
return s_{min}

The destroy function must aim to remove those parts of the solution that has the potential to improve a solution once re-construction is invoked. The "degree of destruction" is thus a vital choice when implementing destroy methods: if destruction is invoked only on a small part of the solution (and is essentially not a large neighbourhood), then the risk is that the current solution cannot be improved as the fixed part of the solution constraints the neighbourhood excessively. If a very large part of the solution is destroyed then it takes too long to find a solution for the neighbourhood and has the potential to degrade the very purpose of using LNS. The destroy method must be chosen in a way such that the search is diversified, increasing the possibilities of finding better solutions from the entire search space.

6.5.2 LNS for I-TTVS

In this section, we describe how we tailor the LNS meta-heuristic framework to solve the I-TTVS. We also focus on explaining the implemented neighbourhood definitions that are relevant to our problem.

Initial Solution: As mentioned in the previous section, we need an initial solution to invoke LNS as the LNS heuristic improves an existing solution. In order to find an initial solution, we can run a chosen solver until a feasible solution is found. This solution can then be used to start the search. It is generally useful to start LNS (or any local search technique) from a known, better solution. But it must also be noted that starting a local search from a better solution does not always result in the search yielding better results after a number of iterations. Sometimes a good solution can "trap" the local search in a region where there is little improvement, while a worse initial solution can give the local search more space to explore.

To aid the LNS from an already known good solution, we consider the Stage-2 solutions generated by the best decomposition strategies from Chapter 5 as initial solutions for LNS. Given this initial solution (indicated as s_{init}), LNS repeatedly considers a sub-problem of the entire BTCP and reconstructs it using a solver of choice (CP or MIP).

Destroy methods: For *destruction* it is necessary to choose a set of decision variables whose values will be changed. For a given bus route $r \in R$ and trip $t \in TR_r$, we consider two decision variables: the bus arrival time $B^{Start}_{r,t}$ and the corresponding bus flow sequence per route and trip $flow_{r1,t1,r2,t2}$. The *reconstruction* will find new values for these decision variables that still satisfy all the constraints (listed in 6.4) and improve the value of the objective. While several problem-specific destruction methods can be applied for the BTCP, we consider defining the following neighbourhoods that favour the search in desirable spaces:

i. `randomNBH`[2]: changes the $B^{start}_{r,t}$ and corresponding $flow_{r1,t1,r2,t2}$ on all the trips $t \in TR_r$ belonging to a randomly chosen route $r \in R$, and on the corresponding trip on all the other routes. Every time this destroy method is invoked, variables to reconstruct are selected uniformly at random from a given pair of $r \in R$ and $t \in TR_r$. This method may seem unfavourable as it could destroy the parts that are fitting to a solution. However, it still has the potential to diversify the search.

ii. `maxNBH`: removes the worst parts of the solution, that is, finds the bus route and trip which has the maximum total passenger wait time, and changes the $B^{start}_{r,t}$ and corresponding $flow_{r1,t1,r2,t2}$ on all the trips on that route, and on the corresponding trip on all the other routes. This is motivated by the need to reconstruct those variables that incur the largest objective costs.

iii. `combNBH`: switches between `randomNBH` and `maxNBH` uniformly at random. While neighbourhoods (i) and (ii) may not perform well individually as destroy methods, with `combNBH` we investigate the possibility of one of the neighbourhoods choosing to reconstruct those parts of the solution that were not searched by the other neighbourhood, thus enabling a more diverse search.

Neighbourhood size: As explained previously, the "degree of destruction" is an important input parameter for every destroy method. In this regard, we define a parameter σ that dictates the size of a neighbourhood in terms of the number of routes and trips to consider in each LNS iteration. In other words, it is the range around the chosen route r and the chosen trip t which controls how many routes and trips are added into the neighbourhood. We increase the degree of destruction by gradually increasing the σ value and test the efficiency of the defined neighbourhoods in re-optimising the variables.

Proposed LNS Algorithm: The proposed LNS algorithm for the I-TTVS is given in Algorithm 2. It begins by taking an initial solution $s_{init} \in S(I)$ as input and makes it the current and best known solution s_{min} (line 3). Lines 4 to 14 form the main body of the algorithm and implement a stopping criterion, which in our case is the maximum number of iterations, max_iterations. The function `randomNBH()` or `maxNBH()` or `combNBH()` returns a route-trip pair r_n, t_n (line 5) which define the neighbourhoods where the destroy methods are to be applied. In lines 7-10, we fix the LNS variables ($flow_{r_1,t_1,r_2,t_2}$) and $B^{start}_{r,t}$ to

[2] The term `"NBH"` here indicates "neighbourhood" of a candidate solution

values from the current and best known solution s_{min} for all routes $r_1, r_2 \in R$ and trips $t_1, t_2 \in TR_r$ outside the chosen LNS neighbourhood. These values are preserved in C for each neighbourhood (line 6). The new solution s is then computed using the $solve()$ function and updated as the current and best known solution if necessary (lines 11-13). Once the consecutive number of iterations are met, the current best solution is returned (lines 14-15).

Algorithm 2: Large Neighbourhood Search (LNS) algorithm for I-TTVS

Input: Problem instance I, an initial solution $s_{init} \in S(I)$, σ, max_iterations
Output: Improved solution s_{min}

1 **Begin:**
2 $l \leftarrow 0$
3 $s_{min} \leftarrow s_{init}$
4 **while** $l < max\_iterations$ **do**
5     $r_n, t_n \leftarrow$ `randomNBH()` or `maxNBH()` or `combNBH()`
6     $C \leftarrow \{\}$
7   **for** *each* $r_1, r_2 \in R \setminus \{r_n - \sigma, \ldots, r_n + \sigma\}, r_1 \neq r_2$ *and* $t_1, t_2 \in TR_r \setminus \{t_n - \sigma, \ldots, t_n + \sigma\}, t_1 \neq t_2$ **do**
8         $C \leftarrow C \cup I(flow_{r_1,t_1,r_2,t_2}) = s_{min}(flow_{r_1,t_1,r_2,t_2})$
9   **for** *each* $r \in R \setminus \{r_n - \sigma, \ldots, r_n + \sigma\}$ *and* $t \in TR_r \setminus \{t_n - \sigma, \ldots, t_n + \sigma\}$ **do**
10         $C \leftarrow C \cup I(B^{Start}_{r,t}) = s_{min}(B^{Start}_{r,t})$
11     $s \leftarrow solve(I_{cons} \cup C)$
12   **if** $cost(s) < cost(s_{min})$ **then**
13         $s_{min} = s$
14     $l \leftarrow l + 1$
15 **return** improved solution s_{min}

6.6 Computational Experiments

In this section, we detail the generation of solutions using the I-TTVS approach through a set of computational experiments. We evaluate the performance of the proposed LNS meta-heuristic algorithm by testing it on our case study area in the City of Wyndham, Melbourne for problem instances with varying PT network sizes namely, *Small* and *Large*. We investigate whether the decomposed solutions for the BTCP (as presented in Chapter 5) can be improved further using heuristics. Essentially here, the Stage-2 solutions generated using the best decomposition strategy by *fixing the bus headways* ($Z \leftarrow z_5$) are used as initial solutions (s_{init}) to invoke the LNS. All experiments are run on a personal computer with Intel Core i7-7600 @ 2.9GHz CPU and 16GB RAM. While the mathematical models are translated into solver specific CP or MIP models written in *MiniZinc*, we use *Python 3.9* to implement the LNS part of the algorithm.

All instances in this study are solved using the *Gurobi* solver and each LNS iteration is run for a maximum CPU time of $\delta = 10$ minutes. We choose the *Gurobi* solver to enable "warm-starts", that is, start the search in a given neighbourhood from a known

Stage-2 solution. All three neighbourhoods namely, `randomNBH`, `maxNBH` and `combNBH` require the same data-inputs and the parameter σ decides the number of route-trip pairs to consider in each neighbourhood, in every iteration. Upon testing multiple values of σ on the given problem instances, $\sigma = 2$ proved to yield the best neighbourhood sizes and the most improvement under LNS. A summary of all the solutions obtained from each neighbourhood at different σ values is presented in Appendix B.

Table 6.1 shows the solutions generated by the proposed heuristic when initiated with the decomposed solutions s_{init} for the *Small* and *Large* network instances. It also compares the solutions against each neighbourhood at $\sigma = 2$ and $\delta = 10$ minutes.

Instance: *Small* (6 *X* 4)

We first tested the proposed LNS algorithm in the *Small* PT network instance comprising 6 bus routes and 4 transfer stations, for each input number of buses. A minimum of 7 buses is required to solve this instance; with less than 7 buses there were no feasible solutions that satisfy the constraints listed in Section 6.4. With 9 buses, we observe that the `combNBH` is able to reach the true optimum value of 2,764 passenger-minutes.

			`randomNBH`	`maxNBH`	`combNBH`
Network Instance	No. of buses	s_{init}	$\sigma = 2$	$\sigma = 2$	$\sigma = 2$
Small (6 *X* 4)	6	*Infeasible*			
	7	15,175	5,398	6,775	6,143
	8	6,691	3,149	5,360	3,716
	9	5,542	3,503	3,324	2,764
Large (24 *X*5)	25	44,544	29,578	37,616	30,936
	26	21,354	18,885	21,354	20,823
	27	18,144	17,336	18,144	17,643
	28	15,084	10,135	8,253	8,134
	29	16,352	12,430	7,365	7,395
	30	12,562	8,404	8,465	6,526
	31	12,101	9,030	6,355	6,055
	32	13,610	7,423	5,754	5,832
	33	13,793	10,156	5,576	5,823

Table 6.1: Objective value comparison with $\delta = 10$ minutes and $\sigma = 2$ in each LNS neighbourhood

Instance: *Large* (24 *X* 5)

In the next step, we tested the capability of our I-TTVS model and LNS algorithm in accommodating the *Large* PT network, comprising 24 bus routes and 5 transfer stations. The pure integer-linear FlatZinc model generated by *MiniZinc* for the *Large* instance is 12.2 times larger than that for the *Small* instance.

With the LNS framework, better solutions are obtained within 10 minutes of CPU time per iteration with buses as few as 25, in comparison to decomposition. The results from Table 6.1 indicate that the three defined neighbourhoods perform well in minimising the objective value, starting from the solutions generated by decomposition and consistently improving on them. The `combNBH` again yielded consistently improving results for each input number of buses, with an objective value reduction up to 5,823 pass-mins with 33 buses. This confirms the potential of this neighbourhood in diversifying the search, in comparison to the other two. The `maxNBH` fares as well as the `combNBH` with the objective value minimised upto 5,576 pass-mins with 33 buses, although the solution improvements are slightly unsteady, possibly owing to the limitations with finding the maximum waiting time values in a given neighbourhood. The `randomNBH` on the other hand could only demonstrate minimal improvement in solutions within the given time limit. Figure 6.2 compares the objective values generated against each input number of buses, using the three LNS neighbourhoods.

Evidently, we do not have a proven optimum or even a best lower bound for the complete problem of BTCP for the *Large* instance. Thus, using LNS based algorithm, we could find better solutions for the complete problem in comparison to direct solving and problem decomposition, but it sacrificed the proof of optimality.

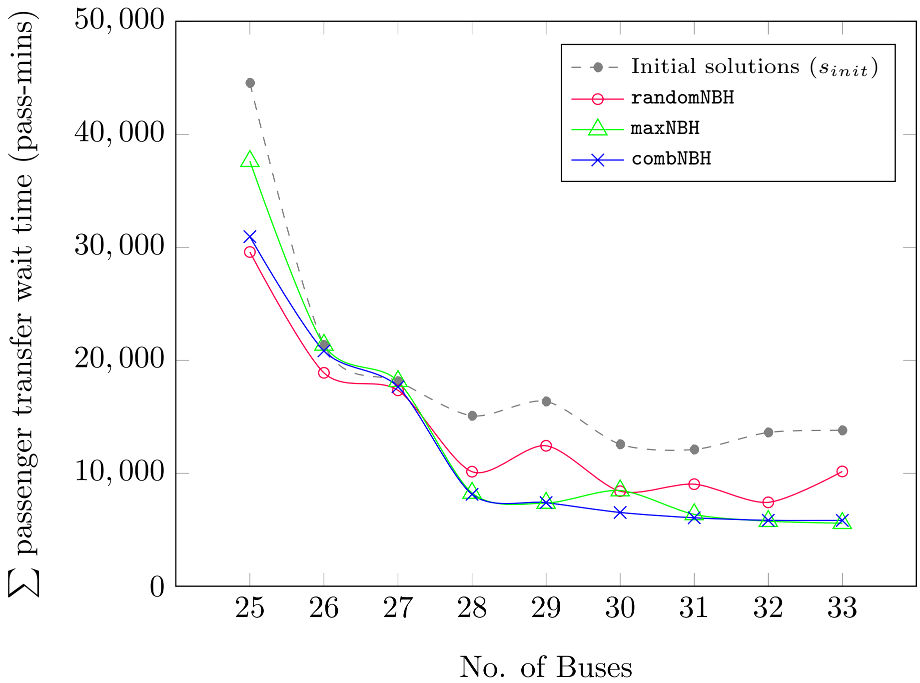


Figure 6.2: Best Objective value comparison with different LNS neighbourhoods for the *Large* network instance

6.7 Conclusions

In this chapter, we present an integrated timetabling and vehicle scheduling approach (I-TTVS) aimed to achieve efficient timetable coordination by minimising the total transfer waiting time incurred by passengers transferring between buses and trains. This approach is based on simultaneously solving the timetabling problem (finding bus schedules that are well-coordinated with the train schedules and at minimum deviation from the target service headway) and vehicle scheduling (finding cost-efficient bus schedule to serve the entire network), accommodating a variety of real-world constraints.

We deployed a LNS based optimisation framework to find improving solutions for the integrated problem. The proposed heuristic starts by using an initial decomposed solution and iteratively re-optimises this solution in order to reach better objective values. The integrated model and its LNS extension were tested on instances of varying network sizes in the City of Wyndham, Melbourne. In all our cases, the proposed LNS framework yielded improvement in solutions within the stipulated running time limits.

This study leaves several avenues for future research: the search efficiency can be improved by defining multiple destroy methods that are specific to the BTCP. For example, multiple alternate decision variables can also be chosen for LNS destruction, for example the choice of trains to meet. Additionally, although the proposed LNS algorithm demonstrates good convergence, there is scope for improving the solutions further. In this regard, a previously obtained solution can be reused to accelerate the solution search on subsequent bus instances. That is, say, the solution at the end of 30 minutes with 25 buses can be used to start the LNS with 26 buses, so on and so forth.

The upcoming Part III: *Timetable Evaluation* discusses the real-world application of our optimisation framework and evaluates the benefits achieved from the optimised scheduling solutions in terms of passenger service and operator requirements.

Part III

Timetable Evaluation

Chapter 7

Evaluation of Optimised Timetables

7.1 Introduction

The second part of this thesis (Part II) presented an optimisation framework to model and solve the *Bus-train Timetable Coordination problem* (BTCP) comprehensively, using real-world scheduling requirements. Findings suggested that identifying operable and practical constraints is critical in generating scheduling solutions that result in the simultaneous optimisation of transfer connectivity and bus fleet size requirements. The BTCP can be decomposed into timetabling and vehicle scheduling sub-problems without losing the problem compatibility with each other. In addition, even the best decomposition strategy can be improved further using meta-heuristic search methods such as the *Large Neighbourhood Search* (LNS) in order to yield good-quality solutions at reasonable computation time.

Part III of this thesis consists of this chapter, where we detail the application of the developed BTCP models in a real-world public transport network in the City of Wyndham, Melbourne. Through multiple assessments using the desired scheduling components inferred from the industry, we demonstrate how the developed models and optimisation techniques can be effective in achieving a reasonable trade-off between timetable coordination and operator cost efficiency. In doing so, this chapter addresses research objective RO_4 and the associated sub-question as stated below:

RO_4 : To evaluate the quality of well-coordinated bus-train timetables

- (4a) What are the benefits of optimised timetable coordination in terms of passenger service and operational cost?
- (4b) How does the developed optimisation framework compare with current commercial practices?

This chapter is structured as follows: In Section 7.2, a comparison of scheduling solutions from the SD-TTVS and I-TTVS approaches (as introduced in Chapters 5 and 6 respectively) is made and several evaluation criteria are laid down in terms of passenger service and operator cost benefits. In Section 7.3, we devise a *MiniZinc* model with an objective to solve the BTCP such that the solutions under the best-case scenario from a commercial scheduling software, HASTUS *NetPlan*[1] are input to evaluate this objective. This chapter concludes with some final remarks on the evaluation of optimised timetables in Section 7.4.

7.2 Benefits of Improved Timetable Coordination

In this section, we express the performances of the proposed decomposed and integrated approaches to solve the BTCP (namely, SD-TTVS and I-TTVS approaches in Chapters 5 and 6 respectively) in terms of passenger service benefits and bus resource optimisation. To demonstrate the impact of improved timetable coordination, the following evaluation criteria are used:

i. Trade-off between timetable coordination and bus fleet-size
ii. Feasible connections and average wait time
iii. Bus headway regularity
iv. Bus route interlining and layover limits

We first examine (i) the trade-off between timetable coordination and bus fleet size requirements from the *MiniZinc* generated optimised solutions. For all the remaining criteria (from ii to iv), we refer to the best results obtained for the 25 bus instance from solving the *Large* network comprising 24 bus routes and 5 train stations.

The baseline timetables (that is, the original scheduled bus and train timetables in the City of Wyndham in 2017) are constructed under a wider set of scheduling constraints and requirements than those that are modelled in *MiniZinc*. For example, in the real-world, the bus fleet size is also used to operate certain school trips during the AM and PM peak in addition to depot pull-in and pull-out requirements. We are hence unable to draw a fair comparison between the *MiniZinc* optimised solutions with the baseline timetables. The evaluations presented here are thus a scenario based subset of a bigger problem and cannot be used to determine the real quality of bus timetables in Wyndham entirely. We put our focus on the value that our *MiniZinc* optimisation framework brings in general on bus-train timetable coordination and operator cost savings.

7.2.1 Timetable Coordination vs bus fleet-size

It is crucial to obtain solutions that can simultaneously optimise the balance between bus operating cost and quality of timetable coordination. Figure 7.1 shows the trade-off between total passenger waiting times (in passenger-minutes) and bus fleet-size requirements as obtained by solving (a) decomposed (SD-TTVS: Stage-1 and Stage-2) and (b) integrated (I-TTVS: Stage-2 + LNS) models for the BTCP. Among the series of test runs using the proposed models, it is evident that the best and consistently improving solutions are reported by implementing the LNS framework on the integrated problem. Moreover,

[1] https://www.giro.ca/en-ca/our-solutions/hastus-software/hastus-for-planners/

it can be observed that the inclusion of variability in bus headway enables better coordination with trains, thus resulting in minimised total passenger waiting times. In any case, the minimum bus fleet size required to serve the sub-network in scope is found to be 25 buses. While the savings in total passenger waiting time are larger as more buses are dispatched, beyond 28 buses, this reduction is only minimal. Thus, any additional bus resource may prove futile.

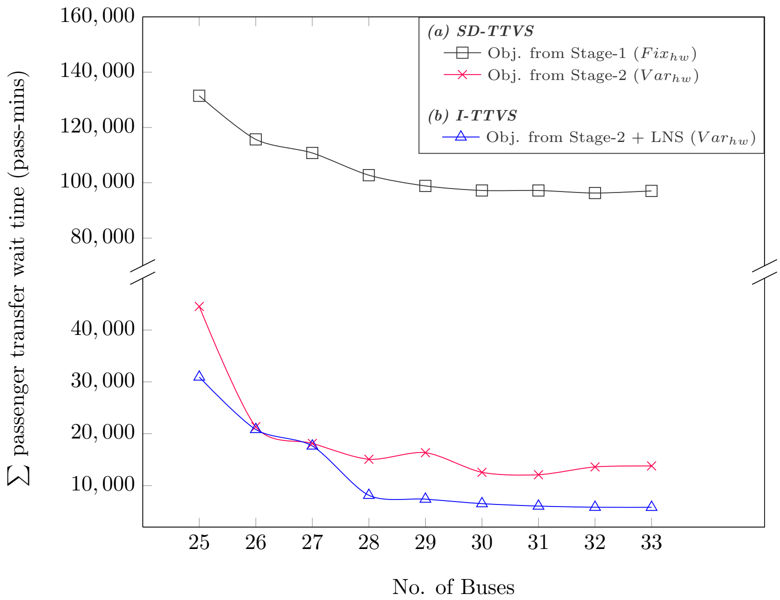


Figure 7.1: Comparison of trade-off between passenger waiting time (mins) and bus fleet size for the BTCP solved using (a) SDTTVS and (b) I-TTVS

7.2.2 Feasible connections and average wait time

In this section we analyse the connectivity levels for the Wyndham sub-network in scope in terms of the feasible number of met trains and average waiting time per passenger. Table 7.2 shows the average optimised transfer waiting time per passenger (inclusive of transfer walking time) for each bus route-station pair between 7:00AM to 3:00PM.

Considering the 24 bus routes and 5 train stations within the scope of this research, the optimised bus-train waiting time per passenger averages to 2.74 minutes. Figure 7.2 shows the distribution of average transfer waiting time per passenger for all optimised bus trips. Note that to account for late bus arrivals or less reliable train departures, we incorporate a buffer time of 5 minutes beyond the minimum walk time for all transfers. This also agrees with the optimisation criteria adopted to create the baseline timetables in reality. Moreover, every bus trip considered has a feasible train meet. Approximately 86% trips show low waiting times per transfer- in the range 0-5 mins, giving a perfect coordination in the correspondence between bus and train services. Around 11% of total trips fall under a waiting time of 5-10mins and the remaining 14% under 10-12 mins.

At every transfer station, the average waiting times are at a reasonable minimum with optimisation and kept below an average threshold of approximately 8 mins. We note

Route	**Transfer Station**	**Met trips**	**Avg. wait per passenger (mins)**
150	Williams Landing	15	1.53
150_A	Tarneit	15	3.13
151	Williams Landing	12	1.17
151_A	Tarneit	12	2.33
153	Hoppers Crossing	12	7.33
153_A	Hoppers Crossing	12	7.42
160	Hoppers Crossing	15	1.87
160_A	Tarneit	15	5.00
161	Hoppers Crossing	12	1.00
161_A	Werribee	12	1.00
166	Hoppers Crossing	12	1.33
166_A	Wyndham Vale	12	2.08
167	Hoppers Crossing	12	1.33
167_A	Tarneit	12	1.08
170	Werribee	24	2.00
170_A	Tarneit	24	3.50
180	Werribee	24	2.21
180_A	Tarneit	24	2.12
190	Werribee	24	2.50
190_A	Wyndham Vale	24	4.08
191	Werribee	10	3.40
191_A	Wyndham Vale	10	5.40
192	Werribee	12	2.08
192_A	Wyndham Vale	12	1.33
Weekday 7:00am-3:00pm		**368**	**2.74 mins**

Table 7.2: Average per passenger wait time from *MiniZinc* optimised bus timetables for the chosen Wyndham sub-network

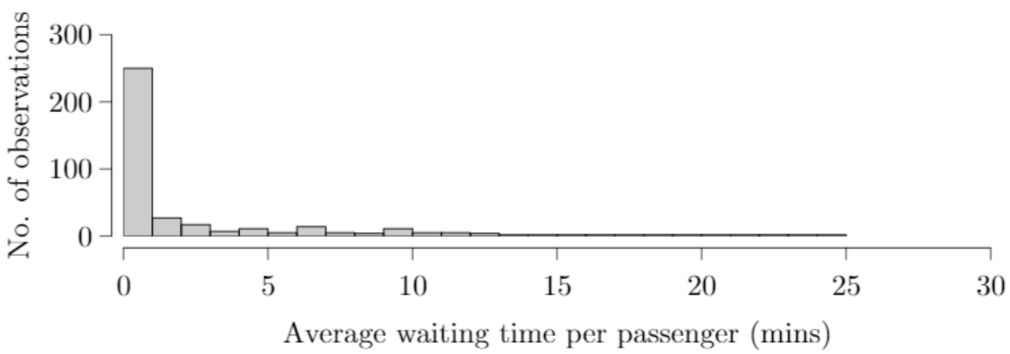


Figure 7.2: Histogram of average transfer waiting time per passenger for all the trips under *MiniZinc* optimised timetables

that the optimised waiting time reduction at Hoppers Crossing for low frequency routes 153 and 153_A is minimal. This can be looked at as a classic example of the cascading effects of coordination between multiple routes when interlining is enabled, as every other route connection time at Hoppers Crossing (and other stations) is much lower. Since the algorithm is able to optimise both peak and contra-peak transfers together (due to the consideration of equal coordination priorities in both directions), coordination is offered for all locations chosen (and not just the "key" transfer points) without impacting the overall solution quality.

7.2.3 Bus headway regularity

While specifying variable headway to serve a bus line, it is necessary to ensure that we avoid excessive deviations in bus headway compared to the industry specified targets. Planners seek to find good quality timetables where the bus trips are at minimum deviation from the target service headways; a concept that is termed *regularity*. Larger deviations mean longer waiting times and are undesirable. Conversely, a shorter than specified headway may seem desirable for a passenger, but from an operator point of view, excessively short headways result in waste of bus resources. With an allocated headway tolerance of ±20% from target for all the routes in scope, the average optimised headway deviation per route is observed to be 5% in the AM-Peak and 1% in the Inter-peak.

Figure 7.3 compares the target and optimised bus headway for an example route 150 connecting at Tarneit station. Having ensured in the model that there is ample headway flexibility at the transition point between AM and inter-peak (7th and 8th bus trips in this example), the optimised schedule demonstrates smooth bus headway transitioning. Moreover, with a headway tolerance allowance of ±20% from target, the algorithm enables minor shifts in timetable to ensure better coordination with trains. For example, as shown in the figure, the optimised schedule shifts the bus headway from 20 mins to 24 mins at the 4th trip such that it aligns this trip closer to the next train departure. Such shifts give the passengers an opportunity to plan ahead and time their departures from the origin (bus stop) and be guaranteed with a train meet, which is more desirable than waiting longer at a transfer location later.

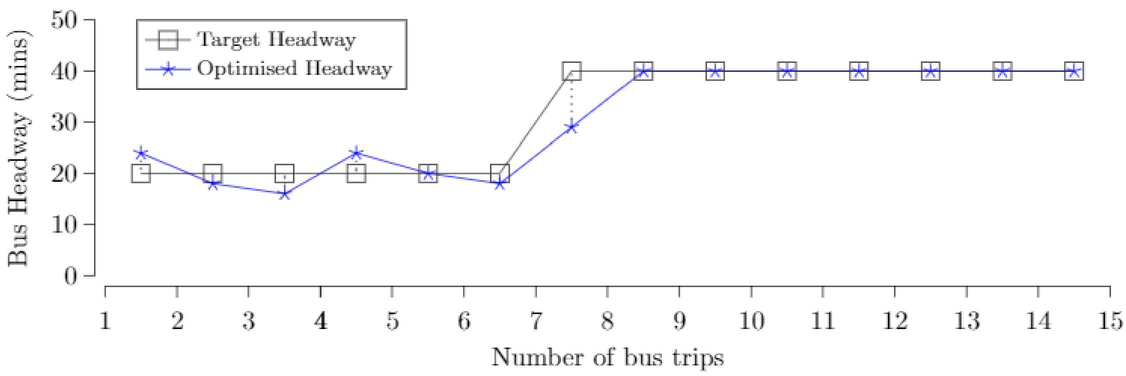


Figure 7.3: Deviation between target and optimised bus headway for example bus route 150

7.2.4 Bus Route Interlining and layover limits

Between consecutive trips, we allow a bus to layover for a minimum of 10% route running time and a maximum of an additional 15 minutes. This figure is considering a short-term recovery time between bus trips to facilitate service reliability and does not cater to driver meal break or bus standby requirements. Since our model allows the bus trips to freely interline between multiple routes, we do not precisely identify which trip must succeed or precede the other.

Figure 7.4 shows an example for an optimised bus block. The values in blue boxes indicate the bus arrival time at each station at the end of a route; red and green lines indicate layover and deadhead respectively. For example, route “160A” starts from Hoppers Crossing and finishes at Tarneit at 10:37, deadheads to Werribee, lays over from 10:51 until 10:57 (6 minutes) and starts the next route “190A” (Werribee to Wyndham Vale) at 10:57. Upon optimisation with 25 buses, layovers constitute an average 17% of total bus hours, with the average values ranging from 2.9 to 7.2 mins in the network. Notably, even with no maximum limits on layover, the optimiser is able to align bus trips as close as possible, incurring very minimal increase in total layover time as compared to using a 15 min limit. This demonstrates the efficiency of the proposed algorithm in reaping savings in unproductive service time, thereby putting the dispatched buses to their maximum use.

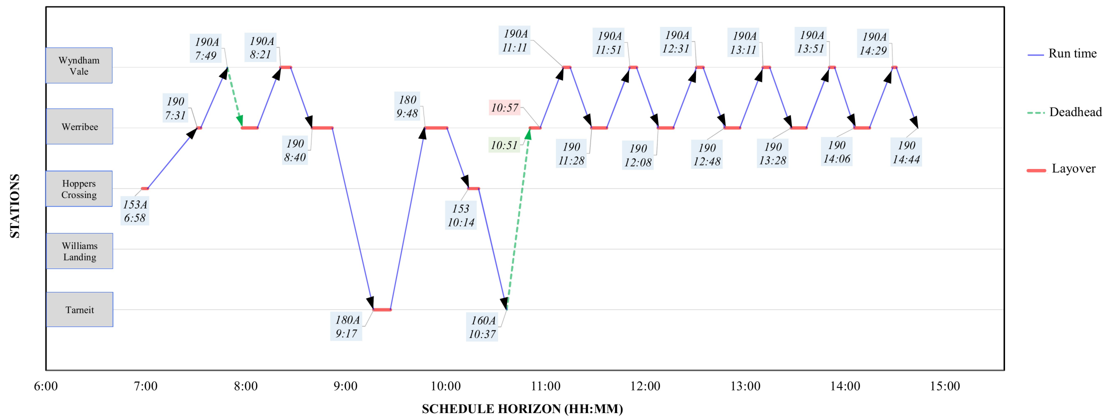


Figure 7.4: Time space diagram showing the activity of a single bus interlining and coordinating at transfer stations in the schedule horizon

7.3 Evaluation using HASTUS-*NetPlan*

This section explains how automated scheduling tools like *NetPlan* can be used to evaluate the planning and optimisation scenarios attempted in this research. First, we outline the features of *NetPlan* including the most relevant data inputs and optimisation criteria that enables the generation of cost-efficient scheduling solutions. Next, we detail how we configure *NetPlan* to solve our problem of bus-train timetable coordination. We create a network planning model for our case study area: City of Wyndham in Melbourne and discuss the optimisation procedure adopted in *MiniZinc* using the outputs extracted from *NetPlan*. The purpose of this exercise is to demonstrate the scalability *MiniZinc* solutions to a real-world application and the ability to produce good solutions at reasonable computational time.

7.3.1 HASTUS-*NetPlan* features

We choose the GIRO-HASTUS scheduling system (Blais and Rousseau, 1988) (v.2014) in order to evaluate the quality of solutions produced using our optimisation framework.

The HASTUS system is very popular in Australia and utilised extensively by various transit agencies for public transport planning and scheduling decisions. The *Network and Timetable Planner* module in HASTUS called "*NetPlan*" (Fleurent and Lessard, 2009) is a service planning tool that focuses on maximising user-defined transit connectivity requirements and minimising the vehicle and/or crew resources required to operate a network of services. It creates and optimises timetables based on a set of network and scheduling constraints desirable to a transit agency and evaluates the trade-off between vehicle operating costs and transfer efficiencies for existing and/or potential new networks. Notably, it facilitates the simple creation of public transport routes and timetables that can be easily modified to compare the cost and quality of scheduling solutions with respect to various planning scenarios.

***NetPlan* data**

To begin with, *NetPlan* takes as input the public transport network features in a graphical representation called the "Connections Diagram", which is a schematic representation of the routes and stations (called "Places") in a given network. *NetPlan* can perform optimisation on existing vehicle schedules in a network or construct new timetables based on certain network specifications. For this exercise, we choose the latter technique, where trips are built based on the headway criteria specified for Wyndham.

We create the headway based "trip builders" for 24 bus routes in scope, consistent with the service specifications for Wyndham as provided by the DoT. The primary inputs for this exercise include: (i) a schedule horizon from 7:00AM to 3:00PM, (ii) fixed[2] and uniform bus route headways in AM-peak and inter-peak using which the number of bus trips are deduced, (iii) a run-time matrix indicating the journey time of a bus route from its origin to destination, (iv) a deadhead matrix indicating the deadheading time requirements[3], (v) bus layover specifications at 10% of route run time, and (vi) to define connectivity requirements, the "meet-builders" for each route are specified with a minimum and maximum transfer waiting time range at a given transfer place (or transfer station). In *NetPlan*, transfer waiting time is inclusive of transfer walking time which is hence not input additionally. We also consider equal coordination priority for each transfer station in scope. Note that we generate connections that provide the highest feasible train meets per bus route in order to extract the *NetPlan* outputs at its best working version. This meant relaxing the connecting wait time ranges with trains significantly to ensure that all bus trips have a feasible train to meet. Appendix C provides a detailed summary of how we configure *NetPlan* to solve the problem of BTCP in Wyndham, including the data inputs, network features and the extracted outputs. Note that we do not incorporate transfer passenger volume and garage requirements for this particular exercise.

[2]It must be noted that by fixed headway, we imply the target headway values that are input and even for a specific time period in the schedule horizon (inferred from the industry). For example, 20 minutes and 40 minutes fixed headway per trip in the AM and inter-peak respectively.

[3]Data source: Google Maps

Optimisation using *NetPlan*

NetPlan performs optimisation[4] by evaluating timetable shifts by pair of "trip builders". For each route, trip builders are shifted iteratively at their origin places, that is, the route start time are changed systematically. The order of optimisation then involves shifting all trips belonging to the created trip builders one after the other, until all possible shifts are tested. One of the key disadvantages with this approach is that localised trip shifting is not undertaken due to the use of *fixed bus headways.* Thus, any shift in starting time on a route carries forward to all the other trips uniformly, which potentially reduces the possibility of certain bus trips to be aligned closer to the trains, especially when there is apparent disharmony between bus and train headways. In the real-world, planners would utilise *NetPlan* to first generate bus trips to a certain pattern and search for the lowest number of vehicles, and provide the 'best connectivity' for those resources. However, consequently this procedure often requires some trips to be shifted manually where the connectivity is observed to be lacking.

The main outputs from *NetPlan* optimisation include (i) an objective cost, which is expressed as an approximation of the generalised cost of operating vehicles for a given schedule horizon; *NetPlan* optimisation minimises this objective cost, subject to user-defined constraints on deadheading, minimum layover times and coordination, (ii) the shifted timetables including start and end times of each trip, (iii) a minimum number of buses required to serve the given network and the corresponding bus blocks, and (iv) a "Synchronisation Quality Index" or SQI as an objective measure of the overall transfer connectivity in a given network which includes the sum of scores for each connection, weighted according to its priority.

7.3.2 Devising a *MiniZinc* model using *NetPlan* data

In order for*MiniZinc* to tackle the same coordination problem defined in *NetPlan* as nearly as possible, we ensure that the modelling data inputs, parameters and constraints used to generate solutions are comparable and consistent with each other. Note that *NetPlan's* optimisation objective for integrated timetabling and vehicle scheduling is different from the defined *MiniZinc* objective. As proposed by Fleurent and Lessard (2009), the *NetPlan* objective comprises two components: the first parts evaluates the pure timetabling part of the problem without considering the vehicle scheduling costs; the second part optimises the vehicle schedules over the trips defined by timetables obtained in the first part. More importantly, this objective allocates certain weights to both timetabling and vehicle scheduling components, representing the relative importance for each.

On the other hand, ***MiniZinc's* optimisation objective** is concerned with the simultaneous optimisation of the timetabling and vehicle scheduling components, with no

[4]Note: There is limited literature on the heuristics adopted by GIRO to perform *NetPlan* optimisation. All the information provided in this section are a combination of first-hand experience with *NetPlan*, under the guidance of industry experts and majorly based on personal interviews with industry experts from the DoT and Phillip Boyle and Associates during 2017-2018. We acknowledge the developments on *NetPlan* algorithm since the release of 2014 version.

weights attached to each. Naturally, this cannot be used to evaluate the quality of scheduling solutions from *NetPlan* but the purpose of this comparison is to indicate that the performance of our models and associated solvers in finding good-quality solutions for large-scale problems are at least comparable to the best available commercial solutions. In order to evaluate the *NetPlan* solutions with our objective, the devised *MiniZinc* model is instantiated using these bus starting times. In other words, we compare the solutions produced by our model with those from *NetPlan under* MiniZinc*'s optimisation objective*.

We utilise the *fixed headway* decomposition strategy developed in Chapter 5, where we compute the objective cost per route and infer the trains to meet in order to solve the full problem of minimising transfer costs. Recall from Chapter 4 that we define transfer costs as the total excessive passenger waiting time for all bus-train transfers in the given network and operator costs in terms of minimum number of buses to run the timetabled trips, along with the unproductive service time like layovers and deadheads. The value of the *MiniZinc objective*- in passenger wait times - is uniquely determined by the bus starting times for each trip on each route. Given a bus fleet size, *MiniZinc objective* also seeks a feasible bus blocking, and reports failure if there is no such blocking.

7.3.3 Objective comparison: ***NetPlan*** vs ***MiniZinc***

Upon optimisation using fixed bus headway, *NetPlan* finds feasible solutions with buses as few as 26. Predominantly for this exercise, the most important *NetPlan* product that serves as an input to our *MiniZinc* model are the *bus starting time per route* as inferred from this 26 bus block (Table C.4, Appendix C).

Table 7.3 shows the objective values from the devised *MiniZinc* model. Under Stage-1 (Fix_{hw}) are the best solutions found by *MiniZinc* using fixed bus headway. Since *NetPlan* reports a minimum bus fleet size of 26, we used this as a reference and tested our model too down to 26 buses. The Stage-1 objective value when initiated with *NetPlan* generated bus starting times is observed to be 260,172 pass-mins and the corresponding *MiniZinc* objective (where bus schedules are decision variables) compare well with *NetPlan* at a further minimised passenger waiting time of 114,855 pass-mins.

With uneven train frequencies at any given time in the schedule horizon, it is disadvantageous to consider a perfectly consistent bus frequency for coordination in Melbourne. Bus arrivals will need to be harmonised with the train departures in a way that minimises total passenger waiting time. From the results reported in Table 7.3, we test the benefits of allowing bus headway variability in Stage-2 (Var_{hw}) in enabling good bus-train coordination as compared to using fixed bus headway in Stage-1. We permit a headway tolerance of 20% enabling the bus headway for all routes to span between a minimum and maximum range. Solving Stage-2 with the choice of trains to meet known from Stage-1, we get significantly better objective values, thus showing the additional benefits of relaxing the unnecessarily fixed bus headway constraint. Notably here, our model is also able to produce feasible solutions with *buses as few as 25*, within a reasonable CPU time of 30 mins. This confirms the efficiency of our algorithm in reaping not only further minimised waiting times but also at one bus savings.

We demonstrated in Chapter 6 that although a proposed decomposition strategy for the BTCP may already be good, we can further improve the solution by implementing a meta-heuristic search method. We repeat the same methodology here, where the *Large Neighbourhood Search* method is implemented to solve the integrated problem of BTCP, using the best Stage-2 solutions as initial solutions. In Table 7.3, the values under I-TTVS show the results for the devised *MiniZinc* model solved using the I-TTVS approach and LNS. Particular to this exercise, we observe the best neighbourhood to be the `combNBH` at a neighbourhood size of $\sigma = 3$. We perform 10 LNS iterations at a time limit δ of 10 mins each. Figure 7.5 summarises the results from all stages of optimisation.

	SD-TTVS				*I-TTVS*
	Stage-1 (Fix_{hw})		**Stage-2** (Var_{hw})		**Stage-2 + LNS** (Var_{hw})
No. of buses	CPU time (s)	Obj (pass-mins)	CPU time (s)	Obj (pass-mins)	Obj (pass-mins)
33	268	91,999	102	10,583	8,032
30	355	91,999	136	10,835	7,258
29	654	92,031	127	10,970	10,256
28	1807	92,071	122	11,214	7,946
27	1806	107,203	1807	16,678	10,898
26	1807	114,855	1807	16,895	11,125
25	1807	161,928	1807	22,028	20,019

Table 7.3: Results from SD-TTVS and I-TTVS models solved using *NetPlan* data

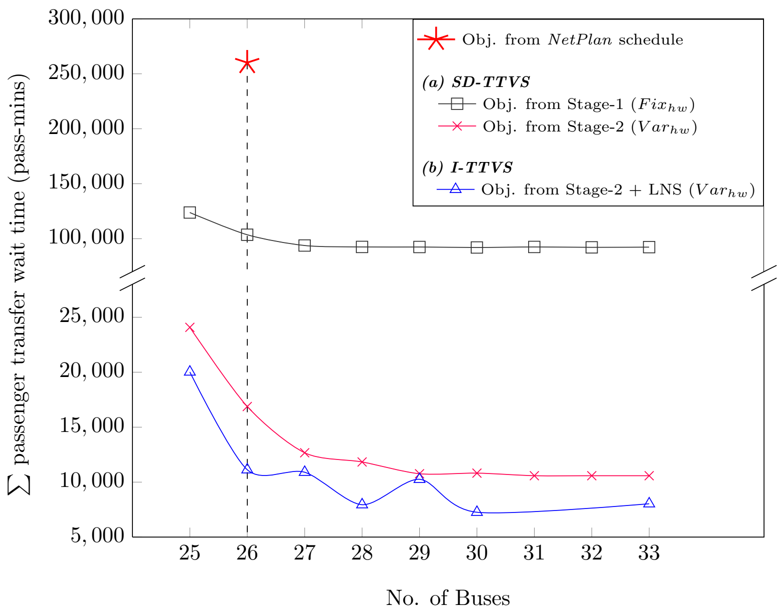


Figure 7.5: Comparison of trade-off between passenger waiting time (mins) and bus fleet size for the BTCP solved using the devised *MiniZinc* model on *NetPlan* data

7.4 Final Remarks

This chapter details the application of the proposed optimisation framework for timetable coordination in a real-world public transport network in the City of Wyndham, Melbourne. We identify that the bus-train sub-network in scope has a prevalence of non-harmonic frequencies and a potential to improve timetable coordination. Through several evaluation criteria, we demonstrate how the developed models and algorithm can be effective in achieving well-coordinated timetables and operator cost efficiency simultaneously.

We achieve the following optimisation goals that are of significance to effective decision making in transit planning:

(1) We ensure maximum connectivity that is, every bus trip modelled in this problem has a feasible train connection and within a waiting time range of 0-12 mins per passenger. Along with peak transfers, the algorithm optimises contra-peak transfers as well, with minimal adverse effects on the overall quality of connections in both directions.

(2) We avoid excessive variation in bus headways compared to the desired targets. With an allocated headway tolerance of ±20% from target for all the routes in scope, the average optimised headway deviation per route in the AM-peak and Inter-Peak are 5% and 1%, respectively.

(3) We allow the buses to freely interline between multiple routes and minimise excessive layover time between their trips by specifying a reasonable recovery time of minimum 10% route running time and an additional 15 minutes as maximum. Considering the optimised bus schedules, layovers constitute 17% of total bus hours with the average values in the network ranging from 2.9 to 7.2 mins.

This chapter then evaluates the quality of *MiniZinc* optimised solutions using *NetPlan*, a commercial scheduling tool. We arrive at the following main findings from this assessment that indicate that the devised *MiniZinc* model gives solutions for large-scale PT problems which appear comparable with the commercial state of the practice:

(1) Bus fleet size
    (a) feasible solutions found with *buses as few as 25* at a CPU time of 30mins, resulting in one bus savings in comparison to *NetPlan* optimisation with *26 buses.*
(2) Total passenger waiting time (pass-mins)
    (a) Fixed bus headway with decomposition: improvement in total passenger waiting time in comparison to *NetPlan* solutions (reduction from 260,172 pass-mins to 114,855 pass-mins for the 26 bus instance);
    (b) Variable bus headway with decomposition: further improvement in total passenger waiting time demonstrating the benefits of accommodating variability in bus headways (reduction from 114,855 pass-mins to 16,895 pass-mins for the 26 bus instance);
    (c) Variable bus headway with integration: further improvements in total passenger waiting time demonstrating the efficiency of the proposed LNS model in integrating the timetabling and vehicle scheduling sub-problems (reduction from 16,895 pass-mins to 11,125 pass-mins for the 26 bus instance);

The overall findings from this chapter indicate that improvements can be made to inter-modal connectivity at minimal impact on operational costs, by allowing a degree of flexibility to the most important scheduling constraints. In addition, the proposed optimisation framework is able to not only achieve the simultaneous optimisation of timetable coordination and bus resource optimisation, but also do so within a reasonable computational effort.

Part IV

Conclusions

Chapter 8

Discussion and Conclusions

8.1 Introduction

Public transport (PT) in Melbourne involves complex and interlinked series of bus and train networks serving multiple stations with a number of transfer connectivity requirements. The aim of this thesis has been to investigate the simultaneous optimisation of public transport timetable coordination and vehicle efficiency. The optimisation framework proposed in this thesis act as a decision making aid for planners to choose multiple, problem-specific service preferences such that comprehensive and integrated timetabling solutions are arrived at.

The first part (Part I) examined the background and theory related to public transport planning and operations. It identified the key gaps in knowledge regarding the limitations with existing modelling and solving techniques for the public transport timetable coordination problem and identified scheduling constraints that represent the problem comprehensively. Part II forms the crux of this thesis by proposing an optimisation framework that comprises a comprehensive mathematical model specifically for the *Bus-train Timetable Coordination Problem* (abbreviated as BTCP) along with two novel methodologies to solve the BTCP, incorporating industry favoured, practical requirements that can optimise both coordination and cost-efficiency with minimal adverse impact on transit service quality. Part III of this thesis comprised one chapter to evaluate the quality of solutions obtained from the proposed optimisation framework and its comparison to the state of the practice.

The fourth and final part of this thesis (Part IV) is the research discussions and conclusion, containing this chapter. In Section 8.2 we re-visit the aim of this research and provide a summary of key findings in relation to this aim and associated research objectives. We also detail here the key gaps in knowledge and the contributions emerging from this thesis and assesses the strengths and weaknesses of the proposed approaches for transit timetable coordination. In Section 8.3, we describe the implications of this research in theory and in practice. The key study limitations, directions for future research and the potential for model transferability are outlined in Sections 8.4 to 8.6. This thesis concludes in Section 8.7.

8.2 Summary of key findings and contributions

As described in Chapter 1, this thesis investigates the following research question:

Can we generate timetables and vehicle schedules that optimise both timetable coordination and operator cost efficiency simultaneously?

To answer this question, we identified four research objectives (RO) and a number sub-questions as follows:

RO_1 : To understand the existing challenges with public transport timetable coordination.

- (1a) What are the limitations with existing approaches in solving the timetable coordination problem?
- (1b) What modelling constraints represent the problem realistically?

RO_2 : To formulate the timetable coordination problem incorporating real-world scheduling constraints.

RO_3 : To solve the timetable coordination problem using state-of-the-art optimisation techniques.

RO_4 : To evaluate the quality of optimised bus-train timetables.

- (4a) What are the benefits of optimised timetable coordination in terms of passenger service and operational cost?
- (4b) How does the developed optimisation framework compare with current commercial practices?

A review of literature in Chapter 2 presented an overview of the stages of public transport planning and the need to integrate these multiple stages. Much emphasis is laid on the transfer optimisation components of timetabling and vehicle scheduling, followed by exploring the current state of the art and state of the practice in finding comprehensive solutions for the public transport timetable coordination problem. This chapter addresses the first research objective RO_1: (1a) by identifying the critical need to tackle the complexity of the timetable coordinating problem and the limitations with current practices in solving them holistically. Collectively, it was established in the context of large scale PT planning and operations that in spite of the complexity associated with transfer optimisation, the majority of existing approaches often involve simplified instances that result in vulnerable models that do not represent real world variability in transit systems. Many studies focus on theoretical problems, which are great foundations to understand the problem complexity and the performance of the solutions proposed. However, current research tends to lack the flexibility required to match the passengers' needs with that of the operators. Inferring from past studies and drawing from requirements prioritised by our industry partners at the Department of Transport (DoT)-Victoria, in Melbourne, Australia, we also address research objective RO_1: (1b) by prioritising those real-world scheduling requirements most relevant to this research.

Chapter 3 presents the overall research approach adopted to address the gaps in knowledge and research opportunities identified in this thesis. Three research phases were outlined: *Phase 1* established a core understanding of the key research gaps, questions and scope. *Phase 2* focused on the development of the mathematical models and optimisation techniques for the timetable coordination problem. *Phase 3*, focused on model testing, validation and evaluation on the case study area.

Addressing RO_2, Chapter 4 explains first a generic model to compute public transport transfer costs that work as a framework to define our research problem. In particular, we model the tactical and operational levels of public transport planning with multiple decisions concerned with determining cost efficient *timetabling* (TT) and *vehicle scheduling* (VS) solutions. We contribute to the state of the art by proposing a *Bus-Train Coordination Problem* which is formulated as a mathematical model incorporating multiple a set of scheduling requirements that have received less attention in the literature so far. We also investigate the scalability of the model with commercial CP or MIP solvers. Inferring from the findings, the solvers for complex problems such as the BTCP require computational effort that, in the worst case, grows exponentially with the size of the problem. Consequently, for larger problems it soon becomes impractical to complete a proof of optimality. Therefore, the challenge is to search for feasible and good quality solutions of such large problem instances in a reasonable time-frame. This limitation serves as a scaffold for the next Chapters 5 and 6 in Part II, where we introduce two novel techniques to solve the BTCP in large scale PT network instances.

While there is ample research on the traditional sequential approach to transit planning, the literature is relatively silent on the various ways a planning problem can be decomposed and/or re-ordered in order to get tractable solutions without compromising the benefits of problem integration, especially on large networks. Chapter 5 addresses this gap by developing a sequential but re-ordered modelling approach for timetabling and vehicle scheduling, with emphasis on timetable coordination. In this study, we explore a key research opportunity in understanding the various ways the BTCP can be decomposed and re-ordered to render it solvable on large network instances. We design a two-stage optimisation approach called the *Sequential-Decomposed Timetabling and Vehicle Scheduling* (SD-TTVS) and formulate the problem with multiple decisions concerned with determining cost efficient timetabling and vehicle scheduling. Through SD-TTVS, we contribute to the state of the art by: (i) proposing five ways the comprehensive BTCP problem can be decomposed (ii) testing the developed method in a real-life case study for the metropolitan bus and train services in the City of Wyndham, Melbourne. Numerical results indicate that the BTCP can be decomposed into timetabling and vehicle scheduling sub-problems without losing the problem compatibility with each other. The solutions from each decomposition strategy at 30min of computation time yield substantially better objective values than solving the full problem with a general purpose solver for long periods of computation time. We observe that the best decomposition strategy is to deduce the objective cost of each route first using fixed headway (which is computationally less complex) and then to solve the full problem using the choice of trains met, returned from solving this objective. With this approach, however, we lack the potential to assess the quality of solutions produced especially for larger network instances; each sub-objective of the decomposition strategy produces an optimum solution against itself and there is no guarantee that the overall objective derived from these are globally optimum. This leads us to the necessity to introduce heuristics based solution search techniques, which forms the next chapter.

The integration of consecutive steps in planning is a predominant research path explored since the last decade or so. To date, the solutions to large-scale transit network problems that include a combination of sub-problems have been mostly reliant on the use of various heuristic, meta-heuristic and matheuristic approaches where the solution search schemes are based on a collection of design guidelines, criteria established from past experiences, and cost and feasibility constraints. However, the inclusion of detailed practical considerations still remain a challenge. Especially in planning large scale networks, compromises between tractability and problems integration are among the major points of interest that invite more investigation. Chapter 6 addresses this gap by developing an *Integrated Timetabling & Vehicle Scheduling* (I-TTVS) with an emphasis on timetable coordination. Using I-TTVS, we contribute to the state of the art by: (i) proposing a *Large Neighbourhood Search* based meta-heuristic algorithm to solve the BTCP through integrated timetabling and vehicle scheduling; (ii) testing the developed method in a real-life case study for the metropolitan bus and train services in the City of Wyndham, Melbourne.

Chapters 5 and 6 collectively address the research objective RO_3, by proposing novel optimisation techniques to render the formulated BTCP solvable and scalable to large network instances. The proposed solution approaches predominantly focus on solving both decomposed and integrated versions of the problem in *MiniZinc* using subsets of constraints defined for the complete problem. We also explore here the efficient and inbuilt solving capabilities within *MiniZinc* that can reduce the computational effort in finding feasible, good quality solutions faster.

Chapter 7 evaluates the quality of the generated scheduling solutions. By addressing RO_4, this chapter demonstrates how the developed models and optimisation techniques can be effective in achieving simultaneous timetable coordination and operator cost efficiency. Discussions on the benefits of having achieved improved coordination follow. In addition, the optimised schedules are evaluated using a commercial scheduling tool called HASTUS *NetPlan* to demonstrate the comparability of our model to current state of the practice.

Summarising the key findings in relation to the identified research gaps, the main contributions emerging from this thesis to the state of the art of transfer optimisation in public transport are as follows:

- An understanding of the compromises with modelling the timetable coordination problem comprehensively and the limitations with solving the problem in the context of large-scale transit networks;
- A comprehensive mathematical model for the BTCP incorporating real-world scheduling requirements (C_1);

- A novel two-stage optimisation approach comprising sequential and decomposed timetabling and vehicle-scheduling sub-problems to solve the timetable coordination problem (C_2);
- A LNS based meta-heuristic approach comprising integrated timetabling and vehicle scheduling sub-problems to solve the timetable coordination problem (C_3)

Table 8.1 provides a summary of key findings and contributions in relation to the corresponding research objectives.

Research Objectives	***Contributions***	***Key Findings***
RO_1: To understand the existing challenges with public transport timetable coordination RO_2: To formulate the timetable coordination problem incorporating real-world scheduling constraints	An efficient and applicable mathematical model for timetable coordination that has the ability to incorporate multiple, conflicting objectives and real-world scheduling requirements (Chapter 4)	• Existing approaches often involve simplified instances that result in models that do not represent real-world variability in transit systems • Identifying operable and practical constraints is critical in generating scheduling solutions that result in the simultaneous optimisation of transfer connectivity and bus fleet size requirements.
RO_3: To solve the timetable coordination problem using state-of-the-art optimisation techniques	A sequential, decomposed timetabling and vehicle scheduling (SD-TTVS) approach for timetable coordination incorporating multiple practical scheduling constraints (Chapter 5)	• The BTCP can be decomposed into timetabling and vehicle scheduling sub-problems without losing the problem compatibility with each other, where only a minimum amount of information is needed to be passed from one sub-problem to the other.
	An integrated timetabling and vehicle scheduling (I-TTVS) approach for timetable coordination such that it incorporates multiple practical scheduling constraints (Chapter 6)	• Although compatible decomposition is beneficial, even the best decomposition strategy can be improved further using meta-heuristic search methods such as the *Large Neighbourhood Search* (LNS) in order to yield good-quality solutions at reasonable computation time.
RO_4: To evaluate the quality of optimised bus-train timetables	• An understanding of the existing state of the practice regarding bus-train timetable coordination in Melbourne • An understanding of the quality of optimised bus schedules in terms of passenger and operator benefits	• Supports the hypothesis that incorporating flexibility in scheduling constraints leads to better solutions, ensuring simultaneous cost and connectivity optimisation.

Table 8.1: Summary of research questions and associated contributions and key research findings arising from this thesis

8.3 Implications for theory and practice

This work has laid out the basis for a practical and useful scheduling tool to address the real-world concerns of schedulers. As a theoretical contribution, we present a constraint based optimisation model that is capable of analysing a wide range of real world scenarios, as detailed in Chapter 4. It addresses the complexity of public transport vehicle interactions and provides cost-effective schedules that are well coordinated temporally. Through demonstrated experiments, we also support the hypothesis that incorporating realistic scheduling requirements and allowing some constraint tolerances can significantly minimise the passenger transfer waiting times and the bus fleet required to serve the entire network, thereby improving timetable coordination between services.

When automating scheduling procedures, having case-specific testing options for schedulers is highly valued, in order to arrive at quicker, cohesive decisions with very little manual intervention. The multiple strategies to model and solve the BTCP, as presented in Chapters 5 and 6 enable such a decision making system, that coincides with the scheduler's requirements. In a real world setting, the data independent nature of our model written entirely in *MiniZinc*, and its ability to consider a wide range of such "what-if" scenarios also makes it possible to represent a variety of constraints realistically. This thereby provides an aid to a scheduler in analysing and comparing the trade-offs between passenger service quality and resource cost optimisation without relying heavily on human intuition.

Optimised timetables and its compliance reflect the quality of transit services. The outcome of this research intends to provide public transport schedulers and operators with robust and flexible solutions to coordination problems, which assist in improving the overall public transport network reliability and patronage. The potential impacts of devising efficient optimisation criteria for transit problems include faster and reliable running PT; increase in ridership; reduction in travel time; reduction in car usage etc. In a bigger picture, inferences from this research can maximise the utility of PT services in order to enable modal shift from private cars to public transport.

8.4 Limitations

While the devised research approach is capable of addressing the research questions that are core to this thesis, the following modelling, solving and evaluation limitations need to be acknowledged.

8.4.1 Modelling limitations

While we consider those constraints that are prioritised for the objective of timetable coordination in this study, the optimisation goals designed for this study can be sometimes competing. A few examples are given below:

- In Wyndham, train frequencies are higher than bus frequencies and upgrading every bus route to meet every train on a line is not feasible. Thus, we do not penalise those trains that do not have feasible bus connections. Hence, we adopt a more realistic objective to maintain a reasonably regular bus frequency with minimum available resources and ensure maximum connectivity for these buses.
- Some bus routes could connect with trains on multiple train lines for the same bus travel direction. For example, a bus trip might pass one or more stations mid-route and then terminate at the destination. It is often difficult to optimise connections at both of those stations. One potential solution to this issue is to force the buses to wait at the mid-route station(s) such that a connection from a second train can be ensured. However, this is disadvantageous to through-riding passengers.
- Sometimes the travel time, the need to coordinate with trains, and limited opportunities to interline will combine to dictate that long layovers times are unavoidable

for certain routes. In these cases the network planner will want to consider making the route shorter or longer to enable more efficient operations.

We exclusively consider the dis-utility of passengers who transfer between two scheduled services. A major challenge in scheduling for improved coordination is to also prioritise the benefits of non-transferring (or through) passengers taking a service. Optimisation for transferring passengers may assign a bus to wait at a station until a train arrives, which could be undesirable for non-transferring passengers. This directs us to the understanding that the needs of transferring and non-transferring passengers must be weighed against each other.

The planning and modelling approaches presented in this thesis focus mostly on improving bus service frequency and its harmonisation with the trains. Less attention is dedicated the improvements that has the potential to arise from better transit infrastructure provisions (like accessible stations, direct bus routing etc) and capacity. This is basically because the former approach is cheap, modest and quick to implement than the latter, which is expensive and time consuming. From a policy point of view, implementing the former must be prioritised before the latter such that the required infrastructure is utilised to its complete potential.

8.4.2 Solving limitations

From the experimental results presented through Chapters 4 to 6, the complexity with solving a combinatorial problem such as the BTCP is evident. Even with advanced optimisation methods, there is no guarantee that the generated solutions are close to the true global optimum, especially considering large network instances. For the current case-study area, we have demonstrated the efficiency of the proposed decomposed and meta-heuristic methods in reaping a reasonable trade-off between connectivity an operator cost, at acceptable computational time. However, its performance on a larger, more dynamic transit network may require the incorporation of additional heuristics to intelligently search for the best candidate solutions at minimum computational efforts.

8.4.3 Limitations with evaluation

There are limitations with measuring the real benefits of improved bus-train coordination in Wyndham as there are resources, constraints and requirements that are not captured in our model yet. In the real world there might also be additional scheduling requirements, such as a desire to cycle buses onto routes that run past depots, consider timetables that allow for efficient driver rostering, restrict certain fleet types to certain routes, operate certain school trips, etc. These requirements were out of the research scope designed for this thesis either due to lack of data or the computational complexity it adds to the model. As our case-study area does not comprise the complete PT network in Wyndham, the data for in-service hours and bus hours specific to the chosen sub-network were not available. This limited the possibility of a direct comparison. Hence, the evaluation criteria presented in Chapter 7 is only limited exclusively to the benefits gained from our optimisation.

8.5 Future Research

Several opportunities for future research exist. First, alternate ways of decomposing the full problem can be investigated. For example, the bus flow sequences can be inferred from Stage-1 as an approximation to the full problem. In doing so, the loss of solution quality when solving multi-stage decomposition can also be investigated. The proposed optimisation models can be generalised to accommodate multiple planning periods, including the PM-peak where it is necessary to prioritise train to bus connections. One could further investigate a number of additional scheduling scenarios such as incorporating variable running times and generating timetables on a real-time setting. One could also aim to include crew scheduling constraints into the model, to ensure that there exist feasible crew schedules for the optimised timetable and vehicle schedules. This would however require extensive research on efficient computation techniques and a strong understanding of the practical requirements concerned with the problem.

In the context of optimisation and transport planning, the following additional goals can be designed:

- What impact does the improved timetable coordination have on increasing the public transport patronage and thereby enabling a modal shift from private cars? There is significant value in quantifying the benefits of high levels of coordination on a network-level.
- How can we determine the optimal timetabled transfer time range for any particular train-bus combination? This decision will mostly be a function of walking distance or time and punctuality of an arriving service. But it comes with a risk of long transfer wait times if the arriving services are highly unreliable.
- A potential derivative question to the above issue here is to explore the growth of feeder bus services (and its patronage) if we improve the punctuality of train arrivals/departures.
- As an industry criteria, the optimised headway must average out to the target headway value for individual time blocks (say AM-peak alone) than just the entire schedule horizon. For this, additional optimisation criteria will need to be undertaken and we acknowledge this is an opportunity for future research.

8.6 Model Transferability

The models developed in this thesis were mainly explored in a context of bus-train transfers in Melbourne PT networks, but are abstract enough to widen its application to alternate scheduled services (such as bus-bus, tram-train, etc.). Moreover, it would be conceivably possible to expand the network scope further, considering bigger sub-sets of networks in Melbourne; the corresponding problem would however be huge, and achieving this from the computational viewpoint requires further research.

A scheduling problem such as the BTCP attempted in *MiniZinc* is simple to develop, easy to extend and can be used to compute feasible solutions for large sets of data. As we see in Appendix A.3, the model and data are often separated, to improve the re-usability of the model for different classes of similar problems (such as objective satisfaction and minimisation/maximisation). The data-independent nature of the models also make it easy to accommodate a wide range of “what-if” operational scenarios. To do this when necessary, a planner would only need to perform minimal and desired modifications at the data level.

8.7 Final Remarks

This thesis provides useful mathematical models and solution methods for PT timetable coordination, with an emphasis on bus and train services. The findings from this research can be looked at as an important addition to the existing set of scheduling tools with which to address the public transport timetable coordination problem. The modelling capabilities demonstrated in this thesis suggest that optimisation techniques have advanced to the point where multiple operational scenarios can be tested and validated quickly. Such a feature is valuable to inform long-term decision making in the field of public transport planning and operations. It is important to note here that while such automation helps us easily handle a wide variety of scenarios with low testing time, the most efficient service schedule is ultimately created also combining the practical expertise of a scheduler. Thus, in a bigger picture, bridging the gap between scheduling in principle and scheduling in practice is highly relevant.

Appendices

Appendix A

Optimisation Framework in *MiniZinc*

A.1 The SD-TTVS models

This section presents the *Sequential-Decomposed Timetabling and Vehicle Scheduling*, (abbreviated as SD-TTVS) modelled in *MiniZinc*. Section A.1.1 shows the Stage-1 model with the sub-objective to minimise route costs with fixed bus headways (z_5). Section A.1.2 shows the Stage-1 model with the remaining sub-objectives (z_1 to z_4). Subsequently, Section A.1.3 shows the Stage-2 model that solves the full objective of BTCP (Z) using the information on the choice of trains to meet from Stage-1.

A.1.1 Stage-1: Fixed headway sub-objective z_5

```
include "globals.mzn";
%-------------------------------------------------------------%
% Case: City of Wyndham, weekday transfers from 7am to 3pm
%-------------------------------------------------------------%
% INPUT DATA
int: bct;              % bus fleet size
int: sct;              % total number of stations
int: tct;              % total number of train departures
int: tlast;            % last train departure time
int: route;            % bus routes
int: pt;               % number of physical trains
int: tmax;             % maximum schedule horizon
int: tothours;         % total number of hours

% Null parameters
int: Nroute= route+1;
int: Ntct= tct+1;
int: Ntime = tmax + 1000;
int: Nbct = bct+1;
int: Nhour = tothours+1;

```

```
% SETS
set of int: TRAINS = 1..Ntct;              % set of all physical trains
set of int: LOCATIONS = 1..Nroute;         % set of all locations
set of int: MAXTIME = 0..Ntime+max(walk); % set of all times
set of int: ROUTES = 1..route;             % set of all routes
set of int: TRIPS = 1..max(trips);         % set of all bus trips
set of int: BUSES = 1..bct;                % set of all buses
set of int: HOURS = 1..tothours;           % set of all hours

% INPUT ARRAYS
array [ROUTES] of int: trips ;             % bus trips on each route
array [1..sct, 1..pt] of MAXTIME: ttime;  % train departure times
array [ROUTES] of int: r_sct;              % coordinating station
array [ROUTES] of int: walk;               % walk time to the station
array [ROUTES] of int: layover;            % routewise layover time
array [HOURS] of int: hours;               % consecutive hours

% Runtime
array [1..route, 1..route] of int: RT_data;
array [LOCATIONS,LOCATIONS] of int: RT=
array2d(LOCATIONS,LOCATIONS,
    [(if i=Nroute then 0 elseif j=Nroute then 0 else RT_data[i,j] endif)
    | i in LOCATIONS, j in LOCATIONS]);
array [ROUTES] of int: RT_Coord ;

% Bus to train passengers
array [HOURS, ROUTES] of int: Pass_data;

% Headways
array [ROUTES, TRIPS,1..2] of int: headways;
array [ROUTES,TRIPS] of int: headways_mid =
    array2d(ROUTES,TRIPS, [(headways[r,t,1]+headways[r,t,2]) div 2|r in
   ROUTES,t in TRIPS]) ;
array [ROUTES] of int:last_train_time =
      [max([ttime[r_sct[r],j]|j in 1..pt where ttime[r_sct[r],j]<Ntime])
    | r in ROUTES] ;
array[ROUTES] of int: latest_start =
     [last_train_time[r] - (sum([headways_mid[r,t]|t in 1..trips[r]-1])+
   RT_Coord[r]+walk[r])
       | r in ROUTES] ;
array[ROUTES] of int: earliest_start =
       [1-(RT_Coord[r]+walk[r]+headways_mid[r,1])
       | r in ROUTES] ;
int: max_coord= max([RT_Coord[r]+walk[r]|r in ROUTES]) ;
int: minstart = min(earliest_start) ;
int: maxstart = max(latest_start);
int: maxdiff = max([latest_start[r]-earliest_start[r]|r in ROUTES]) ;

function 1..pt:next_train(ROUTES:r1,TRIPS:t1,minstart..maxstart:s) =
    if t1=1 then
      min([k + pt*bool2int(ttime[r_sct[r1],k] < s+RT_Coord[r1]+walk[r1])
   | k in 1..pt])
```

```
    elseif t1 <= trips[r1] then
      min([k + pt*bool2int(ttime[r_sct[r1],k] < s+sum([headways_mid[r1,t]|t in
      1..t1-1])+RT_Coord[r1]+walk[r1])| k in 1..pt])
    else pt
    endif  ;

%------------------- Bus Arrival
% The first bus trip on route r must arrive at a time between
% the beginning of time horizon and a certain maximum headway
% on that route since the beginning.
% All buses on a route must meet different actual trains
function int: barr(ROUTES:r,TRIPS:t,minstart..maxstart:s) =
          if t=1 then s+RT_Coord[r]
          elseif t <= trips[r] then
              s+sum([headways_mid[r,t1]|t1 in 1..t-1])+RT_Coord[r]
          else Ntime-max_coord
          endif;

array [ROUTES, minstart..maxstart] of  bool: feas_start =
     array2d(ROUTES,minstart..maxstart,
     [barr(r,1,s) <= headways[r,1,2]
      /\ barr(r,trips[r],s) >= tmax - headways[r,trips[r]-1,2] % Last bus arrives within max headway of tmax
      /\ barr(r,trips[r],s) +walk[r] <= last_train_time[r]
      /\ increasing_trains_met(r,s)
       | r in ROUTES,s in minstart..maxstart]) ;

test increasing_trains_met(ROUTES:r,minstart..maxstart:s) =
     forall(i in 1..trips[r]-1)(next_train(r,i,s)<next_train(r,i+1,s));

function int :waitBT(ROUTES:r,TRIPS:t,minstart..maxstart:s) =
     ttime[r_sct[r],next_train(r,t,s)] - (barr(r,t,s)+walk[r]);

function int: bushour(ROUTES:r,TRIPS:t,minstart..maxstart:s) =
     max([h  * bool2int(barr(r,t,s) > hours[h]) | h in HOURS]);

function int: buspass(ROUTES:r,TRIPS:t,minstart..maxstart:s) =
    if t > trips[r] then 0
    elseif not feas_start[r,s] then sum([Pass_data[h,r]*60|h in HOURS])
    elseif t=1 then  let {int:ht1=bushour(r,1,s)} in
        (Pass_data[ht1,r]* (barr(r,1,s)-hours[ht1])
        + sum(hs in HOURS)(bool2int(hs<ht1)*Pass_data[hs,r]*60))
    else let {int:ht1=bushour(r,t,s), int:ht0=bushour(r,t-1,s)} in
        ((Pass_data[ht0,r]*(hours[ht0]+60- barr(r,t-1,s))
        - Pass_data[ht1,r]*(hours[ht1]+60- barr(r,t,s)))
        + sum(hs in HOURS)(bool2int( hs > ht0 /\ hs  <= ht1) *60* Pass_data[hs,r]))
    endif  ;

array [ROUTES, minstart..maxstart] of int: all_route_costs =
    array2d(ROUTES,minstart..maxstart,
```

```
    [sum(t in 1..trips[r])(buspass(r,t,s)*waitBT(r,t,s))
    | r in ROUTES,s in minstart..maxstart ]);

% Runtime is inclusive of dead head time
% Layover time is at 10% runtime
bool: LAY = true;   % true= with layover; false= w/o layover
array [ROUTES] of int:lay =
    [if LAY then layover[r] else 0 endif | r in ROUTES];

%---------------------------------
% Decision Variables
%---------------------------------
int: min_coord = min([RT_Coord[r]+walk[r]|r in ROUTES]);
array [ROUTES] of var minstart..maxstart: BStart;

int: final_trip = max(trips)+1;
array [ROUTES,0..max(trips),ROUTES,1..final_trip] of var 0..1: flow;

%---------------------------------
% Constraints
%---------------------------------

% Set to 0 all impossible flows
% Flows from r,0 where r>1
constraint forall(r1,r2 in ROUTES,t2 in 1..final_trip where r1>1)
    (flow[r1,0,r2,t2]=0);

% Flows to non-existent bus trips
constraint forall (r1,r2 in ROUTES, t1,t2 in TRIPS where t2>trips[r2])
    (flow[r1,t1,r2,t2]=0) ;

% Flows from non-existent bus trips
constraint forall (r1,r2 in ROUTES, t1,t2 in TRIPS where t1>trips[r1])
    (flow[r1,t1,r2,t2]=0) ;

% Flows from a trip to itself  (this is a redundant constraint)
constraint forall(r in ROUTES, t in TRIPS)(flow[r,t,r,t]=0);
constraint forall(r in ROUTES, t1, t2 in 1..trips[r] where t2<=t1)
    (flow[r,t1,r,t2]=0) ;

% Flow constraints
constraint forall (r1 in ROUTES, t1 in 1..trips[r1])
    (sum([flow[r1,t1,r2,t2]|r2 in ROUTES, t2 in 1..final_trip])=1) ;
constraint forall(r2 in ROUTES, t2 in 1..trips[r2])
    (sum([flow[r1,t1,r2,t2] | r1 in ROUTES, t1 in 0..trips[r1]]) = 1) ;

% Bus fleet size constraint
constraint sum([flow[1,0,r2,t2]| r2 in ROUTES, t2 in 1..trips[r2]]) <=
   bct ;

% If the same bus does two trips, then there must be time (RT+lay)
% to complete the first trip and start the next
```

```
constraint forall(r1,r2 in ROUTES, t1 in 1..trips[r1], t2 in 1..trips[r2
    ])(
    (flow[r1,t1,r2,t2]=1 ->
       (BStart[r2] - BStart[r1] >=
        (sum([headways_mid[r1,t]|t in 1..t1-1])
       - sum([headways_mid[r2,t]|t in 1..t2-1]) )
       + RT[r1,r2] + lay[r2]))
      /\
    (flow[r1,t1,r2,t2]=1 ->
        (BStart[r2]-BStart[r1] <=
         (sum([headways_mid[r1,t]|t in 1..t1-1])
       - sum([headways_mid[r2,t]|t in 1..t2-1]) )
       + RT[r1,r2] + lay[r2] + 15)
       )) ;

%--------------------------------
% Stage 1 Objective (z5)
%--------------------------------
array [ROUTES] of  int: maxcost =
    [max([all_route_costs[r,s] | s in minstart..maxstart])| r in ROUTES
    ];
function var 1..max(maxcost) :
    route_cost(ROUTES:r) = all_route_costs[r,BStart[r]];

var 0..sum(maxcost):obj = sum([route_cost(r)|r in ROUTES]);
solve minimize obj;

%--------------------------------
% Output
%--------------------------------
output
["{\n"] ++
["\"Trainmeets\" : ["] ++
[show([next_train(r,t,fix(BStart[r])) | t in TRIPS]) ++
      if r==max(ROUTES) then "" else "," endif | r in ROUTES] ++
      ["],\n"]++
["\"BStart\": "]++ [show(BStart) ++ ",\n"] ++
["\"Obj\" : \(obj)"  ] ++
["\n}"];
```

A.1.2 Stage-1: Sub-objectives z_1 to z_4

```
include "globals.mzn";
%---------------------------------------------------------%
% Case: City of Wyndham, weekday transfers from 7am to 3pm
%---------------------------------------------------------%

% INPUT DATA
int: bct;               % bus fleet size
int: sct;               % total number of stations
```

```
int: tct;            % total number of train departures
int: tlast;          % last train departure time
int: route;          % bus routes
int: pt;             % number of physical trains
int: tmax;           % maximum schedule horizon
int: tothours;       % total number of hours

% Null parameters
int: Nroute= route+1;
int: Ntct= tct+1;
int: Ntime = tmax + 1000;
int: Nbct = bct+1;
int: Nhour = tothours+1;

% SETS
  set of int: MAXTIME = 0..Ntime+max(walk);  % set of all times
  set of int: ROUTES = 1..route;             % set of all routes in the network
  set of int: TRIPS = 1..max(trips);         % set of all bus trips
  set of int: BUSES = 1..bct;                % set of all buses
  set of int: HOURS = 1..tothours;           % set of all hours

% INPUT ARRAYS
array [ROUTES] of int: trips ;            % bus trips on each route
array [1..sct, 1..pt] of MAXTIME: ttime;  % train departure times
set of int: TTIME = {ttime[s,k] | s in 1..sct,k in 1..pt where ttime[s,k]<Ntime} ;
array [ROUTES] of int: r_sct;             % coordinating station
array [ROUTES] of int: walk;              % walk time to the station
array [ROUTES] of int: layover;           % routewise layover time
array [HOURS] of int: hours;              % consecutive hours

% Runtime
array [1..route, 1..route] of int: RT_data; % runtime data for each route pair
array [ROUTES] of int: RT_Coord ;

% Passengers
array [HOURS, ROUTES] of int: Pass_data; % bus to train volume


array [ROUTES, TRIPS,1..2] of int: headways ;
array [ROUTES] of int: max_hdwy =
    [max([headways[r,t,2]|t in 1..(trips[r]-1)]) | r in ROUTES] ;
array [ROUTES] of int:last_train =
    [max([j|j in 1..pt where ttime[r_sct[r],j]<Ntime]) | r in ROUTES] ;
array [ROUTES] of int:last_train_time =
    [ttime[r_sct[r],last_train[r]]| r in ROUTES] ;
array [ROUTES] of int: max_train_gap =
    [max([ttime[r_sct[r],j+1]-ttime[r_sct[r],j] | j in 1..(last_train[r]-1)]) | r in ROUTES] ;

```

```
bool: LAY = true;         % true= with layover; false= w/o layover
array [ROUTES] of int:lay = [ if LAY then layover[r] else 0 endif | r in
     ROUTES] ;

%---------------------------------
% Decision Variables
%---------------------------------
int: max_coord= max([RT_Coord[r]+walk[r]|r in ROUTES]) ;
int: min_coord= min([RT_Coord[r]+walk[r]|r in ROUTES]) ;
array[ROUTES] of int: earliest_start =
 [1-(RT_Coord[r]+walk[r]+(headways[r,1,1]+headways[r,1,2]) div 2)
  | r in ROUTES] ;
int: minstart = min(earliest_start) ;
array [ROUTES,TRIPS] of var minstart..Ntime-min_coord: BStart ;

% On each matching route and station, which trip meets which train
array [ROUTES,TRIPS] of var 1..pt: trainmeets ;

% The flow variables used to represent the sequence of trips performed
    by each bus
int: final_trip = max(trips)+1;
array [ROUTES,0..max(trips),ROUTES,1..final_trip] of var 0..1: flow;

%---------------------------------
% Constraints
%---------------------------------
% Set to 0 all impossible flows
% Flows from r,0 where r>1
constraint forall(r1,r2 in ROUTES,t2 in 1..final_trip where r1>1)
    (flow[r1,0,r2,t2]=0);

% Flows to non-existent bus trips
constraint forall (r1,r2 in ROUTES, t1,t2 in TRIPS where t2>trips[r2])
    (flow[r1,t1,r2,t2]=0) ;

% Flows from non-existent bus trips
constraint forall (r1,r2 in ROUTES, t1,t2 in TRIPS where t1>trips[r1])
    (flow[r1,t1,r2,t2]=0) ;

% Flows from a trip to itself  (this is a redundant constraint)
constraint forall(r in ROUTES, t in TRIPS)(flow[r,t,r,t]=0);
constraint forall(r in ROUTES, t1, t2 in 1..trips[r] where t2<=t1)
    (flow[r,t1,r,t2]=0) ;

% Flow constraints
constraint forall (r1 in ROUTES, t1 in 1..trips[r1])
    (sum([flow[r1,t1,r2,t2]|r2 in ROUTES, t2 in 1..final_trip])=1) ;
constraint forall(r2 in ROUTES, t2 in 1..trips[r2])
    (sum([flow[r1,t1,r2,t2] | r1 in ROUTES, t1 in 0..trips[r1]]) = 1) ;

% Bus fleet size constraint
```

```
constraint sum([flow[1,0,r2,t2]| r2 in ROUTES, t2 in 1..trips[r2]]) <= bct ;

% Bus arrival times
constraint forall(r in ROUTES, t in TRIPS where t>trips[r])(BStart[r,t]==Ntime-max_coord) ;

%  If the same bus does two trips, then there must be time (RT+lay)
% to complete the first trip and start the next
constraint   forall(r1,r2 in ROUTES, t1 in 1..trips[r1], t2 in 1..trips[r2])(
 (flow[r1,t1,r2,t2]=1 ->
    BStart[r2,t2] >= BStart[r1,t1] + RT_data[r1,r2] + lay[r2])
  /\
 (flow[r1,t1,r2,t2]=1 ->
    BStart[r2,t2] <= BStart[r1,t1] + RT_data[r1,r2] + lay[r2] + 15)) ;
constraint forall(r in ROUTES, t1, t2 in 1..trips[r] where t2<=t1)
       (flow[r,t1,r,t2]=0) ;

% Minimum and Maximum headway constraints:
% The headway between successive bus trips on route has a minimum
% and maximum range. For each route, trip wise headway is considered
% separately as each route has different target headways and transition points
constraint forall(r in ROUTES, t in 1..trips[r]-1)
    (BStart[r,t+1] >= BStart[r,t] + headways[r,t,1] /\
     BStart[r,t+1] <= BStart[r,t] + headways[r,t,2]);

array [ROUTES,TRIPS] of int: headways_mid =
    array2d(ROUTES,TRIPS, [(headways[r,t,1]+headways[r,t,2]) div 2
    |r in ROUTES,t in TRIPS]);

function var int: headwaydiff(ROUTES:r, TRIPS:t) =
    abs((BStart[r,t+1]-BStart[r,t])-headways_mid[r,t]) ;

% First bus problem:
% The first bus trip on route r must arrive at a time between
% the beginning of time horizon and a certain maximum headway
% on that route since the beginning.

% First bus arrives within max headway of T0
constraint forall(r in ROUTES)
    (BStart[r,1]+RT_Coord[r] <= headways[r,1,2])  ;
constraint forall(r in ROUTES,t in 2..trips[r])
    (BStart[r,t]  > ttime[r_sct[r],t-1]-RT_Coord[r]) ;

% Last bus arrives within max headway of tmax
constraint forall(r in ROUTES)
    (BStart[r,trips[r]]+RT_Coord[r] >= tmax - headways[r,trips[r]-1,2]);
constraint forall(r in ROUTES,t in 0..(trips[r]-1))
    (BStart[r,trips[r]-t] <= ttime[r_sct[r],last_train[r]-t] -
    (RT_Coord[r]+walk[r])) ;
```

```

% Earlier trips on a route must meet earlier trains at the station
constraint forall(r in ROUTES, t in 1..trips[r]-1)
    (trainmeets[r,t]<trainmeets[r,t+1]) ;
% The "dummy" train is pt
constraint forall(r in ROUTES, t in TRIPS where t>trips[r])
    (trainmeets[r,t]=pt) ;

% Given decisions about which trips meet which trains, the program
% can infer which trains are met. Each trip must arrive in time to
% meet its train - leaving enough time to walk.
constraint forall (r in ROUTES, t in 1..trips[r])
    (let {var int: k = trainmeets[r,t]} in
    ttime[r_sct[r],k]  >= BStart[r,t] + RT_Coord[r]+walk[r]
    /\
    (k>1 -> ttime[r_sct[r],k-1]  < BStart[r,t] + RT_Coord[r]+walk[r]));

array [ROUTES,1..pt] of HOURS: trainhour  =
    array2d(ROUTES,1..pt,
    [max([h * bool2int(ttime[r_sct[r],k] > hours[h]) | h in HOURS])|r in
     ROUTES, k in 1..pt]);

array[ROUTES, 1..pt] of int: onetrainpass =
    array2d(ROUTES,1..pt,
    [let {int:ht = trainhour[r,1]} in
     if j=1 then Pass_data[ht,r]*(ttime[r_sct[r],1]-hours[ht])
               + sum(h in HOURS where h  < ht )(60*Pass_data[h,r])
     else let {int: hpt = trainhour[r,j-1]} in
        (Pass_data[hpt,r]*(hours[hpt]+60-ttime[r_sct[r],j-1])
       - Pass_data[ht,r]*(hours[ht]+60-ttime[r_sct[r],j]) )
       + sum(h in HOURS where h>hpt /\ h  <= ht )(60*Pass_data[h,r])
     endif
    | r in ROUTES, j in 1..pt ]);

array[ROUTES, TRIPS] of var int: trainpass =
array2d(ROUTES,TRIPS,
    [if t>trips[r] then 0
     elseif t=1 then let {var 1..pt:train = trainmeets[r,1],
            var HOURS:ht = trainhour[r,train]} in
            (Pass_data[ht,r]*(ttime[r_sct[r],train]-hours[ht])
          + sum(h in HOURS where h  < ht )(60*Pass_data[h,r]))
     else let
        {var 1..pt:prevtrain =  trainmeets[r,t-1],
         var 1..pt:train = trainmeets[r,t],
         var HOURS:hpt=trainhour[r,prevtrain],
         var HOURS:ht=trainhour[r,train]} in
            ((Pass_data[hpt,r]*(hours[hpt+1]-ttime[r_sct[r],prevtrain])
            - Pass_data[ht,r]*(hours[ht+1]-ttime[r_sct[r],train]) )
            + sum(h in HOURS where h>hpt /\ h  <= ht )(60*Pass_data[h,r
    ]))
       endif  | r in ROUTES, t in TRIPS]);

```

```

%--------------------------------
% Stage 1 Objective(s) (z1 to z4)
%--------------------------------
% Maximise the number of passengers on trains that are met (z1)
var int: obj = sum([onetrainpass[r,trainmeets[r,t]]
                | r in ROUTES, t in 1..trips[r]]);

% Minimise the maximum number of passengers taking a train (z2)
var int: obj = max([trainpass[r,t]|r in ROUTES, t in 1..trips[r]])  ;

% Minimize variation from target headways (z3)
var int: obj = sum(r in ROUTES,t in 1..trips[r]-1)(headwaydiff(r,t)) ;

% Minimize total passenger waiting time (z4)
var 0..sum([max_hdwy[r]*trips[r]|r in ROUTES]): obj;
array [ROUTES,TRIPS] of var 0..max(max_hdwy): waitBT  =
    array2d(ROUTES,TRIPS,
        [ttime[r_sct[r],trainmeets[r,t]]-(BStart[r,t] + RT_Coord[r]+walk
    [r])
        | r in ROUTES, t in TRIPS]) ;

constraint obj = sum(r in ROUTES,t in 1..trips[r])(waitBT[r,t]) ;
solve minimize obj;

%--------------------------------
% Output
%--------------------------------
["{\n"] ++
["\"Trainmeets\" : ["] ++
[show([fix(trainmeets[r,t]) | t in TRIPS]) ++
    if r==max(ROUTES) then "" else "," endif
    | r in ROUTES] ++ ["],\n"] ++
["\"BStart\": "]++ [show([BStart[r,t]|r in ROUTES, t in 1..trips[r]]) ++
     ",\n"] ++
["\"Flow\": "]++
    [show([flow[r1,t1,r2,t2] | r1 in ROUTES, t1 in 0..max(trips),
    r2 in ROUTES, t2 in 1..final_trip]) ++ ",\n"] ++
["\"Obj\" : \(obj)"  ] ++
["\n}"];
```

A.1.3 Stage-2: Full BTCP Objective

```
include "globals.mzn";
%-----------------------------------------------------------%
% Case: City of Wyndham, weekday transfers from 7am to 3pm
%-----------------------------------------------------------%

% INPUT DATA
int: bct;           % bus fleet size
```

```
int: sct;           % total number of stations
int: tct;           % total number of train departures
int: tlast;         % last train departure time
int: route;         % bus routes
int: pt;            % number of physical trains
int: tmax;          % maximum schedule horizon
int: tothours;      % total number of hours in the schedule horizon

% Null parameters
int: Nroute= route+1;
int: Ntct= tct+1;
int: Ntime = tmax + 1000;

% SETS
set of int: TRAINS = 1..Ntct;               % set of all physical trains
set of int: LOCATIONS = 1..Nroute;          % set of all locations
set of int: MAXTIME = 0..Ntime+max(walk); % set of all times
set of int: ROUTES = 1..route;              % set of all routes
set of int: TRIPS = 1..max(trips);          % set of all bus trips
set of int: BUSES = 1..bct;                 % set of all buses
set of int: HOURS = 1..tothours;            % set of all hours

% INPUT ARRAYS
array [ROUTES] of int: trips ;              % bus trips on each route
array [1..sct, 1..pt] of MAXTIME: ttime;   % train departure times
array [ROUTES] of int: r_sct;               % coordinating station
array [ROUTES] of int: walk;                % walk time to the station
array [ROUTES] of int: layover;             % routewise layover time
array [HOURS] of int: hours;                % consecutive hours
%  array[int] of int: trainmeets_ws;       % train meets from Stage 1

% Runtime
array [1..route, 1..route] of int: RT_data;
array [ROUTES] of int: RT_Coord;

% Bus to train passengers
array [HOURS, ROUTES] of int: Pass_data;
array [ROUTES, TRIPS,1..2] of int: headways;
array [ROUTES] of int:last_train =
      [max([j|j in 1..pt where ttime[r_sct[r],j]<Ntime]) | r in ROUTES];
array [ROUTES] of int:last_train_time =
      [ttime[r_sct[r],last_train[r]]| r in ROUTES] ;
array [ROUTES] of int: max_hdwy =
      [max([headways[r,t,2]|t in 1..(trips[r]-1)]) | r in ROUTES];

% Feeding train meets from Stage-1
array [ROUTES,TRIPS] of 1..pt: trainmeets =
array2d(ROUTES,TRIPS, trainmeets_ws ) ;

% latest bus arrival (trip) time to meet the trains chosen above
array [ROUTES, TRIPS] of 0..Ntime : ttime2 =
    array2d(ROUTES,TRIPS,
```

```
    [ttime[r_sct[r],trainmeets[r,t]]|r in ROUTES, t in TRIPS]);


%--------------------------------
% Decision Variables
%--------------------------------
array[ROUTES] of int: earliest_start =
       [1-(RT_Coord[r]+walk[r]+(headways[r,1,1]+headways[r,1,2]) div 2)
       | r in ROUTES] ;
int: minstart = min(earliest_start) ;
int: max_coord= max([RT_Coord[r]+walk[r]|r in ROUTES]);

array [ROUTES,TRIPS] of var minstart..Ntime-max_coord: BStart;
array [ROUTES] of int:sumPass = [60*sum([Pass_data[h,r]|h in HOURS])| r
    in ROUTES] ;
array [ROUTES,TRIPS] of var 0..max(sumPass) : buspass;
array [ROUTES, TRIPS] of var 0..max(max_hdwy): waitBT;

% The hour during which bus route r, trip t, arrives at the station
array [ROUTES,TRIPS] of var HOURS: bushour ;

% The flow variables used to represent the sequence
% of trips performed by each bus
int: final_trip = max(trips)+1;
array [ROUTES,0..max(trips),ROUTES,1..final_trip] of var 0..1: flow;


%--------------------------------
% Constraints
%--------------------------------
% Set to 0 all impossible flows
% Flows from r,0 where r>1
constraint forall(r1,r2 in ROUTES,t2 in 1..final_trip where r1>1)
    (flow[r1,0,r2,t2]=0);

% Flows to non-existent bus trips
constraint forall (r1,r2 in ROUTES, t1,t2 in TRIPS where t2>trips[r2])
    (flow[r1,t1,r2,t2]=0) ;

% Flows from non-existent bus trips
constraint forall (r1,r2 in ROUTES, t1,t2 in TRIPS where t1>trips[r1])
    (flow[r1,t1,r2,t2]=0) ;

% Flows from a trip to itself  (this is a redundant constraint)
constraint forall(r in ROUTES, t in TRIPS)(flow[r,t,r,t]=0);
constraint forall(r in ROUTES, t1, t2 in 1..trips[r] where t2<=t1)
    (flow[r,t1,r,t2]=0) ;

% Flow constraints
constraint forall (r1 in ROUTES, t1 in 1..trips[r1])
    (sum([flow[r1,t1,r2,t2]|r2 in ROUTES, t2 in 1..final_trip])=1) ;
constraint forall(r2 in ROUTES, t2 in 1..trips[r2])
```

```
    (sum([flow[r1,t1,r2,t2] | r1 in ROUTES, t1 in 0..trips[r1]]) = 1) ;

% Bus fleet size constraint
constraint sum([flow[1,0,r2,t2]| r2 in ROUTES, t2 in 1..trips[r2]]) <= bct ;

% Bus arrival times
constraint forall(r in ROUTES, t in TRIPS where t>trips[r])
    (BStart[r,t]==Ntime-max_coord) ;

bool: LAY = true;   % true= with layover; false= w/o layover
array [ROUTES] of int:lay = [ if LAY then layover[r] else 0 endif | r in ROUTES] ;

% If the same bus does two trips, then there must be time (RT+lay)
% to complete the first trip and start the next
constraint  forall(r1,r2 in ROUTES, t1 in 1..trips[r1], t2 in 1..trips[r2])(
    (flow[r1,t1,r2,t2]=1 ->
        BStart[r2,t2] >= BStart[r1,t1] + RT_data[r1,r2] + lay[r2])
    /\
    (flow[r1,t1,r2,t2]=1 ->
        BStart[r2,t2] <= BStart[r1,t1] + RT_data[r1,r2] + lay[r2] + 15)) ;

constraint forall(r in ROUTES, t1, t2 in 1..trips[r] where t2<=t1)
       (flow[r,t1,r,t2]=0) ;

% Minimum and Maximum headway constraints:
% The headway between successive bus trips on route has a minimum
% and maximum range. For each route, trip wise headway is considered
% separately as each route has different target headways and transition points
constraint forall(r in ROUTES, t in 1..trips[r]-1)
    (BStart[r,t+1] >= BStart[r,t] + headways[r,t,1] /\
     BStart[r,t+1] <= BStart[r,t] + headways[r,t,2]);


% First bus problem:
% The first bus trip on route r must arrive at a time between
% the beginning of time horizon and a certain maximum headway
% on that route since the beginning.

% First bus arrives within max headway of time 0
constraint  forall(r in ROUTES)
    (BStart[r,1] <= headways[r,1,2] - RT_Coord[r]);
constraint  forall(r in ROUTES,t in 2..trips[r])
    (BStart[r,t]  > ttime[r_sct[r],t-1]-RT_Coord[r]);

% Last bus arrives within max headway of tmax
constraint forall(r in ROUTES)
```

```
    (BStart[r,trips[r]] >= tmax - (headways[r,trips[r]-1,2]+RT_Coord[r]));
constraint forall(r in ROUTES)    % Last bus arrives before the last train
    (BStart[r,trips[r]]+RT_Coord[r]  <= last_train_time[r]) ;
constraint forall(r in ROUTES,t in 1..(trips[r]-1))
    (BStart[r,trips[r]-t] <= ttime[r_sct[r],last_train[r]-t] -
    (RT_Coord[r]+walk[r])) ;

% Earlier trips on a route must meet earlier trains at the station
constraint forall(r in ROUTES, t in 1..trips[r]-1)
    (trainmeets[r,t]<trainmeets[r,t+1]) ;
% The "dummy" train is pt
constraint forall(r in ROUTES, t in TRIPS where t>trips[r])
    (trainmeets[r,t]=pt) ;

constraint  forall (r in ROUTES, t in 1..trips[r])
(let {var int: k = trainmeets[r,t]} in
    ttime[r_sct[r],k]   >= BStart[r,t] + RT_Coord[r]+walk[r]
    /\ % Each bus meets a different train
    (k>1 -> ttime[r_sct[r],k-1] < BStart[r,t] + RT_Coord[r]+walk[r]));

% Compute the hour within which each bus arrives
constraint forall(r in ROUTES, t in TRIPS)
    (bushour[r,t] =
    max([h * bool2int((BStart[r,t]+RT_Coord[r]) > hours[h]) | h in HOURS]));

constraint forall(r in ROUTES,t in TRIPS)(buspass[r,t]>=0) ;
constraint forall(r in ROUTES, t in 1..trips[r])(
    buspass[r,t] =
     let {var HOURS:ht1 = bushour[r,t] } in
          if  t=1 then
          (Pass_data[ht1,r]* ((BStart[r,1]+RT_Coord[r])-hours[ht1])
         + sum(hs in HOURS)(bool2int(hs<ht1)*Pass_data[hs,r]*60))
          elseif t>trips[r] then 0
          else
          let {var HOURS:ht0 =  bushour[r,t-1] } in
          ((Pass_data[ht0,r]*(hours[ht0+1]- (BStart[r,t-1]+RT_Coord[r]))
          - Pass_data[ht1,r]*(hours[ht1+1]- (BStart[r,t]+RT_Coord[r])))
          + sum(hs in HOURS)(bool2int(hs > ht0 /\ hs  <= ht1)*60*Pass_data[hs,r]))
          endif
          );

constraint forall (r in ROUTES, t in 1..trips[r])
    (waitBT[r,t] >=
        if t>trips[r] then 0
        else  ttime2[r,t]-(BStart[r,t] + RT_Coord[r]+walk[r])
        endif );

%--------------------------------
```

```
% Stage 2 BTCP Objective (Z)
%---------------------------------
% Maximise the number of passengers on trains that are met
var 0..sum([sumPass[r]*max_hdwy[r]|r in ROUTES]): obj;
constraint obj =   sum(r in ROUTES,t in 1..trips[r])
    (waitBT[r,t]*buspass[r,t]);

solve minimize obj;

output
["{\n"] ++
["\"NRoutes\": \(route),\n"] ++
["\"NTrips\": \(max(trips)),\n"] ++
["\"Trainmeets\": "]++ [show(trainmeets) ++ ",\n"] ++
["\"BStart\": "]++ [show(BStart) ++ ",\n"] ++
["\"Buspass\": "]++ [show(buspass) ++ ",\n"] ++
["\"WaitBT\": "]++ [show([ttime2[r,t]-(BStart[r,t] + RT_Coord[r]+walk[r
   ])
                   |t in TRIPS])    ++"|" ++ " \n" | r in ROUTES] ++
["\"Obj\" : \(Obj2)"  ] ++
["\n}"];
```

A.2 I-TTVS model

This section presents the *Integrated Timetabling and Vehicle Scheduling* approach (abbreviated as I-TTVS) modelled in *MiniZinc*.

```
include "globals.mzn";
%-----------------------------------------------------------%
% Case: City of Wyndham, weekday transfers from 7am to 3pm
%-----------------------------------------------------------%
% INPUT DATA
int: bct;              % bus fleet size
int: sct;              % total number of stations
int: tct;              % total number of train departures
int: tlast;            % last train departure time
int: route;            % bus routes
int: pt;               % number of physical trains
int: tmax;             % maximum schedule horizon
int: tothours;         % total number of hours

% Null parameters
int: Ntime = tmax + 1000;

% SETS
set of int: TRAINS = 1..Ntct;                % set of all physical trains
set of int: LOCATIONS = 1..Nroute;           % set of all locations
set of int: MAXTIME = 0..Ntime+max(walk); % set of all times
set of int: ROUTES = 1..route;               % set of all routes
```

```
set of int: TRIPS = 1..max(trips);          % set of all bus trips
set of int: BUSES = 1..bct;                 % set of all buses
set of int: HOURS = 1..tothours;            % set of all hours

% INPUT ARRAYS
array [ROUTES] of int: trips ;              % bus trips on each route
array [1..sct, 1..pt] of MAXTIME: ttime;   % train departure times
set of int: TTIME = {ttime[s,k] | s in 1..sct,k in 1..pt where ttime[s,k
    ]<Ntime} ;
array [ROUTES] of int: r_sct;               % coordinating station
array [ROUTES] of int: walk;                % walk time to the station
array [ROUTES] of int: layover;             % routewise layover time
array [HOURS] of int: hours;                % consecutive hours

%---------LNS inputs
array[int] of int: trainmeets_ws;
array[int] of int: bstart_ws;
array[int] of int: buspass_ws;
array[int] of int: waitbt_ws;
array[int] of int: flow_ws;

% Runtime
array [1..route, 1..route] of int: RT_data;
array [ROUTES] of int: RT_Coord ;

% Bus train passengers
array [HOURS, ROUTES] of int: Pass_data; % bus to train volume
array [ROUTES, TRIPS,1..2] of int: headways ;
array [ROUTES] of int:last_train =
      [max([j|j in 1..pt where ttime[r_sct[r],j]<Ntime]) | r in ROUTES]
    ;
array [ROUTES] of int:last_train_time =
      [ttime[r_sct[r],last_train[r]]| r in ROUTES] ;
array [ROUTES] of int: max_hdwy =
      [max([headways[r,t,2]|t in 1..(trips[r]-1)]) | r in ROUTES] ;

%--------------------------------
% Decision Variables
%--------------------------------
int: max_coord= max([RT_Coord[r]+walk[r]|r in ROUTES]) ;
int: min_coord= min([RT_Coord[r]+walk[r]|r in ROUTES]) ;
array[ROUTES] of int: earliest_start =
    [1-(RT_Coord[r]+walk[r]+(headways[r,1,1]+headways[r,1,2]) div 2)
    | r in ROUTES] ;
int: minstart = min(earliest_start) ;
array [ROUTES,TRIPS] of var minstart..Ntime-max_coord: BStart ;

array [ROUTES] of int:sumPass = [60*sum([Pass_data[h,r]
    |h in HOURS])| r in ROUTES] ;
array [ROUTES,TRIPS] of var 0..max(sumPass):buspass;
array [ROUTES, TRIPS] of var 0..max(max_hdwy):waitBT;
```

```
% On each matching route and station, which trip meets which train
array [ROUTES,TRIPS] of var 1..pt: trainmeets ;

% The time the train met by bus route r, trip t departs
array [ROUTES, TRIPS] of var 0..Ntime : ttime2 =
    array2d(ROUTES,TRIPS,
    [ttime[r_sct[r],trainmeets[r,t]]|r in ROUTES, t in TRIPS]) ;

% The hour during which bus route r, trip t, arrives at the station
array [ROUTES,TRIPS] of var HOURS: bushour ;

% The flow variables used to represent the sequence of trips performed by each bus
int: final_trip = max(trips)+1;
array [ROUTES,0..max(trips),ROUTES,1..final_trip] of var 0..1: flow;


%--------------------------------
% Constraints
%--------------------------------
% Set to 0 all impossible flows
% Flows from r,0 where r>1
constraint forall(r1,r2 in ROUTES,t2 in 1..final_trip where r1>1)
    (flow[r1,0,r2,t2]=0);

% Flows to non-existent bus trips
constraint forall (r1,r2 in ROUTES, t1,t2 in TRIPS where t2>trips[r2])
    (flow[r1,t1,r2,t2]=0) ;

% Flows from non-existent bus trips
constraint forall (r1,r2 in ROUTES, t1,t2 in TRIPS where t1>trips[r1])
    (flow[r1,t1,r2,t2]=0) ;

% Flows from a trip to itself  (this is a redundant constraint)
constraint forall(r in ROUTES, t in TRIPS)(flow[r,t,r,t]=0);
constraint forall(r in ROUTES, t1, t2 in 1..trips[r] where t2<=t1)
    (flow[r,t1,r,t2]=0) ;

% Flow constraints
constraint forall (r1 in ROUTES, t1 in 1..trips[r1])
    (sum([flow[r1,t1,r2,t2]|r2 in ROUTES, t2 in 1..final_trip])=1) ;
constraint forall(r2 in ROUTES, t2 in 1..trips[r2])
    (sum([flow[r1,t1,r2,t2] | r1 in ROUTES, t1 in 0..trips[r1]]) = 1) ;

% Bus fleet size constraint
constraint sum([flow[1,0,r2,t2]| r2 in ROUTES, t2 in 1..trips[r2]]) <= bct ;

% Bus arrival times
constraint forall(r in ROUTES, t in TRIPS where t>trips[r])
    (BStart[r,t]==Ntime-max_coord) ;

```

```
bool: LAY = true;          % true= with layover; false= w/o layover
array [ROUTES] of int:lay = [ if LAY then layover[r] else 0 endif | r in
     ROUTES] ;

% If the same bus does two trips, then there must be time (RT+lay)
% to complete the first trip and start the next
constraint   forall(r1,r2 in ROUTES, t1 in 1..trips[r1], t2 in 1..trips[
    r2])(
    (flow[r1,t1,r2,t2]=1 ->
        BStart[r2,t2] >= BStart[r1,t1] + RT_data[r1,r2] + lay[r2])
    /\
    (flow[r1,t1,r2,t2]=1 ->
        BStart[r2,t2] <= BStart[r1,t1] + RT_data[r1,r2] + lay[r2] + 15))
     ;

constraint forall(r in ROUTES, t1, t2 in 1..trips[r] where t2<=t1)
    (flow[r,t1,r,t2]=0) ;

% Minimum and Maximum headway constraints:
% The headway between successive bus trips on route has a minimum
% and maximum range. For each route, trip wise headway is considered
% separately as each route has different target headways and transition
    points
constraint forall(r in ROUTES, t in 1..trips[r]-1)
    (BStart[r,t+1] >= BStart[r,t] + headways[r,t,1] /\
     BStart[r,t+1] <= BStart[r,t] + headways[r,t,2]);

% First bus problem:
% The first bus trip on route r must arrive at a time between
% the beginning of time horizon and a certain maximum headway
% on that route since the beginning.

% First bus arrives within max headway of time T0
constraint  forall(r in ROUTES)
    (BStart[r,1] <= headways[r,1,2] - RT_Coord[r]);
constraint  forall(r in ROUTES,t in 2..trips[r])
    (BStart[r,t]  > ttime[r_sct[r],t-1]-RT_Coord[r]);

% Last bus arrives within max headway of tmax
constraint forall(r in ROUTES)
    (BStart[r,trips[r]] >= tmax - (headways[r,trips[r]-1,2]+RT_Coord[r])
    );
constraint forall(r in ROUTES)    % Last bus arrives before the last
    train
    (BStart[r,trips[r]]+RT_Coord[r]  <= last_train_time[r]);
constraint forall(r in ROUTES,t in 1..(trips[r]-1))
    (BStart[r,trips[r]-t] <= ttime[r_sct[r],last_train[r]-t] -
                             (RT_Coord[r]+walk[r])) ;

% Earlier trips on a route must meet earlier trains at the station
constraint forall(r in ROUTES, t in 1..trips[r]-1)
    (trainmeets[r,t]<trainmeets[r,t+1]) ;
```

```
% The "dummy" train is pt
constraint forall(r in ROUTES, t in TRIPS where t>trips[r])
    (trainmeets[r,t]=pt) ;

% Actual trips must meet actual trains
% Each trip must arrive in time to meet its train - leaving enough time to walk
constraint forall (r in ROUTES, t in 1..trips[r])
    (let {var int: k = trainmeets[r,t]} in
        ttime[r_sct[r],k]  >= BStart[r,t] + RT_Coord[r]+walk[r]
        /\ % Each bus meets a different train
        (k>1 -> ttime[r_sct[r],k-1]  < BStart[r,t] + RT_Coord[r]+walk[r]));

% Compute the hour within which each bus arrives
constraint forall(r in ROUTES, t in TRIPS)
    (bushour[r,t] =
    max([h * bool2int((BStart[r,t]+RT_Coord[r]) > hours[h]) | h in HOURS]));

constraint forall(r in ROUTES,t in TRIPS)(buspass[r,t]>=0) ;
constraint forall(r in ROUTES, t in 1..trips[r])(
    buspass[r,t] =
        let { var HOURS:ht1 = bushour[r,t] } in
        if  t=1 then
        (Pass_data[ht1,r]* ((BStart[r,1]+RT_Coord[r])-hours[ht1])
                + sum(hs in HOURS)(bool2int(hs<ht1)*Pass_data[hs,r]*60))
        elseif t>trips[r] then 0
        else let { var HOURS:ht0 =  bushour[r,t-1] } in
        ((Pass_data[ht0,r]*(hours[ht0+1]- (BStart[r,t-1]+RT_Coord[r]))
        - Pass_data[ht1,r]*(hours[ht1+1]- (BStart[r,t]+RT_Coord[r])))
        + sum(hs in HOURS)(bool2int( hs > ht0 /\ hs  <= ht1) *60* Pass_data[hs,r]))
        endif) ;

constraint forall (r in ROUTES, t in 1..trips[r])
      (waitBT[r,t] =
        if t>trips[r] then 0
        else   ttime2[r,t]-(BStart[r,t] + RT_Coord[r]+walk[r])
        endif);

%---------------------------------
% Large Neighbourhood Search
%---------------------------------
int: fLNSEnabled;
int: fFixedHeadway;
int: relR;
int: relT;
int: devR;
int: devT;
array[ROUTES,TRIPS] of int: bstart_ws_2d = array2d(ROUTES,TRIPS, bstart_ws);
```

```
array [ROUTES,0..max(trips),ROUTES,1..final_trip] of var 0..1:
    flow_ws_4d =
    array4d(ROUTES, 0..max(trips), ROUTES, 1..final_trip, flow_ws);

constraint if fLNSEnabled==1 then
    if fFixedHeadway>0 then
        forall(r in ROUTES diff relR-devR..relR+devR)
        (BStart[r,1] = bstart_ws_2d[r,1])
    else
        forall(r in ROUTES diff relR-devR..relR+devR, t in TRIPS diff
    relT-devT..relT+devT)(BStart[r,t] = bstart_ws_2d[r,t])
    endif
  /\
    forall(r1 in ROUTES diff relR-devR..relR+devR, t1 in TRIPS diff relT
    -devT..relT+devT)
    (forall(r2 in ROUTES, t2 in TRIPS)
    (flow[r1,t1,r2,t2] = flow_ws_4d[r1,t1,r2,t2]))
    else true
    endif;

%--------------------------------
% Objective
%--------------------------------
% Maximise the number of passengers on trains that are met
var 0..sum([max_hdwy[r]*sumPass[r]|r in ROUTES]): obj;
constraint obj =   sum(r in ROUTES,t in 1..trips[r])
    (waitBT[r,t]*buspass[r,t])  ;

solve
::seq_search(
[ if length(trainmeets_ws)>0 then warm_start([trainmeets[r,t]
    | r in ROUTES, t in TRIPS], trainmeets_ws)
  else constraint_name("dummy") endif,
  if length(bstart_ws)>0 then warm_start([BStart[r,t]
    | r in ROUTES, t in TRIPS], bstart_ws)
  else constraint_name("dummy") endif,
  if length(buspass_ws)>0 then warm_start([buspass[r,t]
    | r in ROUTES, t in TRIPS], buspass_ws)
  else constraint_name("dummy") endif,
  if length(waitbt_ws)>0 then warm_start([waitBT[r,t]
    | r in ROUTES, t in TRIPS], waitbt_ws)
  else constraint_name("dummy") endif,
  if length(flow_ws)>0 then warm_start([flow[r1,t1,r2,t2]
    | r1 in ROUTES, t1 in 0..max(trips), r2 in ROUTES, t2 in 1..
    final_trip], flow_ws)
  else constraint_name("dummy") endif,
])
minimize obj;

%--------------------------------
% Output
%--------------------------------
```

```
["{\n"] ++
["\"Trainmeets\": "]++ [show(trainmeets) ++ ",\n"] ++
["\"BStart\": "]++ [show(BStart) ++ ",\n"] ++
["\"Buspass\": "]++ [show(buspass) ++ ",\n"] ++
["\"WaitBT\": "]++ [show(waitBT) ++ ",\n"] ++
["\"Flow\": "]++ [show([flow[r1,t1,r2,t2] | r1 in ROUTES, t1 in 0..max(
    trips), r2 in ROUTES, t2 in 1..final_trip]) ++ ",\n"] ++
["\"Obj\" : \(obj)"  ] ++
["\n}"];
```

A.3 Model Independent Datafile

This section presents a simplified input data file created in *MiniZinc* for the case study area: *City of Wyndham*. The data is setup in such a way that it is usable across the three CP models proposed in this research.

```
%-------------LARGE NETWORK INSTANCE-------------------------%
% 24 Bus Routes/ 5 Transfer Stations
% Routes: 150 150A 151 151A 153 153A 160 160A 161 161A 166 166A
%         167 167A 170 170A 180 180A 190 190A 191 191A 192 192A
% Stations: Tarneit, Williams Landing, Hoppers Crossing,
%           Werribee, Wyndham Vale
%------------------------------------------------------------%
bct =25;         % input number of buses
sct= 5;          % total number of actual locations/stations
tmax= 480;       % schedule horizon: 7am = 0 mins, 3pm = 480 mins
route = 24;      % number of bus routes
tothours = 10;   % total hours
hours = [-60,0,60,120,180,240,300,360,420,480];

%---------------TRAIN DATA
pt= 32;          % number of physical trains
tct= 146;        % number of train departures
% last train
tlast = max([ttime[i,j] |i in 1..sct, j in 1..pt
            where ttime[i,j]<Ntime]);

ttime=           % train departure times at each station
% Tarneit
[|4,17,32,42,47,60,71,91,109,122,138,157,178,198,218,
    238,258,278,298,318,338,358,378,398,418,438,458,Ntime,Ntime,Ntime,
    Ntime,Ntime |
% Williams Landing
2,11,17,26,36,46,56,68,78,90,100,111,125,142,157,177,
    197,217,237,257,277,297,317,337,357,377,397,417,437,457,477,Ntime |
% Hoppers Crossing
7,13,22,32,42,52,64,74,86,96,107,121,138,153,173,193,
    213,233,253,273,293,313,333,353,373,393,413,433,453,473,Ntime,Ntime |
% Werribee
```

```
4,10,19,29,39,49,61,71,83,93,104,118,135,150,170,190,210,230,250,270,290,310,330,350,370,390,410,430,450,470,Ntime,Ntime |
% Wyndham Vale
11,22,35,41,54,74,85,99,103,114,132,151,172,192,212,232,252,272,292,312,332,352,372,392,412,432,452,477,Ntime,Ntime,Ntime,Ntime |];


%---------------BUS DATA
% coordinating stations  numbers
r_sct= [2,1, 2,1, 3,3, 3,1, 3,4, 3,5, 3,1, 4,1, 4,1, 4,5, 4,5, 4,5];
% route numbers
r_sct_name=["150", "150A", "151", "151A", "153", "153A", "160", "160A", "161", "161A", "166", "166A", "167", "167A", "170", "170A","180", "180A", "190", "190A", "191", "191A", "192", "192A"];
% trips per route (7am to 3pm)
trips=[15,15,12,12,12,12,15,15,12,12,12,12,12,12,24,24,24,24,24,24,10,10,12,12];
layover = [2,2,3,3,1,1,2,2,5,6,3,3,4,4,2,2,2,3,1,1,2,3,2,2];

RT_data =
% runtime between start of a route to the start of the next route (including deadheading)
array2d(1..route,1..route,[
34,19,34,19,31,19,34,27,31,27,39,27,34,27,34,31,34,31,39,31,39,31,39,31,
21,34,21,34,35,34,21,32,35,32,42,32,21,32,21,35,21,35,42,35,42,35,42,35,
40,25,40,25,37,25,40,33,37,33,45,33,40,33,40,37,40,37,45,37,45,37,45,37,
27,40,27,40,41,40,27,38,41,38,48,38,27,38,27,41,27,41,48,41,48,41,48,41,
44,29,44,29,41,29,44,37,41,37,49,37,44,37,44,41,44,41,49,41,49,41,49,41,
44,43,44,43,31,43,44,38,31,38,42,38,44,38,44,31,44,31,42,31,42,31,42,31,
30,24,30,24,25,24,30,18,25,24,31,18,30,18,30,25,30,25,31,25,31,25,31,25,
17,30,17,30,31,30,17,28,31,28,38,28,17,28,17,31,17,31,38,31,38,31,38,31,
66,60,66,60,61,60,66,54,61,54,67,54,66,54,66,61,66,61,67,61,67,61,67,61,
68,67,68,67,55,67,68,62,55,62,66,62,68,62,68,55,68,55,66,55,66,55,66,55,
42,36,42,36,37,36,42,30,37,30,43,30,42,30,42,37,42,37,43,37,43,37,43,37,
53,53,53,53,43,53,53,47,43,47,34,47,53,47,53,43,53,43,34,43,34,43,34,43,
52,46,52,46,47,46,52,40,47,40,53,47,52,40,52,47,52,47,53,47,53,47,53,47,
39,52,39,52,53,52,39,50,53,50,60,50,39,50,39,53,39,53,60,53,60,53,60,53,
34,33,34,33,21,33,34,28,21,28,32,21,34,28,34,21,34,21,32,21,32,21,32,21,
22,35,22,35,36,35,22,33,36,33,43,33,22,33,22,36,22,36,43,36,43,36,43,36,
34,33,34,33,21,33,34,28,21,28,32,21,34,28,34,21,34,21,32,21,32,21,32,21,
25,38,25,38,39,38,25,36,39,36,46,36,25,36,25,39,25,39,46,39,46,39,46,39,
26,25,26,25,13,25,26,20,13,20,24,20,26,20,26,13,26,13,24,13,24,13,24,13,
33,33,33,33,23,33,33,27,23,27,14,27,33,27,33,23,33,23,14,23,14,23,14,23,
36,35,36,35,23,35,36,30,23,30,34,30,36,30,36,23,36,23,34,23,34,23,34,23,
47,47,47,47,37,47,47,41,37,41,28,41,47,41,47,37,47,37,28,37,28,37,28,37,
34,33,40,33,21,33,34,28,21,28,32,28,34,28,34,21,34,21,32,21,32,21,34,21,
41,41,41,41,31,41,41,35,31,35,22,35,41,35,41,31,41,31,22,31,22,31,22,31
]);

% the time between the start of a route to its coordinating station
RT_Coord =
```

```
[19,21,25,27,13,12,18,17,54,55,30,34,40,39,21,22,21,25,13,14,23,28,21,22];

% walk time
walk = [1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,2,1,2,1,3,1,2,1];

%-----------bus headway (example route 150)
headways =
% route 150
array3d(ROUTES,TRIPS, 1..2,
        [ 16,24,        % AM Peak
          16,24,
          16,24,
          16,24,
          16,24,
          16,24,
          20,40,        % Transition to Inter Peak
          20,40,
                32,48,  % Inter Peak
                32,48,
                32,48,
                32,48,
                32,48,
                32,48,
                32,48]); % PM peak starts

Pass_data=  % rate of passenger volume
array2d(HOURS, ROUTES,
[0,0,0,0,0,0,0,0,3,5,2,1,4,6,2,2,0,0,0,0,2,1,4,5
13,4,19,9,1,1,11,7,7,6,2,3,8,22,4,7,11,14,3,6,4,5,6,12
14,4,14,3,0,0,8,4,10,5,5,2,6,9,4,5,10,15,5,4,2,1,6,3
5,2,5,2,0,0,3,2,1,4,2,2,2,4,2,3,3,6,2,1,1,1,2,3
3,1,5,2,0,0,2,2,3,2,1,1,4,5,2,2,4,4,1,1,1,1,3,1
4,1,2,2,0,0,2,0,1,2,1,1,2,3,2,2,2,3,1,1,1,0,1,2
1,1,2,1,0,0,1,1,1,1,1,1,2,4,2,2,2,2,1,1,1,1,2,1
1,1,1,1,0,0,1,1,1,2,1,1,1,2,1,2,1,2,1,1,1,0,1,1
1,1,1,0,0,0,1,0,1,1,2,1,2,2,1,1,2,1,1,1,0,0,1,0
2,1,2,1,0,0,1,2,1,1,2,1,3,1,1,1,1,2,1,1,1,0,1,1]);
%------------------------------------------------------------------%
```

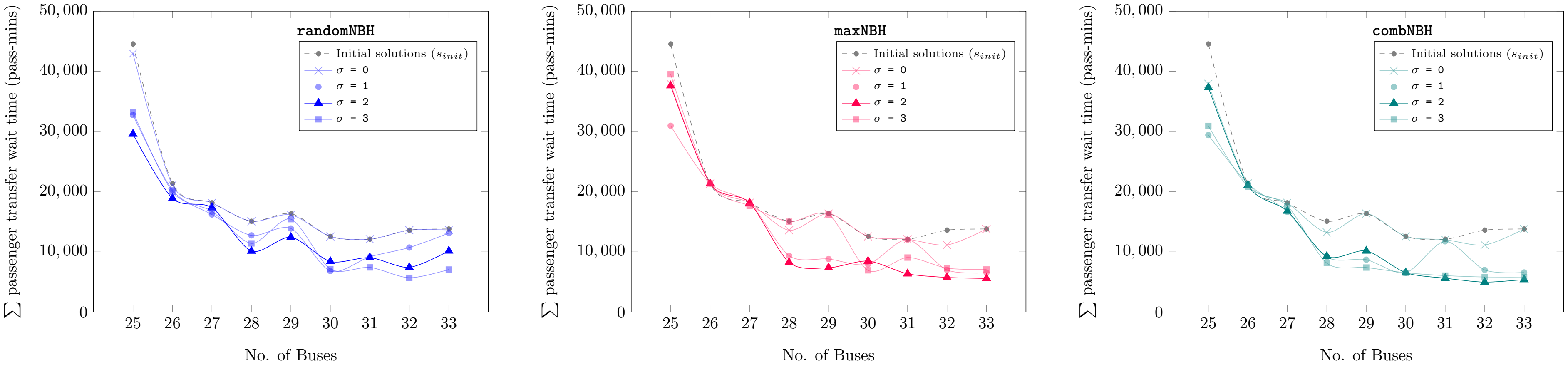
Appendix B

MiniZinc Optimised Results

Table B.1 shows the summary of I-TTVS results when initiated with solutions from SD-TTVS decomposition for the *small* and *large* PT network instances in the City of Wyndham. The solutions generated from the three defined LNS neighbourhoods namely `randomNBH`, `maxNBH` and `combNBH` are presented at a time limit $\delta = 10$ minutes per iteration and against the neighbourhood size σ values ranging from $\{0,1,2,3\}$. Figure B.1 presents a comparison on each neighbourhood performance against different neighbourhood sizes for the *large* network instance.

Network Instance	No. of buses	s_{init}	randomNBH $\sigma = 0$	randomNBH $\sigma = 1$	randomNBH $\sigma = 2$	randomNBH $\sigma = 3$	maxNBH $\sigma = 0$	maxNBH $\sigma = 1$	maxNBH $\sigma = 2$	maxNBH $\sigma = 3$	combNBH $\sigma = 0$	combNBH $\sigma = 1$	combNBH $\sigma = 2$	combNBH $\sigma = 3$
Small (6 X 4)	6							*Infeasible*						
	7	15,175	15,109	11,131	5,398	15,175	15,175	13,969	6,571	6,775	15,109	11,672	6,571	6,143
	8	6,691	6,691	3,595	3,149	3,140	6,577	3,740	5,360	3,284	6,288	3,740	3,716	3,140
	9	5,542	5,542	3,503	2,764	2,764	3,814	3,702	3,324	3,415	3,690	3,023	2,764	2,764
Large (24 X 5)	25	44,544	42,956	32,738	29,578	33,253	37,923	30,954	37,616	39,502	37, 917	29,409	30,936	37,320
	26	21,354	21,102	20,279	18,885	19,982	21,354	21,354	21,354	21,354	21,354	21,354	20,823	21,064
	27	18,144	18,144	16,204	17,336	16,730	18,144	18,144	18,144	17,643	18,144	16,646	17,643	16,784
	28	15,084	15,084	12,738	10,135	11,405	13,590	9,363	8,253	15,015	13,244	9,136	8,134	9,264
	29	16,352	16,157	13,879	12,430	15,429	16,352	8,815	7,365	16,196	16,352	8,693	7,395	10,149
	30	12,562	12,562	6,812	8,404	7,150	12,562	7,935	8,465	6,927	12,562	6,526	6,914	6,539
	31	12,101	12,101	9,178	9,030	7,416	12,101	12,022	6,355	9,052	12,022	11,752	6,055	5,623
	32	13,610	13,610	10,719	7,423	5,698	111,44	6,992	5,574	7,289	11,144	6,992	5,823	4,974
	33	13,793	13,639	13,111	10,156	7,062	13,793	6,531	5,576	7,071	13,793	6,571	5,823	5,391

Table B.1: Summary of objective values (passenger-minutes) obtained from the defined LNS neighbourhoods at σ ={0,1,2,3} and δ = 10 minutes per iteration



(a) randomNBH (b) maxNBH (c) combNBH

Figure B.1: Comparison of solutions (*Large* PT instance) against each of the three defined LNS neighbourhoods

Appendix C

HASTUS *NetPlan* configuration

C.1 Overview

The information regarding *NetPlan* (version.2014) presented in this thesis are obtained through first-hand experience with the software module (including materials prepared for the software packages) and from personal interviews conducted with industry experts. Based on the expert feedback, we understand that trips in HASTUS can be modelled: (i) using fixed (or regular) headways where all the trips move together with a common start time; (ii) freely, that is, individual trip shifting can be enabled for better bus blocking (this can also include a penalty for headway irregularity). The most important observation was that the process of transfer optimisation is then a separate step in the *NetPlan* module after the trips are built.

In this regard, the main research argument of this thesis, that is, "the simultaneous optimisation of timetable coordination and vehicle cost efficiency" was validated. From a planner point of view, the requirement of minimal manual intervention after optimisation was also noted. The evaluation exercise in Section 7.2 using *NetPlan* is based on these inferences.

C.2 Data inputs

NetPlan requires the following input data to build simplified networks and perform trip timetable evaluations:

- Planning Patterns: schematically represents the sequence of planning places (for example, bus stops and train stations) visited by a given route.
- Planning Study: represents the schedule horizon for a given day (for example, 7:00AM to 3:00PM)
- Planning Period: represents the various blocks of time periods in a given planning study (for example AM-peak at 7:00AM to 9:00AM, inter-peak at 9:00AM to 3:00PM etc.) for which trips are built and optimised.
- Run-time version: incorporates a set of scheduled route running times between places within a given planning period.

- Deadhead version: incorporates a set of unproductive service time that a bus takes to travel inter-route without passengers, from one place to the next.
- Layover version: indicates a set of minimum recovery time that a bus spends at a place between consecutive trips.
- Trip Builders: creates trips for all routes in all directions in the given network within the specified planning periods.
- Meet Builders: specifies the criteria for "trip meets", that is, a possible connection between two trips at a transfer place. A minimum, maximum and ideal transfer waiting time between services can be specified.

C.3 *NetPlan* Configuration

In this section, we explain the configuration of *NetPlan* with the Wyndham service specifications such that it is comparable and consistent with the *MiniZinc* modelling criteria.

Network Summary

NetPlan configuration begins with creating a "Connections Diagram", which schematically represents the Wyndham sub-network in scope for this study. Relevant planning places (inclusive of transfer stations) and the bus routes in both directions are modelled between these places with the specified route frequencies.

Table C.1 illustrates the summary of network features as defined in *NetPlan* such as, the route version (denoted as "WynRoutes"), the run-time version ("WynRun"), the deadhead version ("WynDH"), the layover version ("WynLay") and most importantly, the meet builder version ("WynMeet"). These are explained further.

The network inputs concerned with bus scheduling for a set of routes defined under the route version "WynRoutes" are:

- the route running times from origin to destination (or between places/stations) given by "WynRun";
- the deadhead time matrix between places/stations given by "WynDH"; and
- the minimum layover time per route end at 10% of route running time, given by "WynLay".

Summary		
Vehicle Schedule	:	*Wyndham for NetPlan/Weekday*
Versions		
Routes	:	*WynRoutes*
Run times	:	*WynRun*
Deadheads	:	*WynDH*
Layover defaults:	:	*WynLay*
Meet builders	:	*WynMeet*

Table C.1: Summary of network features as defined in *NetPlan*

Trip Builders

Trip-Builders must be defined to generate timetables for a given route, direction and the sequence of places to visit. Trip builders are mainly *Frequency-based* or *Headway-based*. We adopt the latter headway-based criteria where new trips are built based on the desired (fixed) service headway per route, as inferred from the DoT network specifications (Public Transport Victoria, 2014). The run-times used to generate timetables are retrieved from the existing GTFS data within HASTUS.

Meet Builders

The Meet Builder version "WynMeet" is defined to evaluate transfers between built trips that share common places on the network. Criteria for on-trip (covers the dominant service for synchronisation, say bus routes) and related-trip (covers the related trips for synchronisation, say train lines) are specified, with inputs on routes, directions, corresponding meet places and a minimum and maximum connection time range.

Collectively, Table C.2 summarises the bus scheduling and trip meet data inputs required for *NetPlan* evaluation.

WynRoutes					WynMeets		WynRun	WynLay
Bus route	Route direction	Trips per route	AM-Peak headway (mins)	Inter-Peak headway (mins)	Wait time range (mins)	Transfer Station (No.)	Route running time (mins)	Route layover time (mins)
150	In	15	20	40	0-18	wld (2)	19	2
150_A	Out	15	20	40	0-20	tnt (1)	21	2
151	In	12	40	40	0-40	wld (2)	25	3
151_A	Out	12	40	40	0-40	tnt (1)	27	3
153	In	12	40	40	0-40	hcg (3)	29	1
153_A	Out	11	40	40	0-30	hcg (3)	31	1
160	In	15	20	40	0-40	hcg (3)	18	2
160_A	Out	15	20	40	0-40	tnt (1)	17	2
161	In	11	40	40	0-40	hcg (3)	54	5
161_A	Out	11	40	40	0-40	wer (4)	55	6
166	In	11	40	40	0-40	hcg (3)	30	3
166_A	Out	11	40	40	0-60	wvl (5)	34	3
167	In	11	40	40	0-40	hcg (3)	40	4
167_A	Out	12	40	40	0-40	tnt (1)	39	4
170	In	24	20	20	0-40	wer (4)	21	2
170_A	Out	24	20	20	0-40	tnt (1)	22	2
180	In	24	20	20	0-11	wer (4)	21	2
180_A	Out	24	20	20	0-38	tnt (1)	25	3
190	In	24	20	20	0-40	wer (4)	13	1
190_A	Out	24	20	20	0-60	wvl (5)	14	1
191	In	10	40	60	0-40	wer (4)	23	2
191_A	Out	10	40	60	0-60	wvl (5)	28	3
192	In	11	40	40	0-40	wer (4)	21	2
192_A	Out	12	40	40	0-60	wvl (5)	22	2

WynDH: Deadhead time (mins)					
Start/End	*hcg*	*tnt*	*wer*	*wld*	*wvl*
hcg		12	7	8	13
tnt	12		14	15	21
wer	7	14		12	11
wld	8	15	12		20
wvl	13	21	11	20	

Transfer Stations: tnt: *Tarneit (1)*, wld: *Williams Landing (2)*, hcg: *Hoppers Crossing (3)*, wer: *Werribee (4)*, wvl: *Wyndham Vale (5)*

Table C.2: Data inputs for *NetPlan* evaluation

Optimisation

NetPlan considers optimisation in a given time window (say 7:00AM to 3:00PM) and eliminates connections that fall outside it. Say, for example, a bus arriving at 6:55AM connecting with a train at 7:02AM is not considered a "successful" meet. To incorporate this feature, we modify our model and data such that the first bus on each route arrives at a station after 7:00AM and the last bus arrives on or before its maximum headway from 3:00PM.

Upon optimisation using the fixed bus headway criteria for trip-builders, *NetPlan* finds feasible solutions with buses as few as 26. Table C.3 summarises the *NetPlan* optimised bus-train meets for all the routes in the given sub-network of Wyndham.

Bus Block	Bus Start (hh:mm)	Bus Arrival (hh:mm)	Train Departure (hh:mm)	Wait (mins)
Route: 150 (Williams Landing)				
20	6:43	7:02	7:02	0
16	7:03	7:22	7:26	4
26	7:23	7:42	7:46	4
22	7:43	8:02	8:08	6
16	8:03	8:22	8:30	8
12	8:23	8:42	8:51	9
22	8:43	9:02	9:05	3
18	9:23	9:42	9:57	15
1	10:03	10:22	10:37	15
21	10:43	11:02	11:17	15
5	11:23	11:42	11:57	15
3	12:03	12:22	12:37	15
15	12:43	13:02	13:17	15
14	13:23	13:42	13:57	15
23	14:03	14:22	14:37	15
Route: 151 (Williams Landing)				
13	6:35	7:00	7:02	2
10	7:15	7:40	7:46	6
13	7:55	8:20	8:30	10
10	8:35	9:00	9:05	5
13	9:15	9:40	9:57	17
10	9:55	10:20	10:37	17
13	10:35	11:00	11:17	17
10	11:15	11:40	11:57	17
13	11:55	12:20	12:37	17
10	12:35	13:00	13:17	17
13	13:15	13:40	13:57	17
10	13:55	14:20	14:37	17
Route: 153 (Hoppers Crossing)				
22	6:47	7:00	7:07	7
12	7:27	7:40	7:42	2
18	8:07	8:20	8:26	6
1	8:47	9:00	9:01	1
21	9:27	9:40	9:53	13
5	10:07	10:20	10:33	13
3	10:47	11:00	11:13	13
15	11:27	11:40	11:53	13
14	12:07	12:20	12:33	13
23	12:47	13:00	13:13	13
18	13:27	13:40	13:53	13
20	14:07	14:20	14:33	13
Route: 160 (Hoppers Crossing)				
17	6:42	7:00	7:07	7
19	7:02	7:20	7:22	2
17	7:22	7:40	7:42	2
19	7:42	8:00	8:04	4
17	8:02	8:20	8:26	6
19	8:22	8:40	8:47	7
17	8:42	9:00	9:01	1
19	9:22	9:40	9:53	13
17	10:02	10:20	10:33	13
19	10:42	11:00	11:13	13
17	11:22	11:40	11:53	13
19	12:02	12:20	12:33	13
17	12:42	13:00	13:13	13
19	13:22	13:40	13:53	13
17	14:02	14:20	14:33	13

Bus Block	Bus Start (hh:mm)	Bus Arrival (hh:mm)	Train Departure (hh:mm)	Wait (mins)
Route: 150_A (Tarneit)				
16	6:39	7:00	7:04	4
26	6:59	7:20	7:32	12
22	7:19	7:40	7:42	2
16	7:39	8:00	8:00	0
12	7:59	8:20	8:31	11
22	8:19	8:40	8:49	9
18	8:39	9:00	9:18	18
1	9:19	9:40	9:58	18
21	9:59	10:20	10:38	18
5	10:39	11:00	11:18	18
3	11:19	11:40	11:58	18
15	11:59	12:20	12:38	18
14	12:39	13:00	13:18	18
23	13:19	13:40	13:58	18
18	13:59	14:20	14:38	18
Route: 151_A (Tarneit)				
10	6:33	7:00	7:04	4
13	7:13	7:40	7:42	2
10	7:53	8:20	8:31	11
13	8:33	9:00	9:18	18
10	9:13	9:40	9:58	18
13	9:53	10:20	10:38	18
10	10:33	11:00	11:18	18
13	11:13	11:40	11:58	18
10	11:53	12:20	12:38	18
13	12:33	13:00	13:18	18
10	13:13	13:40	13:58	18
13	13:53	14:20	14:38	18
Route: 153_A (Hoppers Crossing)				
8	6:29	6:41	6:46	5
20	7:09	7:21	7:22	1
26	7:49	8:01	8:04	3
16	8:29	8:41	8:47	6
22	9:09	9:21	9:33	12
18	9:49	10:01	10:13	12
1	10:29	10:41	10:53	12
21	11:09	11:21	11:33	12
5	11:49	12:01	12:13	12
3	12:29	12:41	12:53	12
15	13:09	13:21	13:33	12
14	13:49	14:01	14:13	12
Route: 160_A (Tarneit)				
19	6:43	7:00	7:04	4
17	7:03	7:20	7:32	12
19	7:23	7:40	7:42	2
17	7:43	8:00	8:00	0
19	8:03	8:20	8:31	11
17	8:23	8:40	8:49	9
19	8:43	9:00	9:18	18
17	9:23	9:40	9:58	18
19	10:03	10:20	10:38	18
17	10:43	11:00	11:18	18
19	11:23	11:40	11:58	18
17	12:03	12:20	12:38	18
19	12:43	13:00	13:18	18
17	13:23	13:40	13:58	18
19	14:03	14:20	14:38	18

Continued on the next page...

Bus Block	Bus Start (hh:mm)	Bus Arrival (hh:mm)	Train Departure (hh:mm)	Wait (mins)
Route: 161 (Hoppers Crossing)				
14	6:36	7:30	7:32	2
23	7:16	8:10	8:14	4
8	7:56	8:50	9:01	11
20	8:36	9:30	9:33	3
9	9:16	10:10	10:13	3
16	9:56	10:50	10:53	3
22	10:36	11:30	11:33	3
1	11:16	12:10	12:13	3
7	11:56	12:50	12:53	3
21	12:36	13:30	13:33	3
3	13:16	14:10	14:13	3
Route: 166 (Hoppers Crossing)				
4	7:08	7:38	7:42	4
6	7:48	8:18	8:26	8
7	8:28	8:58	9:01	3
4	9:08	9:38	9:53	15
6	9:48	10:18	10:33	15
4	10:28	10:58	11:13	15
11	11:08	11:38	11:53	15
6	11:48	12:18	12:33	15
11	12:28	12:58	13:13	15
6	13:08	13:38	13:53	15
5	13:48	14:18	14:33	15
Route: 167 (Hoppers Crossing)				
5	6:20	7:00	7:07	7
3	7:00	7:40	7:42	2
2	7:40	8:20	8:26	6
14	8:20	9:00	9:01	1
23	9:00	9:40	9:53	13
8	9:40	10:20	10:33	13
20	10:20	11:00	11:13	13
9	11:00	11:40	11:53	13
16	11:40	12:20	12:33	13
22	12:20	13:00	13:13	13
1	13:00	13:40	13:53	13
7	13:40	14:20	14:33	13
Route: 170 (Werribee)				
23	6:49	7:10	7:10	0
21	7:09	7:30	7:39	9
8	7:29	7:50	8:01	11
25	7:49	8:10	8:11	1
20	8:09	8:30	8:33	3
21	8:29	8:50	8:58	8
26	8:49	9:10	9:15	5
25	9:09	9:30	9:30	0
16	9:29	9:50	9:50	0
26	9:49	10:10	10:10	0
22	10:09	10:30	10:30	0
15	10:29	10:50	10:50	0
26	10:49	11:10	11:10	0
25	11:09	11:30	11:30	0
7	11:29	11:50	11:50	0
26	11:49	12:10	12:10	0
21	12:09	12:30	12:30	0
18	12:29	12:50	12:50	0
26	12:49	13:10	13:10	0
25	13:09	13:30	13:30	0
4	13:29	13:50	13:50	0
26	13:49	14:10	14:10	0
24	14:09	14:30	14:30	0
25	14:29	14:50	14:50	0

Bus Block	Bus Start (hh:mm)	Bus Arrival (hh:mm)	Train Departure (hh:mm)	Wait (mins)
Route: 161$_A$ (Werribee)				
11	6:33	7:28	7:29	1
5	7:13	8:08	8:11	3
3	7:53	8:48	8:58	10
2	8:33	9:28	9:30	2
14	9:13	10:08	10:10	2
23	9:53	10:48	10:50	2
8	10:33	11:28	11:30	2
20	11:13	12:08	12:10	2
9	11:53	12:48	12:50	2
16	12:33	13:28	13:30	2
22	13:13	14:08	14:10	2
Route: 166$_A$ (Wyndham Vale)				
7	7:01	7:35	7:35	0
4	7:41	8:15	8:25	10
6	8:21	8:55	9:12	17
7	9:01	9:35	9:52	17
4	9:41	10:15	10:32	17
6	10:21	10:55	11:12	17
4	11:01	11:35	11:52	17
11	11:41	12:15	12:32	17
6	12:21	12:55	13:12	17
11	13:01	13:35	13:52	17
6	13:41	14:15	14:32	17
Route: 167$_A$ (Tarneit)				
2	6:57	7:36	7:42	6
14	7:37	8:16	8:31	15
23	8:17	8:56	9:18	22
8	8:57	9:36	9:37	1
20	9:37	10:16	10:18	2
9	10:17	10:56	10:58	2
16	10:57	11:36	11:38	2
22	11:37	12:16	12:18	2
1	12:17	12:56	12:58	2
7	12:57	13:36	13:38	2
21	13:37	14:16	14:18	2
Route: 170$_A$ (Tarneit)				
21	6:44	7:06	7:17	11
8	7:04	7:26	7:32	6
25	7:24	7:46	7:47	1
20	7:44	8:06	8:11	5
21	8:04	8:26	8:31	5
26	8:24	8:46	8:49	3
25	8:44	9:06	9:18	12
16	9:04	9:26	9:37	11
26	9:24	9:46	9:58	12
22	9:44	10:06	10:18	12
15	10:04	10:26	10:38	12
26	10:24	10:46	10:58	12
25	10:44	11:06	11:18	12
7	11:04	11:26	11:38	12
26	11:24	11:46	11:58	12
21	11:44	12:06	12:18	12
18	12:04	12:26	12:38	12
26	12:24	12:46	12:58	12
25	12:44	13:06	13:18	12
4	13:04	13:26	13:38	12
26	13:24	13:46	13:58	12
24	13:44	14:06	14:18	12
25	14:04	14:26	14:38	12
14	14:24	14:46	15:18	32

Continued on the next page...

Bus Block	Bus Start (hh:mm)	Bus Arrival (hh:mm)	Train Departure (hh:mm)	Wait (mins)
Route: 180 (Werribee)				
18	6:43	7:04	7:04	0
12	7:03	7:24	7:29	5
1	7:23	7:44	7:49	5
18	7:43	8:04	8:11	7
11	8:03	8:24	8:33	9
1	8:23	8:44	8:44	0
5	8:43	9:04	9:15	11
11	9:03	9:24	9:30	6
3	9:23	9:44	9:50	6
5	9:43	10:04	10:10	6
2	10:03	10:24	10:30	6
3	10:23	10:44	10:50	6
14	10:43	11:04	11:10	6
2	11:03	11:24	11:30	6
23	11:23	11:44	11:50	6
14	11:43	12:04	12:10	6
8	12:03	12:24	12:30	6
23	12:23	12:44	12:50	6
20	12:43	13:04	13:10	6
8	13:03	13:24	13:30	6
9	13:23	13:44	13:50	6
20	13:43	14:04	14:10	6
16	14:03	14:24	14:30	6
9	14:23	14:44	14:50	6
Route: 190 (Werribee)				
24	6:49	7:02	7:04	2
25	7:09	7:22	7:29	7
24	7:29	7:42	7:49	7
21	7:49	8:02	8:11	9
24	8:09	8:22	8:23	1
25	8:29	8:42	8:44	2
24	8:49	9:02	9:15	13
21	9:09	9:22	9:30	8
24	9:29	9:42	9:50	8
25	9:49	10:02	10:10	8
24	10:09	10:22	10:30	8
25	10:29	10:42	10:50	8
24	10:49	11:02	11:10	8
15	11:09	11:22	11:30	8
24	11:29	11:42	11:50	8
25	11:49	12:02	12:10	8
24	12:09	12:22	12:30	8
25	12:29	12:42	12:50	8
24	12:49	13:02	13:10	8
18	13:09	13:22	13:30	8
24	13:29	13:42	13:50	8
25	13:49	14:02	14:10	8
4	14:09	14:22	14:30	8
26	14:29	14:42	14:50	8
Route: 191 (Werribee)				
15	6:37	7:00	7:04	4
9	7:17	7:40	7:49	9
15	7:57	8:20	8:23	3
9	8:37	9:00	9:15	15
15	9:36	9:59	10:10	11
7	10:35	10:58	11:10	12
18	11:34	11:57	12:10	13
4	12:33	12:56	13:10	14
2	13:32	13:55	14:10	15
11	14:31	14:54	15:10	16

Bus Block	Bus Start (hh:mm)	Bus Arrival (hh:mm)	Train Departure (hh:mm)	Wait (mins)
Route: 180_A **(Tarneit)**				
12	6:35	7:00	7:04	4
1	6:55	7:20	7:32	12
18	7:15	7:40	7:42	2
11	7:35	8:00	8:00	0
1	7:55	8:20	8:31	11
5	8:15	8:40	8:49	9
11	8:35	9:00	9:18	18
3	8:55	9:20	9:37	17
5	9:15	9:40	9:58	18
2	9:35	10:00	10:18	18
3	9:55	10:20	10:38	18
14	10:15	10:40	10:58	18
2	10:35	11:00	11:18	18
23	10:55	11:20	11:38	18
14	11:15	11:40	11:58	18
8	11:35	12:00	12:18	18
23	11:55	12:20	12:38	18
20	12:15	12:40	12:58	18
8	12:35	13:00	13:18	18
9	12:55	13:20	13:38	18
20	13:15	13:40	13:58	18
16	13:35	14:00	14:18	18
9	13:55	14:20	14:38	18
22	14:15	14:40	15:18	38
Route: 190_A **(Wyndham Vale)**				
25	6:53	7:07	7:11	4
24	7:13	7:27	7:35	8
21	7:33	7:47	7:54	7
24	7:53	8:07	8:14	7
25	8:13	8:27	8:39	12
24	8:33	8:47	9:12	25
21	8:53	9:07	9:12	5
24	9:13	9:27	9:31	4
25	9:33	9:47	9:52	5
24	9:53	10:07	10:12	5
25	10:13	10:27	10:32	5
24	10:33	10:47	10:52	5
15	10:53	11:07	11:12	5
24	11:13	11:27	11:32	5
25	11:33	11:47	11:52	5
24	11:53	12:07	12:12	5
25	12:13	12:27	12:32	5
24	12:33	12:47	12:52	5
18	12:53	13:07	13:12	5
24	13:13	13:27	13:32	5
25	13:33	13:47	13:52	5
4	13:53	14:07	14:12	5
26	14:13	14:27	14:32	5
24	14:33	14:47	15:12	25
Route: 191_A **(Wyndham Vale)**				
9	6:32	7:00	7:11	11
15	7:12	7:40	7:41	1
9	7:52	8:20	8:25	5
15	8:32	9:00	9:12	12
11	9:31	9:59	10:12	13
18	10:30	10:58	11:12	14
2	11:29	11:57	12:12	15
5	12:28	12:56	13:12	16
8	13:27	13:55	14:12	17
4	14:26	14:54	15:12	18

Continued on the next page...

Bus Block	Bus Start (hh:mm)	Bus Arrival (hh:mm)	Train Departure (hh:mm)	Wait (mins)
Route: 192 (Werribee)				
6	6:59	7:20	7:29	9
7	7:39	8:00	8:01	1
4	8:19	8:40	8:44	4
6	8:59	9:20	9:30	10
7	9:39	10:00	10:10	10
11	10:19	10:40	10:50	10
6	10:59	11:20	11:30	10
4	11:39	12:00	12:10	10
2	12:19	12:40	12:50	10
5	12:59	13:20	13:30	10
11	13:39	14:00	14:10	10

Bus Block	Bus Start (hh:mm)	Bus Arrival (hh:mm)	Train Departure (hh:mm)	Wait (mins)
Route: 192_A (Wyndham Vale)				
4	6:43	7:05	7:11	6
6	7:23	7:45	7:54	9
7	8:03	8:25	8:25	0
4	8:43	9:05	9:12	7
6	9:23	9:45	9:52	7
7	10:03	10:25	10:32	7
11	10:43	11:05	11:12	7
6	11:23	11:45	11:52	7
4	12:03	12:25	12:32	7
2	12:43	13:05	13:12	7
5	13:23	13:45	13:52	7
11	14:03	14:25	14:32	7

Table C.3: Summary of *NetPlan* optimised bus-train meets for each route and station pair in the Wyndham sub-network

C.4 Extracted output

Predominantly for this exercise, we only consider the *bus starting time per route* for the optimised 26 bus block to instantiate our *MiniZinc* optimisation. Table C.4 shows the *bus starting time per route* as extracted from *NetPlan* after optimisation. Note that the results reported here are devoid of any manual adjustments (that is, without any manual timetable shifts to improve meets further). The process of instantiating our models with this extracted output data is explained in Section 7.3.2.

***Routes*:**	150	150A	151	151A	153	153A	160	160A	161	161A	166	166A
***Start times*:**	6:43	6:39	6:35	6:33	6:47	7:09	6:42	6:43	6:36	6:33	7:08	7:01
***Routes*:**	167	167A	170	170A	180	180A	190	190A	191	191A	192	192A
***Start times*:**	6:20	6:57	6:49	6:44	6:43	6:35	6:49	6:53	6:37	6:32	6:59	6:43

Table C.4: Bus starting times per route after *NetPlan* optimisation (26 buses)

References

Australian Bureau of Statistics (2016), '2016 Census Quickstats: Greater Melbourne', https://quickstats.censusdata.abs.gov.au/census_services/getproduct/census/2016/quickstat/2GMEL. (Accessed on: 12/06/2019).

Baita, F., Pesenti, R., Ukovich, W. and Favaretto, D. (2000), 'A comparison of different solution approaches to the vehicle scheduling problem in a practical case', *Computers & Operations Research* **27**(13), 1249–1269.

Barra, A., Carvalho, L., Teypaz, N., Cung, V. and Balassiano, R. (2007), 'Solving the transit network design problem with constraint programming', *in* 11th World Conference in Transport Research-WCTR 2007.

Belov, G., Stuckey, P. J., Tack, G. and Wallace, M. (2016), 'Improved linearization of constraint programming models', *in* International Conference on Principles and Practice of Constraint Programming, Springer, pp. 49–65.

Bent, R. and Van Hentenryck, P. (2004), 'A two-stage hybrid local search for the vehicle routing problem with time windows', *Transportation Science,* **38**(4), 515–530.

Bertossi, A. A., Carraresi, P. and Gallo, G. (1987), 'On some matching problems arising in vehicle scheduling models', *Networks,* **17**(3), 271–281.

Bisbee, E., Devanney, J., Ward, D., Von Saal, R. and Kuroda, S. (1968), 'Dispatching policies for controlled systems', *in* Beitrage zur Theorie des Verkehrsflusses (Fourth International Symposium on the Theory of Traffic Flow, Karlsruhe 1968).

Blais, J. and Rousseau, J. (1988), 'Overview of hastus current and future versions', *in* Computer-Aided Transit Scheduling, Springer, pp. 175–187.

Bookbinder, J. H. and Desilets, A. (1992), 'Transfer optimization in a transit network', *Transportation Science,* **26**(2), 106–118.

Boschetti, M. A., Maniezzo, V., Roffilli, M. and Röhler, A. B. (2009), 'Matheuristics: Optimization, simulation and control', *in* International Workshop on Hybrid Metaheuristics, Springer, pp. 171–177.

Botea, A., Nikolova, E. and Berlingerio, M. (2013), 'Multi-modal journey planning in the presence of uncertainty', *in* Twenty-third international conference on automated planning and scheduling.

Bruno, G., Improta, G. and Sgalambro, A. (2009), 'Models for the schedule optimization problem at a public transit terminal', *OR Spectrum,* **31**(3), 465–481.

Bunte, S. and Kliewer, N. (2009), 'An overview on vehicle scheduling models', *Public Transport,* **1**(4), 299–317.

Bussieck, M. R., Winter, T. and Zimmermann, U. T. (1997), 'Discrete optimization in public rail transport', *Mathematical Programming,* **79**(1-3), 415–444.

Carosi, S., Frangioni, A., Galli, L., Girardi, L. and Vallese, G. (2019*a*), 'A tool for practical integrated time-table design and vehicle scheduling in public transport systems', *in* A View of Operations Research Applications in Italy, 2018, Springer, pp. 207–217.

Carosi, S., Frangioni, A., Galli, L., Girardi, L. and Vallese, G. (2019*b*), 'A matheuristic for integrated timetabling and vehicle scheduling', *Transportation Research Part B: Methodological,* **127**, 99–124.

Carpaneto, G., Dell'Amico, M., Fischetti, M. and Toth, P. (1989), 'A branch and bound algorithm for the multiple depot vehicle scheduling problem', *Networks,* **19**(5), 531–548.

Castelli, L., Pesenti, R. and Ukovich, W. (2004), 'Scheduling multimodal transportation systems', *European Journal of Operational Research,* **155**(3), 603–615.

Ceder, A. (2001), 'Efficient timetabling and vehicle scheduling for public transport', *in* Computer-Aided Scheduling of Public Transport, Springer, pp. 37–52.

Ceder, A. (2007), *Public Transit Planning and Operation: Modeling, Practice and Behavior, First Edition (1st ed.)*, CRC press.

Ceder, A. (2011), 'Public-transport vehicle scheduling with multi vehicle type', *Transportation Research Part C: Emerging Technologies,* **19**(3), 485–497.

Ceder, A. (2016), *Public Transit Planning and Operation: Modeling, Practice and Behavior, Second Edition (2nd ed.)*, CRC press.

Ceder, A., Golany, B. and Tal, O. (2001), 'Creating bus timetables with maximal synchronization', *Transportation Research Part A: Policy and Practice,* **35**(10), 913–928.

Ceder, A. and Wilson, N. H. (1986), 'Bus network design', *Transportation Research Part B: Methodological,* **20**(4), 331–344.

Cevallos, F. and Zhao, F. (2006), 'Minimizing transfer times in public transit network with genetic algorithm', *Transportation Research Record: Journal of the Transportation Research Board,* (1971), 74–79.

Chakroborty, P. (2003), 'Genetic algorithms for optimal urban transit network design', *Computer-Aided Civil and Infrastructure Engineering,* **18**(3), 184–200.

Chakroborty, P., Deb, K. and Porwal, H. (1997), 'A genetic algorithm based procedure for optimal transit systems scheduling', *in* Proceedings of Fifth International Conference on Computers in Urban Planning and Urban Management, Mumbai, India, pp. 330–341.

Chakroborty, P., Deb, K. and Sharma, R. K. (2001), 'Optimal fleet size distribution and scheduling of transit systems using genetic algorithms', *Transportation Planning and Technology,* **24**(3), 209–225.

Chakroborty, P., Deb, K. and Subrahmanyam, P. (1995), 'Optimal scheduling of urban transit systems using genetic algorithms', *Journal of Transportation Engineering,* **121**(6), 544–553.

Claessens, M., van Dijk, N. M. and Zwaneveld, P. J. (1998), 'Cost optimal allocation of rail passenger lines', *European Journal of Operational Research,* **110**(3), 474–489.

Currie, G. (2005), 'The demand performance of bus rapid transit', *Journal of Public Transportation,* **8**(1), 3.

Currie, G. and Bromley, L. (2005), 'Developing measures of public transport schedule coordination quality', *in* Transporting the Future: Transport in a changing environment: Proceedings of the 28th Australasian Transport research Forum, pp. 1–14.

Currie, G. and Loader, C. (2010), 'Bus network planning for transfers and the network effect in melbourne, australia', *Transportation Research Record: Journal of the Transportation Research Board,* (2145), 8–17.

Daduna, J. R. and Voß, S. (1995), 'Practical experiences in schedule synchronization', *in* Computer-Aided Transit Scheduling, Springer, pp. 39–55.

De Cea, E., Florian, M. and Cabrera, E. (1994), 'Network equilibrium models with combined modes', *Transportation Science* **28**(3).

de Palma, A. and Lindsey, R. (2001), 'Optimal timetables for public transportation', *Transportation Research Part B: Methodological,* **35**(8), 789–813.

Deb, K. and Chakroborty, P. (1998), 'Time scheduling of transit systems with transfer considerations using genetic algorithms', *Evolutionary Computation,* **6**(1), 1–24.

Desaulniers, G. and Hickman, M. D. (2007), 'Public transit', *Handbooks in Operations Research and Management Science,* **14**, 69–127.

Desrosiers, J., Dumas, Y., Solomon, M. M. and Soumis, F. (1995), 'Time constrained routing and scheduling', *Handbooks in Operations Research and Management Science,* **8**, 35–139.

Domschke, W. (1989), 'Schedule synchronization for public transit networks', *Operations-Research-Spektrum,* **11**(1), 17–24.

Dou, X., Gong, X., Guo, X. and Tao, T. (2017), 'Coordination of feeder bus schedule with train service at integrated transport hubs', *Transportation Research Record: Journal of the Transportation Research Board,* (2648), 103–110.

Dou, X., Meng, Q. and Guo, X. (2015), 'Bus schedule coordination for the last train service in an intermodal bus-and-train transport network', *Transportation Research Part C: Emerging Technologies,* **60**, 360–376.

Evans, J. E. (2004), 'Transit scheduling and frequency - traveler response to transportation system changes (TCRP-95), Technical report, Transportation Research Board, Washington D.C.

Fleurent, C. and Lessard, R. (2009), 'Integrated timetabling and vehicle scheduling', Technical report, GIRO.

Fleurent, C., Lessard, R. and Séguin, L. (2004), 'Transit timetable synchronization: Evaluation and optimization', *in* Proceedings of the 9th International Conference on Computer-Aided Scheduling of Public Transport (CASPT). San Diego, California.

Fonseca, J. P., van der Hurk, E., Roberti, R. and Larsen, A. (2018), 'A matheuristic for transfer synchronization through integrated timetabling and vehicle scheduling', *Transportation Research Part B: Methodological,* **109**, 128–149.

Freling, R., Huisman, D. and Wagelmans, A. P. (2001), 'Applying an integrated approach to vehicle and crew scheduling in practice', *in* Computer-aided scheduling of public transport, Springer, pp. 73–90.

Gkiotsalitis, K. and Maslekar, N. (2018), 'Towards transfer synchronization of regularity-based bus operations with sequential hill-climbing', *Public transport* **10**(2), 335–361.

Guedes, P. C. and Borenstein, D. (2015), 'Column generation based heuristic framework for the multiple-depot vehicle type scheduling problem', *Computers & Industrial Engineering,* **90**, 361–370.

Guedes, P. C., Borenstein, D., Visentini, M. S., de Araújo, O. C. B. and Neto, A. F. K. (2019), 'Vehicle scheduling problem with loss in bus ridership', *Computers & Operations Research,* **111**, 230–242.

Guihaire, V. and Hao, J. K. (2008*a*), 'Transit network design and scheduling: A global review', *Transportation Research Part A: Policy and Practice,* **42**(10), 1251–1273.

Guihaire, V. and Hao, J. K. (2008*b*), 'Transit network re-timetabling and vehicle scheduling', *in* International Conference on Modelling, Computation and Optimization in Information Systems and Management Sciences, Springer, pp. 135–144.

Guihaire, V. and Hao, J. K. (2010), 'Transit network timetabling and vehicle assignment for regulating authorities', *Computers & Industrial Engineering,* **59**(1), 16–23.

Guo, Z. (2003), 'Assessment of the transfer penalty to transit trips in Downtown Boston: a GIS-based disaggregate modeling approach', PhD thesis, Massachusetts Institute of Technology.

Hadas, Yuval Ceder, A. (2010), 'Optimal coordination of public-transit vehicles using operational tactics examined by simulation', *Transportation Research Part C: Emerging Technologies,* **18**(6), 879–895.

Hadjar, A., Marcotte, O. and Soumis, F. (2006), 'A branch-and-cut algorithm for the multiple depot vehicle scheduling problem', *Operations Research,* **54**(1), 130–149.

Hassold, S. and Ceder, A. (2012), 'Multiobjective approach to creating bus timetables with multiple vehicle types', *Transportation Research Record,* **2276**(1), 56–62.

Horowitz, A. J. and A, T. N. (1994), *Evaluation of intermodal passenger transfer facilities*, Report Prepared for the Federal Highway Administration, Milwaukee, WI.

Ibarra-Rojas, O., Delgado, F., Giesen, R. and Muñoz, J. (2015), 'Planning, operation, and control of bus transport systems: A literature review', *Transportation Research Part B: Methodological,* **77**, 38–75.

Ibarra-Rojas, O. J., Giesen, R. and Rios-Solis, Y. A. (2014), 'An integrated approach for timetabling and vehicle scheduling problems to analyze the trade-off between level of service and operating costs of transit networks', *Transportation Research Part B: Methodological,* **70**, 35–46.

Ibarra-Rojas, O. J. and Rios-Solis, Y. A. (2012), 'Synchronization of bus timetabling', *Transportation Research Part B: Methodological,* **46**(5), 599–614.

Iseki, Hiroyuki Taylor, B. D. (2009), 'Not all transfers are created equal: Towards a framework relating transfer connectivity to travel behaviour', *Transport Reviews,* **29**(6), 777–800.

Jansen, L. N., Pedersen, M. B. and Nielsen, O. A. (2002), 'Minimizing passenger transfer times in public transport timetables', *in* 7th Conference of the Hong Kong Society for Transportation Studies, Transportation in the information age, Hong Kong, pp. 229–239.

Johnson, D., Shires, J., Nash, C. and Tyler, J. (2006), 'Forecasting and appraising the impact of a regular interval timetable', *Transport Policy,* **13**(5), 349–366.

Klemt, W. D. and Stemme, W. (1988), 'Schedule synchronization for public transit networks', *in* Computer-Aided Transit Scheduling, Springer, pp. 327–335.

Kliewer, N., Mellouli, T. and Suhl, L. (2006), 'A time–space network based exact optimization model for multi-depot bus scheduling', *European Journal of Operational Research* **175**(3), 1616–1627.

Knoppers, P. and Muller, T. (1995), 'Optimized transfer opportunities in public transport', *Transportation Science,* **29**(1), 101–105.

Kroon, L., Huisman, D., Abbink, E., Fioole, P. J., Fischetti, M., Maróti, G., Schrijver, A., Steenbeek, A. and Ybema, R. (2009), 'The new dutch timetable: The or revolution', *Interfaces,* **39**(1), 6–17.

Kwan, C. M. and Chang, C. S. (2008), 'Timetable synchronization of mass rapid transit system using multiobjective evolutionary approach', *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews),* **38**(5), 636–648.

Lee, K. K. and Schonfeld, P. (1991), 'Optimal slack time for timed transfers at a transit terminal', *Journal of Advanced Transportation,* **25**(3), 281–308.

Li, W., Luo, Q. and Cai, Q. (2019), 'Coordination of last train transfers using potential passenger demand from public transport modes', *IEEE Access* **7**, 126037–126050.

Li, X., Yamamoto, T., Yan, T., Lu, L. and Ye, X. (2020), 'First train timetabling for urban rail transit networks with maximum passenger transfer satisfaction', *Sustainability* **12**(10), 4166.

Liebchen, C., Proksch, M. and Wagner, F. H. (2008), 'Performance of algorithms for periodic timetable optimization', *in* Computer-aided systems in public transport, Springer, pp. 151–180.

Liebchen, C. and Stiller, S. (2012), 'Delay resistant timetabling', *Public Transport,* **1**(1), 55–72.

Liu, T. and Ceder, A. (2017), "Integrated public transport timetable synchronization and vehicle scheduling with demand assignment: a bi-objective bi-level model using deficit function approach", *Transportation Research Procedia* pp. 341–361.

Liu, Z., Jinsheng, S., Haixing, W. and Wei, Y. (2007), 'Regional bus timetabling model with synchronization', *Journal of Transportation Systems Engineering and Information Technology,* **7**(2), 109–112.

Liu, Z. and Shen, J. (2007), 'Regional bus operation bi-level programming model integrating timetabling and vehicle scheduling', *Systems Engineering-Theory & Practice,* **27**(11), 135–141.

Löbel, A. (1998), 'Vehicle scheduling in public transit and lagrangean pricing', *Management Science* **44**(12-part-1), 1637–1649.

Lübbecke, M., Puchert, C., Schiewe, P. and Schöbel, A. (2019), 'Integrating line planning, timetabling, and vehicle scheduling', *in* Proceedings of the 14th International Conference on Advanced Systems in Public Transport, Brisbane, Australia.

Mees, P. (2000), *A very public solution: Transport in the dispersed city*, Melbourne University Press.

Michaelis, M. and Schöbel, A. (2009), 'Integrating line planning, timetabling, and vehicle scheduling: a customer-oriented heuristic', *Public Transport,* **1**(3), 211.

Nachtigall, K. (1998), 'Periodic network optimization and fixed interval timetables'.

Nethercote, N., Stuckey, P. J., Becket, R., Brand, S., Duck, G. J. and Tack, G. (2007), 'Minizinc: Towards a standard cp modelling language', *in* International Conference on Principles and Practice of Constraint Programming, Springer, pp. 529–543.

Newell, G. F. (1971), 'Dispatching policies for a transportation route', *Transportation Science,* **5**(1), 91–105.

Nonner, T. (2012), Polynomial-time approximation schemes for shortest path with alternatives, *in* European Symposium on Algorithms, Springer, pp. 755–765.

Pätzold, J., Schiewe, A., Schiewe, P. and Schöbel, A. (2017), 'Look-ahead approaches for integrated planning in public transportation', *in* 17th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2017), Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

Pepin, A. S., Desaulniers, G., Hertz, A. and Huisman, D. (2009), 'A comparison of five heuristics for the multiple depot vehicle scheduling problem', *Journal of Scheduling,* **12**(1), 17.

Petersen, H. L., Larsen, A., Madsen, O. B., Petersen, B. and Ropke, S. (2013), 'The simultaneous vehicle scheduling and passenger service problem', *Transportation Science,* **47**(4), 603–616.

Pisinger, D. and Ropke, S. (2007), 'A general heuristic for vehicle routing problems', *Computers & Operations Research,* **34**(8), 2403–2435.

Pisinger, D. and Ropke, S. (2010), 'Large Neighborhood Search, *in* Handbook of Metaheuristics, Springer, pp. 399–419.

Polinder, G. (2015), Resolving infeasibilities in the pesp model of the dutch railway timetabling problem, Master's thesis, Delft University of Technology, the Netherlands.

Poorjafari, V. and Yue, W. (2013), Synchronisation of public transport services: comparison of methods, *in* Proceedings of the 10th International Conference of Eastern Asia Society for Transportation Studies. EASTS Press, Taiwan.

Public Transport Victoria (2012), Network Development Plan - Metropolitan Rail, Technical report.

Public Transport Victoria (2014), Service Specification: Wyndham April 2015 Bus Network (CDC Victoria), Technical report.

Public Transport Victoria (2020), 'Operator contact details', https://www.ptv.vic.gov.au/footer/customer-service/operator-contact-details/. (Accessed on: 20/07/2020).

Robenek, T., Azadeh, S. S., Maknoon, Y. and Bierlaire, M. (2017), 'Hybrid cyclicity: Combining the benefits of cyclic and non-cyclic timetables', *Transportation Research Part C: Emerging Technologies,* **75**, 228–253.

Salzborn, F. J. (1972), 'Optimum bus scheduling', *Transportation Science,* **6**(2), 137–148.

Schiewe, P. (2020), 'Integrating timetabling and vehicle scheduling', *in* Integrated Optimization in Public Transport Planning, Springer, pp. 91–98.

Schmid, V. and Ehmke, J. F. (2015), 'Integrated timetabling and vehicle scheduling with balanced departure times', *OR Spectrum,* **37**(4), 903–928.

Schöbel, A. (2017), 'An eigenmodel for iterative line planning, timetabling and vehicle scheduling in public transportation', *Transportation Research Part C: Emerging Technologies,* **74**, 348–365.

Schröder, M. and Solchenbach, I. (2006), Optimization of transfer quality in regional public transit, Technical report, Berichte des Fraunhofer Instituts for Techno-und Wirtschafts.

Schuele, I., Schroeder, M. and Kuefer, K. (2009), 'Synchronization of regional public transport systems', *in* Urban Transport XV-Urban Transport and the Environment in the 21st Century: 22-24 June, p. 301.

Serafini, P. and Ukovich, W. (1989), 'A mathematical model for periodic scheduling problems', *SIAM Journal on Discrete Mathematics,* **2**(4), 550–581.

Shafahi, Y. and Khani, A. (2010), 'A practical model for transfer optimization in a transit network: Model formulations and solutions', *Transportation Research Part A: Policy and Practice,* **44**(6), 377–389.

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

Shrivastava, P. and Dhingra, S. (2002), 'Development of coordinated schedules using genetic algorithms', *Journal of Transportation Engineering* **128**(1), 89–96.

Tilahun, S. L. and Ong, H. C. (2012), 'Bus timetabling as a fuzzy multiobjective optimization problem using preference-based genetic algorithm', *Promet-Traffic & Transportation,* **24**(3), 183–191.

Torrance, K. B., Haire, A. R., Machemehl, R. B. et al. (2009), 'Vehicle and driver scheduling for public transit', Technical report, Center for Transportation Research, University of Texas at Austin.

Van den Heuvel, A., Van den Akker, J. and Van Kooten, M. (2008), Integrating timetabling and vehicle scheduling in public bus transportation, Technical report.

Van Oort, N. and van Nes, R. (2004), 'Service regularity analysis for urban transit network design', *in* Proceedings of 83rd Annual Meeting of Transportation Research Board, pp. 1–26.

Voß, S. (1992), 'Network design formulations in schedule synchronization,, *in* Computer-Aided Transit Scheduling, Springer, pp. 137–152.

Vuchic, V. (2005), *Urban transit: operations*, John Wiley & Sons.

Walker, J. (2012), *Human transit: How clearer thinking about public transit can enrich our communities and our lives*, Island Press.

Wallace, M. (2020), *Building Decision Support Systems: Using MiniZinc*, Springer Nature.

Weiszer, M., Fedorko, G. and Čujan, Z. (2010), 'Multiobjective evolutionary algorithm for integrated timetable optimization with vehicle scheduling aspects', *Perner's Contacts* **5**(4), 286–294.

Wong, R. C. and Leung, J. M. (2004), 'Timetable synchronization for mass transit railway', *in* Proceedings of the 9th International Conference on Computer-Aided Scheduling of Public Transport (CASPT), San Diego, CA.

Wong, R. C., Yuen, T. W., Fung, K. W. and Leung, J. M. (2008), 'Optimizing timetable synchronization for rail mass transit', *Transportation Science* **42**(1), 57–69.

Wu, J., Liu, M., Sun, H., Li, T., Gao, Z. and Wang, D. Z. (2015), 'Equity-based timetable synchronization optimization in urban subway network', *Transportation Research Part C: Emerging Technologies,* **51**, 1–18.

Zhao, F. and Zeng, X. (2008), 'Optimization of transit route network, vehicle headways and timetables for large-scale transit networks', *European Journal of Operational Research,* **186**(2), 841–855.