Modelling, Solution and Evaluation Techniques for Train Timetable Rescheduling via Optimisation

—

Submitted for the degree of Doctor of Philosophy (PhD)
STOR-i, Lancaster University
December 2020

—

Edwin Reynolds, BA. MSc. MRes.

—

STOR-i
excellence with impact

Lancaster University

Abstract

It is common on railways for a single train delay to cause other trains to become delayed, multiplying the negative consequences of the original problem. However, making appropriate changes to the timetable in response to the initial delay can help to reduce the amount of further delay caused. In this thesis, we tackle the Train Timetable Rescheduling Problem (TTRP), the task of finding the best combination of timetable changes to make in any given traffic scenario.

The TTRP can be formulated as an optimisation problem and solved computationally to aid the process of railway traffic control. Although this approach has received considerable research attention, the practical deployment of optimisation methods for the TTRP has hitherto been limited. In this thesis, we identify and address three outstanding research challenges that remain barriers to deployment.

First, we find that existing TTRP models for large station areas are either not sufficiently realistic or cannot be solved quickly enough to be used in a real-time environment. In response, a new TTRP model is introduced that models the signalling system in station areas in fine detail. Using a new set of real instances from Doncaster station, we show that our tailored solution algorithm can obtain provably optimal or near-optimal solutions in sufficiently short times.

Second, we argue that existing ways of modelling train speed in TTRP models are

either unrealistic, overly complex, or lead to models that cannot be solved in real-time. To address this, innovative extensions are made to our TTRP model that allow speed to be modelled parsimoniously. Real instances for Derby station are used to demonstrate that these modelling enhancements do not incur any extra computational cost.

Finally, a lack of evidence is identified concerning the fairness of TTRP models with respect to competing train operators. New evaluation techniques are developed to fill this gap, and these techniques are applied to a case study of Doncaster station. We find that unfairness is present when efficiency is maximised, and find that it mostly results from competition between a small number of operators. Moreover, we find that fairness can be improved up to a point by increasing the priority given to local trains.

This work represents an important step forward in optimisation techniques for the TTRP. Our results, obtained using real instances from both Doncaster and Derby stations, add significantly to the body of evidence showing that optimisation is a viable approach for the TTRP. In the long run this will make deployment of such technology more likely.

Acknowledgements

I am profoundly grateful for all of the support I have received during my PhD study. First and foremost, I'd like to thank my team of supervisors:

Matthias Ehrgott	for offering sage guidance and encouraging research freedom;
	for your Olympian ability to spot errors in draft texts;
Stephen Maher	for all the time and effort you have put in to help me succeed;
	for always having a smart suggestion up your sleeve;
Judith Wang	for helping me to see the bigger picture, and making me laugh;
	for the unforgettable trip to Japan, and my trips to Leeds;
Tony Patman	for helping me to navigate Network Rail, and making it fun;
	for your enthusiasm for and commitment to the project.

Second, thanks go to people in the communities of which I have been a part:

STOR-i Staff	for making the centre well-organised, busy and friendly;
STOR-i Students	for the friendships, banter, and what I have learnt from you;
Jon Tawn	for your tireless mentorship;
Mng Sci Dept	for seminars, reading groups and interesting conversations.

Finally, enduring thanks go to all the people that have supported me in my personal life. I'm glad that we got through 2020 together.

- To all of my friends in Lancaster and otherwise: you are more important to me than you probably realise.
- In particular to Euan, Γεωργία, 瀚葵, Seán and Zak: you can't escape me just by moving away from Lancaster. I think we are friends for life.
- To my family: I am so lucky to have your unconditional love and support in whatever endeavours I pursue. I *literally* wouldn't be here without you.
- To Rob: Your selfless willingness to move with me to Lancaster made this possible. We will look back with joy on the happy years we spent here together.

Addendum: I'd like to thank Jesper Larsen and Adam Letchford for examining this thesis. It was a genuine pleasure to be able to discuss my work with them.

Declaration

I declare that the work in this thesis has been done by myself and has not been submitted elsewhere for the award of any other degree.

Chapter 5 has been submitted for publication as Reynolds, E., Ehrgott, M., Maher, S. J., Patman, A. and Wang, J. Y. T. (2020). A multicommodity flow model for rerouting and retiming trains in real-time to reduce reactionary delay in complex station areas. In Chapters 6 and 7, references to this paper (which look like [Reynolds et al., 2020]) can be interpreted as referring to Chapter 5.

Chapter 6 has been submitted for publication as Reynolds, E. and Maher, S. J. (2021). A data-driven, variable-speed model for the train timetable rescheduling problem.

Chapter 7 is being prepared for publication as Reynolds, E., Ehrgott, M. and Wang, J. Y. T. (2021). An evaluation of the fairness of railway timetable rescheduling in the presence of competition between train operators.

Edwin Reynolds

Contents

Abstract **2**

Acknowledgements **4**

Declaration **6**

Contents **12**

List of Figures **15**

List of Tables **15**

1 Introduction **16**

1.1 Context 18

1.1.1 Delays 19

1.1.2 Timetable Rescheduling 21

1.1.3 The Train Timetable Rescheduling Problem 22

1.2 Motivation 23

1.2.1 The Deployment Deficit 24

1.2.2 Barriers to Deployment 25

1.3 Research Aim 25

1.4 Significance and Impact 26

1.5 Overview 27

2 Optimisation Background 30

2.1 Mixed Integer Programming 31

2.2 Linear Programming 33

2.3 Column Generation 36

2.4 Dantzig-Wolfe Reformulation 38

2.5 Branch-and-price 41

2.6 Multicommodity Flow and Time-space Graphs 42

2.7 Summary 44

3 Optimisation for Railway Planning 46

3.1 Planning Horizons 47

3.2 Railway Infrastructure Representation 49

3.3 Timetabling and Timetable Rescheduling 50

3.4 Timetable, Crew and Rolling Stock Rescheduling 52

3.5 Delay Management 53

3.6 Summary 54

4 TTRP Formulations 55

4.1 A Generic TTRP Formulation 55

4.2 Disjunctive Formulations 57

4.2.1 Alternative Graph Model 59

4.2.2 Disjunctive MILP Formulations 61

4.2.3 Discussion 63

4.3 Time-indexed Formulations 63

4.3.1 Arc Packing 64

4.3.2 Path Packing 65

4.3.3 Resource Trees 67

4.3.4 Path Configuration 68
4.3.5 Cumulative Flow 68
4.4 Discussion 69

5 A multicommodity flow model for rerouting and retiming trains in real-time to reduce reactionary delay in complex station areas 71

5.1 Introduction 71
5.2 Railway Signalling 73
5.3 Literature Review 75
5.4 Modelling the Train Timetable Rescheduling Problem 81
5.4.1 Description of the Problem 81
5.4.2 Modelling Approach 81
5.4.3 The Time Horizon 82
5.4.4 The Route Graph 83
5.4.5 The Time-space Graph 85
5.4.6 Track Capacity 87
5.4.7 The Schedule 96
5.4.8 The Objective Function 97
5.4.9 Arc Weights 101
5.5 Integer Programming Formulation 103
5.5.1 Arc Formulation 103
5.5.2 Danzig-Wolfe Decomposition into a Path Formulation 104
5.6 Branch-and-Price Algorithm 106
5.6.1 Column Generation 106
5.6.2 Subproblem Solution 107
5.6.3 Acceleration Strategy 1: Partial Pricing 108
5.6.4 Acceleration Strategy 2: Reduced Cost Variable Fixing 108
5.6.5 Branching 110

5.7 Instance Data . . . 113
5.7.1 Track Data . . . 114
5.7.2 Traversal Time Estimation . . . 115
5.7.3 Timetable Data . . . 117
5.8 Computational Results . . . 119
5.8.1 Solving to Optimality . . . 120
5.8.2 Solving with a Strict Time Limit . . . 124
5.9 Conclusion . . . 125

6 A data-driven, variable-speed model for the train timetable rescheduling problem 127

6.1 Introduction . . . 127
6.1.1 Problem Description . . . 128
6.1.2 Structure of the Paper . . . 129
6.2 Literature Review . . . 129
6.2.1 Fixed-speed and Variable-speed Models . . . 130
6.2.2 Physics-based and Data-driven Models . . . 133
6.2.3 Contributions . . . 135
6.3 Model Overview . . . 136
6.3.1 Example TST Graph . . . 137
6.3.2 Speed Profile Types . . . 139
6.4 Traversal Time Estimation . . . 140
6.4.1 Method 1: Estimating a Single Time (Fixed-Speed) . . . 141
6.4.2 Method 2: Extension to Multiple Times (Variable-Speed) . . . 143
6.4.3 Method 3: Transition-based Multiple Traversal Time Estimation 146
6.5 Model Description and Solution Method . . . 148
6.5.1 TST Graph . . . 148
6.5.2 Track Capacity Constraints . . . 150

6.5.3 Antichain Condition 153
6.5.4 Objective Function and Arc Weights 155
6.5.5 MIP Formulation and Solution 157
6.6 Computational Study 158
6.6.1 Instance Data 159
6.6.2 Traversal Time Estimation 163
6.6.3 Algorithmic Performance 164
6.7 Conclusion 168

7 An evaluation of the fairness of railway timetable rescheduling in the presence of competition between train operators 170

7.1 Introduction 170
7.2 Literature Review 173
7.2.1 Fairness 173
7.2.2 Fairness and the TTRP 176
7.3 The Case Study 178
7.4 Efficiency and Fairness 180
7.4.1 Efficiency 181
7.4.2 Fairness 182
7.5 Evaluation of Fairness under Maximal Efficiency 185
7.6 Identifying Sources of Unfairness 188
7.7 The Effect of Train Class Weight on Efficiency and Fairness 193
7.8 Conclusions 196

8 Conclusions 198

8.1 Contributions 198
8.2 Further Research 201
8.2.1 Modelling Larger Areas 201

8.2.2 Aggregation based Solution Methods 202

8.2.3 Simulation and Benchmarking for Model Evaluation 203

Bibliography **205**

List of Figures

5.1 An example berth graph and corresponding route graph. 85

5.2 An example time-space graph. 86

5.3 An example of routes that share track circuits. 89

5.4 An example of routes to illustrate banning. 89

5.5 An example of $W_{r,t}$, $\bar{W}_{r,t}$, $A_{r,t}$ and $\bar{A}_{r,t}$. 91

5.6 An example in which Assumption 1 does not hold. 95

5.7 A sketch of γ. 100

5.8 A berth diagram of Doncaster Station. 115

5.9 A berth diagram of the area of track modelled. 116

5.10 Histograms showing the test set distribution of the number of (a) trains, (b) conflicts and (c) train pair conflicts. 119

5.11 A cumulative histogram showing the proportion of instances solved for each number of seconds. 121

5.12 Performance profiles at assess column generation acceleration strategies. 122

5.13 The relationship between conflicts and the number of branch-and-bound nodes. 123

5.14 A bar plot of the number of reroutings in each solution. 124

5.15 A histogram of the percentage gap between the best integer feasible solution found, and the best bound on the optimal solution. 125

6.1 A plot showing how the speed profile assumptions made in fixed-speed models compare to actual speed profiles. 131
6.2 The route graph G_r for the area centred on Derby station that is used in our computational experiments. 137
6.3 An example of a TST graph. 138
6.4 Speed profile type graph for a fixed-speed model. 139
6.5 Speed profile type graph for our approach. 140
6.6 Histogram of historical traversal times for route 5332_5330. 142
6.7 The Markov chain $(X_t)_{t\in\mathbb{N}}$, with estimated transition probabilities shown. Probabilities are rounded to two decimal places. 145
6.8 Boxplots showing the distribution of transition probabilities by type. 146
6.9 Histogram of classified historical traversal times for a particular route. 148
6.10 An example TST graph, with examples of arcs from different sets A_1–A_6 labelled. 150
6.11 An example of $W_{r,t}$ and $A_{r,t}$. 154
6.12 A berth diagram of the modelled area. 161
6.13 A histogram showing the distribution of the number of trains in the instances. 162
6.14 A comparison between (FS) and (VS) of number of conflicts. 162
6.15 A comparison between (FS) and (VS) of traversal time lengths for routes that have two traversal times in (VS). 165
6.16 Cumulative histograms showing (a) solution time (seconds), (b) number of branch-and-bound nodes and (c) number of variables generated, over the set of instances. 167
6.17 Cumulative histogram of optimality gaps after 20 seconds for (FS) and (VS). 168

7.1 A berth diagram of Doncaster Station. 179

7.2 A berth diagram of the area of track modelled. 179
7.3 Normed aggregated utility, number of instances with utility of 1, and number of instances for each operator. 186
7.4 Histogram of α-fairness scores of instances, with $\alpha = 1$. 187
7.5 The distribution of $\hat{U}_{i,o}(x_i)$ over the set of instances. 187
7.6 Values of $\hat{U}_{o'\setminus o}(\mathbf{x})$ in 100^{ths} of a percent. 190
7.7 Number of instances out of 310 for which $\hat{U}_{o'\setminus o}(x^i) > 0$. 190
7.8 A graph of the operators in which the width of arc (o', o) corresponds to $\hat{U}_{o'\setminus o}(\mathbf{x})$. 191
7.9 Efficiency and fairness for different values of w. 194
7.10 The change in normalised aggregated operator utility for different values of w. 194

List of Tables

6.1 A comparison between (FS) and (VS) of train pair conflicts. 163

7.1 The passenger operators in the case study. 180

Chapter 1

Introduction

Passenger railway services form an important part of the transport mix in many countries around the world. This is due to the unique combination of benefits that railway transport offers. Train services are attractive to passengers because they are the fastest way to travel between many pairs of urban centres. Train services are also able to maintain very high levels of both safety and passenger comfort. When the volume of passenger demand for railway services is high, they can be delivered at a competitive cost. For this reason, railways are ideal for both intercity travel and commuting within large urban areas.

Passenger railway transport also offers significant benefits to wider society. The environmental impact per passenger kilometre — with regard to the emission of CO_2, harmful air pollutants and noise — compares favourably with private car use. High levels of railway usage can also contribute to a reduction in road congestion, making cities more pleasant for road users and pedestrians alike. Because of the high modal share railway enjoys with commuters and business travellers, railways also have a significant strategic importance to the economies they serve.

However, the realisation of these potential benefits poses organisational challenges for railways. Because railway infrastructure is expensive to build and maintain, it must be highly utilised in order for services to be delivered at a competitive cost. This is particularly true of railway track, which is often the most expensive part. In order to achieve the necessary levels of infrastructure utilisation, railway systems require high levels of coordination and planning.

Railway planning, which ranges from very long term planning to operational decision making, is subject to several acute challenges. Railways are complex physical networks of technology that span large geographical areas. As a result, technical and organisational failures are an ever-present reality. Furthermore, the parts and processes that make up railway networks are highly interdependent, with decisions about one part of the system often having ramifications in a another part. This means that successful planning must encompass combinations of decisions and consider their likely effects on the system as a whole. However, efforts at coordinated planning lead to planning problems with many stakeholders, who must each coordinate their access to railway infrastructure with a fixed and limited capacity. To further complicate planning, railways are safety-critical systems which must be operated in compliance with a myriad of safety protocols.

A specific challenge for railway planning is the delivery of punctual and reliable passenger services. Whilst good quality long and medium term planning can help to reduce the likelihood of train delays, the possibility of delays cannot be eliminated. When delays do occur, the interdependency of train movements can cause them to propagate to other trains, causing potentially large-scale traffic disruption. In this thesis, we consider ways to counter this propagation with proactive and coordinated real-time planning. Whilst this problem exemplifies the challenges inherent in railway planning, it also has great potential to boost the unique combination of benefits offered by railway travel.

1.1 Context

Railway transport was pioneered in Great Britain, beginning with the opening of the world's first public inter-city railway, the Liverpool and Manchester Railway, in 1830. Over the last 190 years, this has grown into a large public passenger railway system that serves almost all corners of the island. Today, the network consists of approximately 20,000 miles of track, and serves 2566 stations [Department for Transport, 2019].

The railway is owned by Network Rail, which is a public sector company. This includes the track, signalling equipment, bridges, viaducts, tunnels, level crossings and 18 of the largest stations. In addition to maintaining and improving this network, Network Rail is responsible for operating it. This includes providing signalling and control services, and coordinating the planning of the system as a whole.

Passenger and freight services are provided by Train Operating Companies (TOCs) and Freight Operating Companies (FOCs) respectively. These are separate organisations from Network Rail, and operate services within a partially deregulated system. TOCs include private companies, public sector companies, foreign state-owned companies and consortia of the above. TOCs operate services under both franchise agreements and open access agreements. This system is supported by a number of other regulatory and coordinating organisations such as Department for Transport, the Office of Rail and Road, the Rail Delivery Group and the Rail Safety and Standards Board. A more detailed explanation of the structure of the industry is provided by Williams [2019a].

The number of passenger railway journeys taken in Great Britain increased by 97% in the 20 year period 1999–2019 [Department for Transport, 2019]. In 2018–2019, around 1.8 billion passenger journeys were made, the largest number in the history

of the network. Despite the severe disruption to passenger numbers caused by the COVID-19 pandemic beginning in March 2020, this trend is widely expected to continue in the long term. In response to rising passenger numbers, the number of scheduled passenger services has increased to satisfy demand. In the nine years to 2018–2019, the number of planned services increased by 24%. In many parts of the network, this has occurred at a faster rate than increases in the capacity of the track. As a result, the timetables operated in these areas are very dense, with the planned time between trains little more than is necessary from a safety perspective. Indeed, the British railway system is now one of the most heavily congested in Europe, surpassed only by Switzerland and the Netherlands [Williams, 2019b, p. 22].

1.1.1 Delays

The increasing density of timetables in recent years has contributed to an increase in delays. A *train delay* has occurred when a train arrives at a passenger stop later than advertised by the timetable. Delays are a common feature of railway systems around the world, including the British railway system where they are a widespread and frequent occurrence. The development of novel methods for reducing delays is the topic of this thesis.

One measure of the prevalence of delays used by Network Rail is the *Public Performance Measure* (PPM). This measures the percentage of trains running their entire planned journey including all scheduled stations and arriving at their terminating station within 5 (local services) or 10 (long distance services) minutes. PPM was 87% in 2019, lower than the average value over the previous 10 years [Network Rail, 2020]. In April 2019, Network Rail began using a new measure called *On-time at All Recorded Stations*. This measures the percentage of all train stops that occurred on time, and within various bands of lateness. In 2018–2019, only 65.3% of recorded

passenger train stops occurred on time [Network Rail, 2020].

Train delays have a variety of negative impacts. The most obvious is that they cause passengers to arrive late at their destinations. In addition to the obvious social costs, this has significant economic costs. For example, in Great Britain the time that passengers lose each year as a result of arriving late at their destinations has been estimated to be worth at least £1 billion [National Audit Office, 2008, p. 46].

Train delays also have negative impacts for the railway industry as a whole. For example, there are costs directly associated with delays, such as compensation for passengers and additional labour costs for managing delay. There is evidence that the ways in which train companies deal with delays are the biggest drivers of passenger dissatisfaction [Williams, 2019b, p. 13]. In the long term, poor punctuality will make railway services less attractive to passengers, and dampen demand for them.

The causes of delays are extensively recorded and analysed by Network Rail. This is important because understanding the causes is a prerequisite to reducing delays. It is also important because TOCs and Network Rail are subject to Schedule 8, a legal mechanism by which organisations responsible for causing delays must compensate those affected.

Delays to train services can be classified as *primary* or *reactionary*. Primary delays are delays that are caused directly by an incident such as an asset failure, adverse weather, vandalism or trespass. Network Rail identify over 250 different causes of primary delay. As a result, reducing primary delays requires different strategies for different causes. This includes the design and maintenance of infrastructure, and the coordination of staff and emergency services.

Reactionary (or secondary) delays are delays that arise as a consequence of prior delays to different trains. Prior delays may prevent a train from using the required infrastructure, staff or rolling stock to deliver the service in a punctual manner. A

single initial primary delay can cause multiple reactionary delays, each of which can themselves cause further reactionary delays. As a result, reactionary delay tends to propagate through the railway network. In 2019, 64.35% of total train delay minutes were due to reactionary delays. It is therefore a very significant part of the overall punctuality problem.

1.1.2 Timetable Rescheduling

During delay incidents, railway services do not run as planned. The most immediate consequence of this is that it causes *conflicts* in the timetable. A conflict is a scenario in which two or more trains require the same piece of track at the same time, despite the fact that it can only accommodate at most one of them at once.

When conflicts arise, they must be resolved. Without intervention, the train that occupies the contested piece of track first will prevent the other trains in the conflict from occupying it at the required time. As a result, these trains may become more delayed. This is one of the basic mechanisms through which reactionary delay propagates.

However, the impact of conflicts on reactionary delay can be reduced by pre-emptively changing the previously planned timetable. This process is called *timetable rescheduling*. Timetable rescheduling happens in real-time, in response to an initial delay or set of delays. The result of timetable rescheduling is a new timetable that is free of conflicts. The geographical and temporal scope of the changes varies depending on the specific context.

It is helpful to name some specific types of timetable changes that can be made. Changing the times at which trains use parts of the track is called *retiming*. This can result in *reordering*, which is changing the order in which two or more trains use

a piece of track. Changing which parts of the track a train uses during its journey is called *rerouting*. This can include small changes to the routes taken within a station area, or changes to the stations visited. Within stations, rerouting also includes *replatforming*, which is changing the platform at which a train calls.

Other types of rescheduling, such as crew and rolling stock rescheduling, may be required in severe delay scenarios, also known as *disruptions*. However, in this thesis we are primarily concerned with less severe delay scenarios called *disturbances* that can be handled by timetable rescheduling.

1.1.3 The Train Timetable Rescheduling Problem

In some scenarios with traffic disturbances, there are a very large number of different possible ways to reschedule the timetable. There may be many trains to consider, each with many different possible routes and timings. Moreover, some rescheduled timetables are preferred over others because they lead to less reactionary delay, or a pattern of delays that is preferred over the alternatives. It is therefore natural to ask: *which rescheduled timetable is the best?*

The problem of calculating the 'best' rescheduled timetable is known as the Train Timetable Rescheduling Problem. This problem is the main subject of study in this thesis, and will be referred to by the abbreviation TTRP.

The definition of the TTRP given in the previous paragraph is intentionally vague. This is because there is no single agreed definition of the problem. Rather, certain features of the problem are specific to the context in which it is designed to be employed. For example, the geographical and temporal scope of the timetable depend on the location in which rescheduling is to be carried out. The constraints that the timetable must satisfy depend on the local safety, signalling and operational require-

ments. The types of rescheduling that are permitted depend on both policy and the geographical scope of the timetable. Finally, the interpretation of the word 'best' depends on the priorities and preferences of the organisation taking the rescheduling decisions. A precise description of the problem considered in this thesis will be given in Chapter 5.

A key feature of the TTRP is the scarcity of time available to solve it. By definition, timetable rescheduling must be performed in real-time as and when conflicts arise on the railway. If decisions about how to reschedule the timetable are not taken quickly, then preventable reactionary delays will occur and the state of the railway will have changed, making any decisions obsolete.

The large number of possible solutions makes the TTRP challenging for humans to solve in the time available. However, the TTRP can be addressed using optimisation techniques. In particular, it can be formulated as an optimisation problem, enabling the automated computation of rescheduled timetables. That is the focus of this thesis.

1.2 Motivation

The motivation for this thesis arises from the observation that although timetable rescheduling via optimisation has great potential to reduce delays, it has not yet been widely deployed on railways. One important reason for this is that inadequacies in existing optimisation modelling, solution and evaluation techniques act as barriers to deployment.

1.2.1 The Deployment Deficit

In Great Britain, timetable rescheduling is the responsibility of Network Rail as part of their role in signalling and traffic control. Although there is still a patchwork of different arrangements in existence throughout the country, most of the network is operated from ten Rail Operating Centres (ROCs). These bring together all aspects of railway operations within the regions they serve, including the management of delay incidents.

Most timetable rescheduling is currently carried out manually. Small disturbances are handled by signallers, whilst larger incidents are handled by traffic controllers. In reality, there is a strong communication between these two functions, and with representatives from Train Operating Companies (TOCs). Train diagrams, that depict the progress of trains along a railway line over time, are routinely used as a visual aid to rescheduling. Contingency plans are also used to respond to anticipated delay events. Although more sophisticated decision support software has been deployed in several areas, we do not know of any areas nationally where the TTRP is solved using optimisation.

This picture is changing. Network Rail has launched the Digital Railway programme with the aim of using digital technology to increase capacity on the network. One of the key technologies in this transition will be the Traffic Management System (TMS). The purpose of a TMS is to enable greater levels of automation in traffic control, including timetable rescheduling. In 2019, Network Rail announced a new TMS project for the East Coast Mainline — the research in this thesis has been communicated to the managers of this project.

The introduction of TMS technology is also occurring in various railway systems internationally. In fact, there are now several international examples of optimisation models for the TTRP being deployed. Lamorgese et al. [2018] describe examples of

deployment on railway lines in Italy, Latvia and Norway and an example of deployment in Roma Tiburtina station, in Rome. Deployment on the Stavanger-Moi line in Norway (see [Lamorgese and Mannino, 2015]) is especially exciting, since exact optimisation techniques were used, rather than heuristics.

1.2.2 Barriers to Deployment

There are many economic and organisational barriers to deployment for optimisation-based timetable rescheduling. As noted in the previous section, automated timetable rescheduling is largely dependent on the deployment of TMSs to provide the necessary real-time data and human-computer interfaces. TMSs are, by nature, extremely complex, expensive and safety-critical. This means that the acquisition of TMS technology typically involves years of planning, procurement, design and operational change on behalf of the infrastructure manager.

However, this is not the only barrier to deployment of optimisation-based timetable rescheduling. The optimisation techniques available for modelling, solving and evaluating the TTRP are still inadequate in many ways. Although there are now a very large number of publications proposing different models and solution methods for solving different versions of the TTRP, there are some systematic weaknesses with the current literature. Many of these weaknesses are reviewed and addressed by the work in this thesis.

1.3 Research Aim

The overall aim of this thesis is:

> To develop new optimisation modelling, solution and evaluation tech-

niques for the TTRP that can help overcome the barriers to deployment that exist for optimisation-based rescheduling.

The scope of the thesis is limited to the the application of the TTRP to areas around large railway stations. Railway lines, high-speed lines and metro systems are not considered, although the techniques developed might also be applicable to these settings. This restriction is appropriate since large stations are the key traffic bottlenecks in many parts of Great Britain.

1.4 Significance and Impact

This PhD has been carried out in collaboration with Network Rail. The collaboration has been immensely important to the research in several respects. First, insights gained from conversations with various people in the organisation have informed my understanding of the barriers to implementation for TTRP technology. This has been invaluable in framing the research questions and guiding the overall direction of the thesis. Second, access to technical information has been very important, particularly relating to railway signalling. Finally, access to data has made it possible to test the methods developed using instances based on real data.

As well as greatly informing the research, the collaboration with Network Rail has provided an opportunity for achieving impact. In particular, it has facilitated knowledge transfer in both directions between the railway industry and optimisation researchers. In addition to this thesis, the many conversations, meetings and presentations that have occurred during its preparation have helped to disseminate knowledge. This is true of both the existing state-of-the-art in TTRP research, and the original contributions made in this thesis.

Part of the impact has also been to elucidate the benefits of optimisation-based

timetable rescheduling, and the use of optimisation in other railway planning problems. Optimisation techniques for the TTRP are not generally familiar even to specialist modellers in the railway industry, let alone managers. Experience suggests that there is more familiarity with other decision support methodologies such as simulation and artificial intelligence. This PhD has provided the opportunities to remedy this situation by promoting optimisation as a technology within Network Rail. This is especially important for planning problems with large, combinatorial solution spaces, such as the TTRP.

One particular opportunity for impact that has arisen is the TMS project on the northern English part of the East Coast Main Line. We have been able to disseminate the research in this thesis to people involved. Since the project is in its early stages, we have encouraged them to consider whether techniques developed in this thesis could be incorporated into the TMS.

The thesis has a wider significance beyond its practical impact. This relates to the original modelling and methodological contributions made in each chapter. These span the areas of modelling, solution techniques and model evaluation.

1.5 Overview

The remainder of this thesis begins with Chapter 2, which introduces the optimisation techniques that are used in subsequent chapters. A broad introduction to the use of optimisation in railway planning is then given in Chapter 3. Chapter 4 provides a more tightly focused review of the different types of optimisation models that have been used for the TTRP.

Each of Chapters 5, 6 and 7 identifies a distinct barrier to deployment for optimisation-based rescheduling in a separate literature review and proposes novel modelling,

solution or evaluation techniques to overcome it:

- Chapter 5 addresses the lack of existing TTRP models that can be solved to optimality for realistically sized TTRP instances in sufficiently short times. It is entitled

  > *A multicommodity flow model for rerouting and retiming trains in real-time to reduce reactionary delay in complex station areas.*

  Its focus is developing a new model and solution method for the TTRP. In many ways, this chapter forms the cornerstone of the thesis.

- Chapter 6 addresses the need to be able to accurately model the speed of trains within TTRP models. It is entitled:

  > *A data-driven, variable-speed model for the train timetable rescheduling problem.*

  This is a significant extension of the model presented in Chapter 5 that improves the way in which speed is modelled.

- Chapter 7 considers the lack of evidence on the likely impact of TTRP models on fairness between railway operators. It is entitled:

  > *An evaluation of the fairness of railway timetable rescheduling in the presence of competition between train operators.*

  The chapter develops a framework for evaluating fairness between Train Operating Companies, and applies this to analyse the fairness of model presented in Chapter 5 using a case study.

Collectively, Chapters 5 and 6 address the lack of experimental evidence using real data on timetable rescheduling within large stations in the British railway network. Results are presented from two new sets of instances that have been created for Don-

caster Station and Derby station, respectively. The final part of the thesis, Chapter 8, draws overall conclusions and includes some suggestions for further research.

This thesis has an interdisciplinary flavour. Whilst the principal methodology used is Optimisation, an area of Operational Research, the application area is in railway technology which is traditionally in the domain of Civil Engineering. On top of this, Chapter 6 incorporates methodology from Statistics, whilst Chapter 7 uses ideas from Economics. This serendipitous blend is a testament to the range of challenges presented by the TTRP.

Chapter 2

Optimisation Background

In this thesis, we model the TTRP as an optimisation problem. This involves designing a set X and a function $f : X \to \mathbb{R}$ such that the problem can be stated mathematically as:

Find a solution $x_0 \in X$ such that $f(x) \geq f(x_0)$ for all $x \in X$.

Such a formulation allows the problem to be tackled with computational techniques from the mature field of Optimisation. The set X is referred to as the *feasible set*, and the function f as the *objective function*. The feasible set should reflect the constraints of the problem accurately, whilst the objective function should capture, as far as possible, the motivations of the decision makers.

2.1 Mixed Integer Programming

The TTRP can be modelled as a *Mixed Integer Linear Program* (MILP) (see Chen et al. [2010]). This means that the problem takes the form $\min_{x \in X} f(x)$ where

$$f(x) = c^T x \text{ and } X = \{x \in \mathbb{Z}_+^l \times \mathbb{R}_+^{n-l} : Ax \geq b\}. \tag{2.1}$$

Note that in (2.1), $m, n \in \mathbb{Z}_+$, $c \in \mathbb{Q}^n$ and $A \in \mathbb{Q}^{m \times n}$. In other words, a MILP is a problem in n non-negative variables $x_1, \ldots, x_n$ (of which the first l must take integer values) with linear constraints and a linear objective function. A set in the form X is referred to as a *mixed-integer set*. The models proposed in this thesis are actually *Binary Integer Linear Programs* (BILPs), which are special cases of MILPs in which X takes the form

$$X = \{x \in \{0,1\}^n : Ax \geq b\}. \tag{2.2}$$

However, we discuss MILP solution techniques because they are also applicable to our models. MILPs are a very well-studied problem class, for which an enormous arsenal of computational techniques and software are available. It is also an extremely flexible problem class that can be used to model a very wide range of problems.

A general approach for solving MILPs is *LP-based branch-and-bound* (LP B&B) [Land and Doig, 1960]. This is an algorithm that solves a MILP by solving a sequence of related *Linear Programs* (LPs) in order to obtain upper and lower bounds on the optimal solution of the MILP. A Linear Program is a MILP that has only real-valued variables, and no integer variables (i.e. $l = 0$). Whilst LP modelling is less flexible than MILP modelling (LPs cannot model integer variables), fast algorithms for solving LPs exist (see Section 2.2). LP B&B exploits the ease of solution of LPs to solve MILPs.

LP B&B consists of two steps: *branching* and *bounding*. First, we describe bounding,

the process of obtaining bounds on the optimal value of a MILP by solving an LP. Suppose we have a MILP described by f and X with unknown optimal objective value $z^* = \min_{x \in X} f(x)$. The *LP relaxation* of this MILP is the LP formed from the MILP by setting $l = 0$. Denoting the feasible set of this LP relaxation by X^{LP}, suppose that the LP relaxation is solved, yielding an optimal solution x^{LP} with optimal objective value $z^{LP} = f(x^{LP})$. Given that $X \subseteq X^{LP}$, we know that z^{LP} is a lower bound for z^*. If, by chance, $x^{LP} \in X$ then z^{LP} is also an upper bound on z^*. In this situation, we have that $z^* = z^{LP}$ and that x^{LP} is an optimal solution to the original MILP.

In cases where $x^{LP} \notin X$, branching is performed. Branching creates two new MILPs (1) $\min_{x \in X_1} f(x)$ and (2) $\min_{x \in X_2} f(x)$ that can be used to learn about bounds for the original MILP. The new feasible sets X_1 and X_2 must be Mixed Integer Linear sets (i.e. valid feasible sets for MILPs), and their union $X_1 \cup X_2$ must be equal to X. Typically $X_1 \cap X_2 = \emptyset$, making X_1 and X_2 a partition of X, but this is not strictly necessary. Furthermore, the LP relaxations of X_1 and X_2 must not contain x^{LP}. Without this last property, bounding (1) and (2) would produce identical bounds to those obtained by bounding the original MILP. It is hoped that a well defined branch will lead to (1) and (2) producing better (higher) lower bounds. This is because the smallest lower bound out of those produced by (1) and (2) is also a lower bound for the original MILP. In addition, if either (1) or (2) produces an upper bound via the discovery of a feasible solution, then this is also feasible for the original MILP and hence and upper bound for the original MILP.

LP-based branch and bound is a recursive combination of the branching and bounding steps outlined above. The relationships between MILPs created by branching can be encoded in a *branch-and-bound tree*. This is a connected acyclic graph with a node for each MILP. The presence of an arc between two nodes indicates that one MILP was obtained from another by branching. The tree is recursively grown by

bounding new nodes, updating the bounds for all other previously bounded nodes, and branching if appropriate. A node can be disregarded without being branched on if

1. The feasible set is empty.
2. The upper and lower bounds for that node are equal.
3. The lower bound for the node is higher than the upper bound for the root node.

In the second case, the optimal solution to this node is already known, and in cases 1 and 3, the problem at this node cannot possibly yield an optimal solution to the MILP. Disregarding these nodes is very important, since it reduces the space that needs to be searched for an optimal solution. The algorithm terminates with an optimal solution when the smallest upper bound and largest lower bound found so far for the root node (the original MILP) are equal.

The description provided of LP-based branch-and-bound includes only the most basic principles. High quality algorithms for branching and for choosing which node to bound next are essential. Moreover, the good performance of branch-and-bound on many problems is contingent on extra steps in the algorithm such as the use of cutting planes, preprocessing and heuristics. Each of these are large topics that are not summarised here.

2.2 Linear Programming

As highlighted in Section 2.1, solving a MILP by LP B&B requires the solution of a sequence of Linear Programs (LPs). A comprehensive introduction to LPs is given

by Vanderbei [2014]. Any LP can be written in the form

$$(P) \quad \min\{c^T x : Ax = b, x \in \mathbb{R}^n_+\}, \tag{2.3}$$

where $A \in \mathbb{R}^{n\times m}$ and $b^T \in \mathbb{R}^m$ i.e. there are m constraints with $m \leq n$. When solved as part of LP B&B, LPs are often solved using the *simplex algorithm*. We describe the simplex algorithm in some detail in order to motivate column generation, which is described in Section 2.3. However, it should be noted that there are several other algorithms for solving LPs.

The simplex algorithm divides the n variables into m *basic variables* $x_B \in \mathbb{R}^m_+$ and $(n-m)$ *non-basic variables* $x_N \in \mathbb{R}^{n-m}_+$ so that when permuted appropriately, the variable vector can be written $x = (x_B, x_N)$. Using this new notation, the problem can be written as

$$(P) \quad \min\{f = c_N^T x_N + c_B^T x_B : A_N x_N + A_B x_B = b, x_N \in \mathbb{R}^m_+, x_B \in \mathbb{R}^{n-m}_+\}. \tag{2.4}$$

Assuming that A_B is invertible, we can calculate that

$$x_B = A_B^{-1} b - A_B^{-1} A_N x_N \quad \text{and} \quad f = c_B^T A_B^{-1} b + (c_N^T - c_B^T A_B^{-1} A_N) x_N. \tag{2.5}$$

Letting $x_N = 0$ yields a *basic solution* $x = (A_B^{-1}b, 0)$ with objective value $f = c_B^T A_B^{-1} b$. The same procedure can be applied with any choice of basis to obtain a basic solution. If the values of $x_B = A_B^{-1}b$ in a basic solution satisfy $x_B \geq 0$, then $x = (x_B, 0)$ is also a feasible solution to the LP, called a *basic feasible solution* (BFS). Some choices of basis give rise to a BFS, whilst others produce a basic solution that is not feasible.

The feasible set $\{x \in \mathbb{R}^n_+ : Ax = b\}$ of an LP is a convex polytope. Provided the optimal objective value is finite, it can be shown that one of the vertices of

this polytope is an optimal solution to the LP. The simplex algorithm works by iteratively moving from one vertex to an adjacent vertex with a better objective value, terminating when no adjacent vertex offers a better value. Since the polytope is convex, this final vertex corresponds to a globally optimal solution.

It can be shown that each vertex of the feasible polytope corresponds to a BFS. As a result, moving from one vertex to another can be achieved by changing the basis x_B and calculating the corresponding BFS. To obtain an adjacent BFS, it is necessary (but not always sufficient) to change the basis by entering a previously non-basic variable into the basis, causing a previously basic variable to leave the basis. Equation (2.5) shows that entering the i^{th} non-basic variable will lead to an improvement (decrease) in objective value only if the i^{th} component of the vector $\bar{c}_N := c_N^T - c_B^T A_B^{-1} A_N$ is negative. This implies that the incumbent BFS is optimal for the LP if and only if $\bar{c}_N \geq 0$. If it is not optimal, then entering a variable $(x_N)_i$, for which $(\bar{c}_N)_i < 0$, into the basis will yield a better BFS. The vector $\bar{c}_N$ is called the *reduced cost*.

Every LP (P) in the form of (2.3) has a *dual* LP, which can be written

$$(D) \quad \max\{b^T y : A^T y \leq c, y \geq 0\}. \tag{2.6}$$

If x_B is an optimal basis for (P), then $\pi = c_B^T A_B^{-1}$ is an optimal solution to (D). This is called the *optimal dual solution* for (P), and plays an important role. In particular, note that the reduced cost can be rewritten $\bar{c}_N := c_N^T - \pi A_N$. The dual value vector π is crucial for checking the optimality condition $\bar{c}_N \geq 0$ and deciding which variable should enter the basis.

2.3 Column Generation

Suppose that we have an LP written in the form

$$(MP) \quad \min \left\{ \sum_{j \in J} c_j \lambda_j : \sum_{j \in J} a_{ij} \lambda_j \geq b_i \quad \forall i = 1, \ldots, m, \lambda_j \geq 0 \quad \forall j \in J \right\}, \quad (2.7)$$

where the number of variables $|J|$ is very large in comparison to the number of constraints m. We will refer to this problem the *master problem* (MP). Since any basis for this LP is of size m, an optimal basic feasible solution contains at most m non-zero variables. Therefore, most of the variables in any optimal solution are non-basic and take the value zero.

Suppose that an optimal basis B for (MP) is known and that $J' \subset J$ is a subset of the variables containing all the variables in B. Then solving the *restricted master problem* (RMP), given by

$$(RMP) \quad \min \left\{ \sum_{j \in J'} c_j \lambda_j : \sum_{j \in J'} a_{ij} \lambda_j \geq b_i \quad \forall i = 1, \ldots, m, \lambda_j \geq 0 \quad \forall j \in J' \right\}, \quad (2.8)$$

would yield a solution that can be extended to an optimal solution to (MP) by letting all of the variables $j \in J \setminus J'$ take the value zero. This is because the BFS $x = (A_B^{-1} b, 0)$ corresponding to basis B doesn't depend on the data for the non-basic variables. In other words, the values of a_{ij} and c_j for $j \in J \setminus J'$ are not required. To summarise, we have shown that if an optimal basis B to (MP) is known, then an optimal solution to (MP) can be obtained by solving (RMP). This idea is appealing because (RMP) can be solved more quickly than (MP) using the simplex algorithm since it has fewer variables.

Of course, an optimal basis B for (MP) is not known in advance of solving (MP). The idea of column generation is to build one iteratively. Given an initial set $J' \subset J$,

solving (RMP) yields optimal primal and dual solutions, λ^* and π^*. Since π^* doesn't depend on any of the data for non-basic variables (recall that $\pi = c_B^T A_B^{-1}$), it is unaffected by the omission of the variables $J \setminus J'$ in (RMP). We can therefore use the standard simplex optimality condition, $\bar{c}_N = c_N^T - \pi^* A_N \geq 0$, to test if λ^* is optimal for (MP). Any non-basic variables in J' have non-negative reduced cost by the optimality of λ^* for (RMP), so only variables currently outside the model need to be checked. The reduced cost for a variable $j \in J \setminus J'$ is

$$\bar{c}_j = c_j - \pi^* A_j, \tag{2.9}$$

where $A_j = (a_{1j}, \ldots, a_{mj})^T$ is a column vector containing data for variable j. Calculating $\bar{c}_j$ for each $j \in J \setminus J'$ could still be a very time consuming exercise, given that $|J|$ is very large. We therefore solve the following optimisation problem, known as the *subproblem*:

$$(SP) \quad \min_{j \in J} \{c_j - \pi A_j\}. \tag{2.10}$$

If the optimal value of SP is non-negative, then $\bar{c}_N \geq 0$ and λ^* is optimal for (MP). Otherwise, the variable λ_j corresponding to the optimal j for (SP) has a negative reduced cost and can profitably be added to (RMP) in order to enter the basis. In other words, λ_j is added to J' and the whole process is repeated until an optimal solution to (MP) is obtained.

When column generation is used simply as an algorithm for solving LPs with many variables, in the manner described, its benefits over the simplex algorithm are entirely computational. It facilitates the solution of (MP), an LP with $|J|$ variables, by solving a sequence of smaller optimisation problems. These smaller problems consist of (RMP), which is an LP with $|J'| < |J|$ variables, and (SP), which is a search over a set of size $|J|$. Column generation is most useful when J has a special structure that makes (SP) efficiently solvable.

2.4 Dantzig-Wolfe Reformulation

We showed in the previous section that column generation can be an effective way to solve LPs with a very large number of variables. In this section, it is shown how this can be very useful for solving certain types of large MILPs.

Consider a problem of the form

$$\begin{aligned} \min \quad & \sum_{k=1}^{K} c^{kT} x^k \\ \text{s.t.} \quad & \sum_{k=1}^{k} A^k x^k \geq b \\ & D^k x^k \geq d^k \quad \forall k = 1, \ldots, K \\ & x^k \in \{0,1\}^{n_k} \quad \forall k = 1, \ldots, K. \end{aligned} \tag{2.11}$$

In this problem, the variables are divided into K groups and each group $k = 1, \ldots, K$ consists of n_k variables that are subject to their own constraint $D^k x^k \geq d^k$. This is often referred to as *block-diagonal* structure, since if the constraints are amalgamated into the single linear system

$$\left[\begin{array}{ccc|c} A^1 & \ldots & A^K & \\ \hline D^1 & & & \\ & \ddots & & \\ & & D^K & \end{array}\right] \left[\begin{array}{c} x^1 \\ \vdots \\ x^K \end{array}\right] = \left[\begin{array}{c} b \\ D^1 \\ \vdots \\ D^K \end{array}\right] \tag{2.12}$$

then the matrix on the left hand side is bordered block diagonal. We can think of this problem as being a group of k smaller problems

$$\min\{c_k^T x^k : D^k x^k \geq d^k, x^k \in \{0,1\}^{n_k}\} \tag{2.13}$$

that are linked together only by the constraints $\sum_{k=1}^{k} A^k x^k \geq b$, which are accord-

ingly called *linking constraints*. This is a commonly observed problem structure across a wide range of optimisation applications because it arises naturally from problems that involve coordinating a group separate but linked entities or organisational units. In this thesis, those entities are trains.

Danzig-Wolfe reformulation is a very well established technique that was introduced by Dantzig and Wolfe [1960]. The idea is to perform a variable transformation that changes the problem into one that can be usefully tackled using column generation. This variable transformation is applied separately within each of the K sets

$$X^k = \{x^k \in \{0,1\}^{n_k} : D^k x^k \geq d^k\} \tag{2.14}$$

that make up the feasible set of the problem. In the general case where $x^k \in \mathbb{R}_+^{n_k}$ (as opposed to $\{0,1\}^{n_k}$ as we have written above), X^k is a polyhedron and this variable transformation is performed by applying the Minkowski-Weyl Theorem (see Theorem 1 of [Vanderbeck and Wolsey, 2010, p. 3]). This Theorem states that any polyhedron can be represented as a convex combination of its extreme points and a linear combination of its extreme rays. However, for the purposes of this thesis we will always have $x^k \in \{0,1\}^{n_k}$, and this makes the transformation significantly simpler. Let P^k be the finite index set of X^k so that $X^k = \{x^{k,p} : p \in P^k\}$, where each $x^{k,p}$ is a (constant) member of the set X^k. Then we can use the variable transformation defined by

$$x^k = \sum_{p \in P^k} \lambda^{k,p} x^{k,p}, \quad \sum_{p \in P^k} \lambda^{k,p} = 1, \quad \lambda^{k,p} \in \{0,1\} \quad \forall p \in P^k. \tag{2.15}$$

The new binary variables $\lambda^{k,p}$ are indicators, in the sense that for each possible value $x^{k,p}$ of x^k, we have that $x^k = x^{k,p} \iff \lambda^{k,p} = 1$. The resulting Danzig-Wolfe

reformulation of Problem (2.11) is

$$\begin{aligned} \min \quad & \sum_{k=1}^{K} \sum_{p \in P^k} c^{kT} x^{k,p} \lambda^{k,p} \\ \text{s.t.} \quad & \sum_{k=1}^{k} \sum_{p \in P^k} A^k x^{k,p} \lambda^{k,p} \geq b \\ & \sum_{p \in P^k} \lambda^{k,p} = 1 \quad \forall k = 1, \ldots, K \\ & \lambda^{k,p} \in \{0,1\} \quad \forall k = 1, \ldots, K \quad \forall p \in P^k. \end{aligned} \tag{2.16}$$

Note that the constraints $D^k x^k \geq d^k$ have been dropped. This is because, by definition, the points $x^{k,p}$ all implicitly satisfy these constraints.

There are two potential advantages to performing this transformation. The first is that because the constraints $D^k x^k \geq d^k$ have been dropped and only K constraints added, the new formulation potentially has many fewer constraints.

The second potential advantage of using a Danzig-Wolfe reformulation is related to the quality of LP relaxation bounds. Let

$$\begin{aligned} x &= (x^1, \ldots, x^K), \\ n &= \sum_{k=1}^{K} n_k, \\ Q^A &= \{x \in \{0,1\}^n : \sum_{k=1}^{k} A^k x^k \geq b\}, \\ \text{and } Q^k &= \{x \in \{0,1\}^n : D^k x^k \geq d^k\}. \end{aligned} \tag{2.17}$$

Then the feasible set of the original problem can be written as $Q^A \cap \bigcap_{k=1}^{K} Q^k$ and the feasible set of the reformulated problem is $Q^A \cap \bigcap_{k=1}^{K} conv\left(Q^k\right)$ where $conv\left(Q^k\right)$ is the convex hull of Q^k. Although these sets are equal, the LP relaxation of the latter is a subset of the LP relaxation of the former. As a result, the reformulation sometimes

yields stronger lower bounds, which can be very advantageous when solving the problem with branch-and-bound. Unfortunately, this advantage fails to materialise when the LP relaxation of Q^k is equal to $conv\left(Q^k\right)$.

2.5 Branch-and-price

The most obvious disadvantage of transforming problem (2.11) into (2.16) via Danzig-Wolfe reformulation is that the latter has many more variables ($\sum_{k=1}^{k} |P^k|$ as opposed to the original $\sum_{k=1}^{k} n_k$). However, the vast majority of these variables can be expected to take value 0 in any optimal solution to the LP relaxation. This means that the LP relaxation of (2.16) is ideal for the application of column generation.

A *branch-and-price* algorithm is an LP-based branch-and-bound algorithm in which the LP at every node of the branch-and-bound tree is solved using column generation [Barnhart et al., 2001]. In the case of problem (2.16), there are K separate column generation subproblems, where subproblem k is given by

$$(SP^k) \quad \min_{p \in P^k} \{(c^{kT} - \pi A^k) x^{k,p}\}. \tag{2.18}$$

Provided there is an efficient way to search over P^k, this approach can be very successful. Typically, this means that (SP) should reduce to a well-studied combinatorial problem such as a shortest path problem or a knapsack problem.

The most challenging aspect of designing a branch-and-price algorithm is branching. Branching on a fractional variable $\lambda^{k,p}$ of (RMP) by setting $\lambda^{k,p} = 0$ in one branch and $\lambda^{k,p} = 1$ in the other is usually unsuccessful for two reasons. First, it often fails to lead to improved lower or upper bounds in the child nodes. The $(\lambda^{k,p} = 0)$ branch has ruled out just a single solution $x^{k,p} \in X^k$ and so is likely to produce

exactly the same lower bound. On the other hand, the ($\lambda^{k,p} = 1$) branch completely decides the solution for entity k. Unless its objective value $c^{kT}x^{k,p}$ happens to be good, this branch is unlikely to be worth branching on. The second problem with this branching strategy is that it is difficult to enforce the branch $\lambda^{k,p} = 0$. Unless this constraint is added to the subproblem (SP^k), the variable $\lambda^{k,p}$ will simply be generated and added to the problem again. However, adding constraint $\lambda^{k,p} = 0$ to (SP^k) typically compromises the structure of the problem, and prevents it from being solved efficiently.

Instead, it is better to design a branching rule that is *compatible* with the subproblem. A compatible branching rule is one that can be enforced using constraints in the subproblem that do not compromise the structure that allows it to be solved efficiently. These branching rules are often expressed in terms of the original variables x_i^k, since these are the variables used in the subproblem. We design a branching rule of this type in Section 5.6.5 for solving the TTRP.

2.6 Multicommodity Flow and Time-space Graphs

In this thesis, a branch-and-price algorithm is used to solve a MILP that is a variant of the *multicommodity flow problem* (MCFP). The MCFP is a canonical optimisation problem defined on a directed graph $G = (N, A)$ that involves finding a minimum cost flow for each of several *commodities* $k = 1, \ldots, K$, between source and sink nodes (that may be different for each commodity). The MCFP has been used in many different application areas as diverse as communication networks and distribution systems [Ahuja et al., 1993, p. 654], as well as transport applications such as the one considered in this thesis.

Letting x_{ij}^k be the amount of flow of commodity k along each arc $(i, j) \in A$, the

MCFP can be formulated as a Linear Program:

$$\begin{aligned}
\min \quad & \sum_{k=1}^{K} \sum_{(i,j)\in A} c_{ij}^k x_{ij}^k & & \\
\text{s.t.} \quad & \sum_{k=1}^{k} x_{ij}^k \leq u_{ij} & & (i,j) \in A \\
& \sum_{(i,j)\in A} x_{ij}^k - \sum_{(j,i)\in A} x_{ji}^k = b_i^k & & i \in N \quad k \in \{1,\ldots,K\} \\
& 0 \leq x_{ij}^k \leq u_{ij}^k & & (i,j) \in A \quad k \in \{1,\ldots,K\}.
\end{aligned} \tag{2.19}$$

In this formulation, c_{ij}^k is the unit cost of flow of commodity k along arc (i,j) and u_{ij}^k is an upper bound on that flow. The total amount of flow of all commodities along (i,j) is constrained to be less than u_{ij}. Finally, the values of b_i^k represent the supply of commodity k at node i. At the source nodes of commodity k this takes a positive value, at sink nodes it takes a negative value, and at all other nodes it is equal to zero meaning that flow is conserved.

The MCFP formulation presented has a special structure that can be exploited by effective specialised algorithms (see [Ahuja et al., 1993, p. 649-694]). However, in the problems solved in this thesis, the flows along each arc are required to take values in $\{0,1\}$, so that they represent paths from a source node to a sink node for each commodity. This makes the problem significantly more difficult in both theory and practice. We also introduce capacity constraints that are more complex than the simple arc capacities in the standard MCFP.

Variants of the MCFP are particularly well suited to Dantzig-Wolfe reformulation and branch-and-price solution algorithms. For each commodity, the constraints ensuring a valid path from source to sink define a subproblem that is efficiently solvable using shortest paths algorithms. Meanwhile, the arc capacities (and any side constraints) act as linking constraints that are retained in the master problem. Barnhart et al.

[2000] provides a good introduction to the use of branch-and-price to solve the integer MCFP.

In this thesis, the MCFP is defined on a *time-space graph* (also variously called a *time-space network*, or a *time-expanded network*). Time-space graphs are used for modelling *time dependent* problems, in which decisions are to be made about the time at which certain events occur. They are obtained by expanding graphs in which each node represents one of these events. To obtain a time-space graph, the planning horizon is first partitioned into discrete periods of time, called *time intervals*. Each node in the original graph is then duplicated once for each time interval. Arcs in a time-space graph describe possible transitions from one event in a particular time interval to a different event in a different time interval. Arcs therefore implicitly encode the number of time intervals required for each event.

Time-space graphs have a long history in optimisation, the early part of which is recorded by Aronson [1989]. They are ubiquitous in transport applications of optimisation, since these problems are very often time-dependent. They have been widely used in TTRP modelling, for example by Caprara et al. [2002], Bettinelli et al. [2017], Zhou et al. [2017] and Binder et al. [2017].

2.7 Summary

Several solution methods were introduced in this chapter to provide the necessary background for the remainder of the thesis. The material presented is particularly relevant to Chapter 5, which introduces a multicommodity flow formulation for the TTRP that is based on a time-space graph. Enhancements are made in Chapter 5 to the basic column generation algorithm outlined here. In addition, a problem-specific branching rule is proposed. These solution methods are then used throughout the

thesis.

Chapter 3

Optimisation for Railway Planning

The successful operation of a railway system, like any large system, requires extensive planning. Planning is carried out by many different organisations with differing and complimentary responsibilities, including infrastructure managers, train operators, industry regulators and government. Everything that is required for a passenger service to run successfully must be planned: the track, timetable, rolling stock, train crews, stations, safety systems and much else besides. Optimisation can be used most effectively as a decision aid during planning when there are too many different options for a human to compare them all. This situation tends to arise in parts of the planning process that involve scheduling the utilisation of resources over time.

Optimisation research for railway planning begins by identifying a *planning problem*. Although there are almost as many problem definitions as there are researchers, they can be grouped into general categories. For example, the TTRP refers to the general problem of timetable rescheduling in response to delays, despite the fact that the type of railway, severity of delays and possible rescheduling outcomes

might vary considerably depending on the particular application. Many aspects of railway planning problems have now been studied using optimisation, and have abundant and growing bodies of academic literature dedicated to them. A significant part of the challenge lies in modelling: identifying the constraints, stakeholders, objectives, uncertainties and dependencies of each problem. However, developing effective solution methods for the models proposed is an equally pressing concern. Despite major advances in optimisation solution methods over the last few decades, railway planning problems continue to provide a rich source of problems that cannot easily be solved in the time that is available in practice to solve them. They tend to be characterised by complexity and large scale.

The literature on optimisation for railway planning is too large to review here in detail. However, some key concepts are introduced, and the TTRP is situated in relation to similar planning problems. For a more in-depth account, the reader is referred to two excellent books that bring together surveys from across the spectrum of railway planning, [Borndörfer et al., 2018] and [Hansen and Pachl, 2014].

3.1 Planning Horizons

The *planning horizon* (alternatively *time horizon*) of a problem is the period of time with which the planning is concerned. Railway planning problems can be classified according to the length of the planning horizon, into the following groups:

- *Strategic* problems have the longest planning horizon, as they consider decisions that will continue to have consequences many years into the future. An example is the problem of designing the layout of the track network. Because they are usually related to the provision of railway services a long way into the future, uncertainty about future passenger demand can make strategic prob-

lems difficult to tackle. They also tend to be characterised by having a large number of stakeholders and objectives, because strategic decisions often involve significant investment in infrastructure. Where governments are involved these decisions can become political, which makes the application of optimisation difficult.

- *Tactical* problems are those which must be addressed weeks or months in advance of their planned effect. A good example is the Train Timetabling Problem. Tactical problems are often logistical problems related to allocating resources to be used by train services. They are therefore influenced by the amount of resource availability planned for at the strategic planning stage. Tactical problems are a significant success area for optimisation. They are often well defined problems in which solutions obtained via optimisation can be evaluated before implementation.

- *Operational* problems are those that are faced day-to-day in order to adjust or supplement decisions taken at the tactical level. The TTRP is an operational problem, as are the crew and rolling stock rescheduling problems. Operational problems have short planning horizons because they depend on information that arises in real-time and is therefore subject to change, such as the current status of the railway system. The implementation of optimisation models for operational decisions faces several unique hurdles. The short computation times required to make quick decisions is a challenge for optimisation methods. We address this challenge for the TTRP in Chapter 5 by producing an algorithm with sufficiently short computation times. The fast nature of operational decisions also means that solutions cannot always be extensively evaluated before implementation. It is therefore important that optimisation models for operational problems reliably produce solutions that can be implemented in practice. This motivates the very detailed modelling of both track

capacity in Chapter 5, and train speed in Chapter 6. Further, models need to be integrated into the real-time operations of the railway. They therefore need to be extremely safe and reliable, and have well-functioning human-computer interfaces.

Operational railway planning problems such as the TTRP usually adopt a very short time horizon of one hour up to a few hours. This is partly because longer time horizons increase the difficulty of solving the problem in the short time available. However, it is also a response to the uncertainty in the future evolution of traffic conditions. Corman and Meng [2015] outline different approaches to the question of when a problem should be solved. In *open loop control* settings, problems are solved only once using forecasts for the times of future events. On the other hand, in *closed loop control* a problem is solved repeatedly, taking into account the latest available data and forecasts about the traffic situation each time. The latter approach allows for current plans to be as up-to-date and accurate as possible. However, this must be weighed against the difficulties of implementing a solution that is frequently changing.

3.2 Railway Infrastructure Representation

Planning problems that schedule the utilisation of railway infrastructure over time require an appropriate representation of that infrastructure. It is almost universal practice to use directed graphs for this purpose. Infrastructure is viewed as a set of discrete *track components* which are the nodes of the graph, whilst directed arcs indicate possible transitions between these components. Data is usually associated with each node (or each arc, or both), although the nature of this data depends on the application in question.

Different infrastructure representations arise from different choices of what the track components are. *Microscopic* representations contain the highest level of detail because they use the smallest track components. The track components can be *block sections* or *track circuits* (see Section 5.2). *Macroscopic* representations, by contrast, use much larger track components that aggregate the information available in microscopic models. Typically, the track components are stations and junctions. Macroscopic representations are generally preferred for planning tasks with a wide geographical scope, because they result in smaller graphs. However, the detail included in microscopic representations can be crucial depending on the planning task. A more comprehensive introduction to this topic is given by Radtke [2014].

3.3 Timetabling and Timetable Rescheduling

One of the most studied tactical railway planning problems is the Train Timetabling Problem. The aim of this is to construct a timetable by deciding on the arrival and departure times of trains at station calling points. The timetable must satisfy the desired level of service frequency on modelled lines, whilst respecting the capacity of the railway network. The Train Timetabling Problem has received considerable attention in the literature, and several reviews are available [Caprara et al., 2010; Cacchiani et al., 2015].

Once a timetable has been proposed, the Train Routing Problem is normally solved. Whereas timetables are usually produced at macroscopic level, the train routing problem involves finding paths through the track infrastructure for each train at the microscopic level, so that its arrival and departure times match those in the timetable. This is often only necessary within bottlenecks such as junctions or stations. In many stations, allocating trains to platforms determines their route and so the problem

reduces to the Train Platforming Problem, which is to assign each train a platform. Some authors refer to the routing and platforming problems interchangeably. A review of models for the train routing problem is given by Lusby et al. [2011].

The Train Timetabling Problem and the Train Routing Problem are similar to the TTRP. Indeed, they are the tactical, 'offline' equivalent of the TTRP. As a result, there are many similarities between the models that have been used for both problems. Furthermore, timetable scheduling and rescheduling can be seen as playing complementary roles in the prevention of delays. The impact of unexpected delays can be mitigated by designing robustness into the timetable at the tactical planning stage (see [Lusby et al., 2018] for a review). When delays inevitably occur to which the timetable is not sufficiently robust, timetable rescheduling can then be used to recover.

However, there are several important differences between scheduling and rescheduling that arise from practical considerations. One difference is that at the rescheduling phase, the original schedule and the past and present location of the trains are already known, while they are not at the scheduling phase. Another difference is that at the scheduling phase the objective is to design a timetable that is attractive to passengers and operators, whereas at the rescheduling phase there is more emphasis on reducing delays and recovering the originally planned timetable. A third difference is that periodic timetables (e.g. timetables that repeat every hour) are often sought at the scheduling phase, but this is inappropriate for rescheduling. This means that popular models for timetabling such as the Periodic Event Scheduling Problem (PESP) [Serafini and Ukovich, 1989] cannot be used for timetable rescheduling. Finally, significantly more time is available to solve timetabling and routing problems than rescheduling problems, because the former are tactical problems, rather than operational.

3.4 Timetable, Crew and Rolling Stock Rescheduling

In the presence of small disturbances, timetable rescheduling is the main focus of the response. However, large disruptions can also result in the crew and rolling stock for a subsequent train service being unavailable or in the wrong location. The response to these major disruptions therefore requires the rescheduling of all three components: the timetable, crew schedules and rolling stock schedules.

These three rescheduling problems are heavily interdependent. The process is typically sequential: The timetable is rescheduled first by the infrastructure manager, and the operators respond by rescheduling rolling stock, then crews. Whilst this approach might help to keep the problems tractable, changes to the timetable might have very undesirable implications for rolling stock or crew. This has led some researchers to try to integrate these problems (see Cacchiani et al. [2014] for a review). In general, such models are very large and complex, and unable to consider all aspects of all problems in sufficient detail. Moreover in many railway systems, responsibility for timetable rescheduling lies with the infrastructure manager, whilst responsibility for crew and rolling stock lies with operators. The problems therefore involve separate decision makers and are not suitable for integration.

The Rolling Stock Rescheduling Problem is usually carried out after timetable rescheduling. The aim is to ensure that trains of the correct capacity and type are available in the right place at the right time to provide the services in the rescheduled timetable. Because trains are usually composed of multiple train units, shunting operations must be planned carefully. Significant recent contributions to this problem have been made by Nielsen et al. [2012], Haahr et al. [2016] and Lusby et al. [2017].

The Crew Rescheduling Problem is usually carried out after the timetable and rolling stock have been rescheduled. The new schedules may make the original crew schedule

infeasible, so drivers and conductors must be re-assigned to different tasks in order to complete their duties. For a duty to be feasible, the crew must be in the right place for each task. It must also satisfy certain workload regulations, regarding the frequency and duration of breaks, transfer times in between tasks and specific start and end locations. It is also desirable to change the duties as little as possible. The formulations used for crew rescheduling in railway systems are almost all based on the extended set covering model. Examples include Huisman [2007], Potthoff et al. [2010], Rezanova and Ryan [2010] and Veelenturf et al. [2016]

3.5 Delay Management

The *delay management problem* is closely related to the TTRP. It is concerned with responding to delays specifically by making *wait-depart decisions*. These decisions are made whenever one train is late arriving into a station and another train is ready to depart from the same station. They concern whether the latter train should depart on time, or alternatively wait until the late train has arrived in order to maintain any passenger connections. This problem has been studied by Schöbel [2001, 2007] and Dollevoet et al. [2012], among others.

Whilst both the delay management problem and the TTRP aim to tackle the effects of traffic perturbation by making real-time changes to the timetable, they are quite different paradigms. Delay management is fundamentally a passenger-oriented approach, since data about the complete journeys of passengers is required and the timetable is optimised to best serve these. The operational feasibility of the chosen set of wait-depart decisions is in general not modelled. The TTRP, on the other hand, explicitly models the track capacity to make sure that the rescheduled timetable is feasible in practice. TTRP objectives are usually dependent on train performance,

and do not explicitly consider passengers. Some researchers have produced delay management models that incorporate aspects of the TTRP by considering station capacities [Dollevoet et al., 2015], combining them in an iterative framework [Dollevoet et al., 2014], and even integrating them completely [Corman et al., 2016].

3.6 Summary

A general introduction to the role of optimisation in railway planning was given in this chapter. Our description of the challenges inherent in operational planning is particularly pertinent to the remainder of the thesis, which addresses specific challenges for the TTRP. In addition, our review of planning problems that are closely related to the TTRP provides important context for Chapter 4, where we will discuss formulations for the TTRP.

Chapter 4

TTRP Formulations

Although many optimisation models have been proposed for different variants of the TTRP, most of the formulations used can be distilled into a few main types. Models conforming to each type share similar definitions of decision variables, and have in common ways of modelling track capacity constraints. These basic strategies for formulating the problem, most of which are also relevant for the offline Train Timetabling Problem, are described below. Different perspectives on TTRP formulations are given by Lusby et al. [2011], Harrod [2012] and Toletti [2018].

4.1 A Generic TTRP Formulation

To model the TTRP, the infrastructure utilisation of a set of trains $\mathcal{K} = \{1, \ldots, K\}$ over a time horizon must be modelled. It is assumed that the track is represented by a directed graph, as described in Section 3.2. In this graph, each node represents a track component, and the presence of a directed arc signifies that a train can move from one track component directly to another.

The evolution of the modelled system, comprising of track and trains, can be thought of as a series of *discrete events* that occur at particular points in time. Discrete events usually concern a particular train and track component. For example, a very common type of event is the arrival of a train at a track component. In macroscopic models this might correspond to an arrival at a station, whilst in microscopic models this might mean entering a new block section. Other events can also be defined, such as the departure of a train from a track component.

To model the problem as a MILP, the events that occur to each train $k \in \mathcal{K}$ and the times at which they do so must be represented by a finite number of real variables $\mathbf{x}^k \in \mathbb{R}^{n_k}$. Although these may not be the only decision variables in a TTRP model, they are the most important and ubiquitous, since their optimal values describe the optimal rescheduled timetable.

There are two fundamental types of constraints that are necessary in order to constrain the rescheduled timetable to be achievable in practice:

- **Individual train constraints** pertain to only the variables $\mathbf{x}^k$ of one train $k \in \mathcal{K}$. They ensure that the sequence of events that occurs to train k over the time horizon is feasible in the absence of other trains. Typical individual train constraints ensure that:
  - Trains begin in the correct initial track component at the correct time, reflecting current traffic conditions.
  - Trains traverse feasible sequences of track components.
  - Minimum and maximum traversal times and dwell times are respected, taking into account the kinematic capabilities of trains and any speed limits.
- **Linking constraints** involve decision variables relating to more than one train.

They ensure that the events that occur to different trains do not interact with each other in a way that is not operationally possible. The most ubiquitous linking constraints are *track capacity constraints*, which ensure that there are no train conflicts in the rescheduled timetable. The most common type are *single occupancy* constraints, which ensure that each track component is utilised by at most one train at any one time. Other linking constraints are sometimes present, such as those regarding passenger connections, crews or rolling stock.

We can therefore propose a completely general train routing formulation:

$$\max\{c(\mathbf{x}^1,\ldots,\mathbf{x}^K)|\mathbf{x}^k \in \mathcal{D}^k \quad \forall k \in \mathcal{K} \text{ and } (\mathbf{x}^1,\ldots,\mathbf{x}^K) \in \mathcal{A}\} \tag{4.1}$$

where $n = \sum_{k\in\mathcal{K}} n_k$, each $\mathcal{D}^k \in \mathbb{R}^{n_k}$, $\mathcal{A} \in \mathbb{R}^n$ and $c : \mathbb{R}^n \to \mathbb{R}$. The individual train constraints for train k are defined by $\mathcal{D}^k$, whilst the linking constraints are defined by $\mathcal{A}$. The objective function c represents the quality of the rescheduled timetable defined by $(\mathbf{x}^1,\ldots,\mathbf{x}^K)$. If problem (4.1) can be formulated using linear constraints and objective function, then this formulation is a MILP with the same bordered block diagonal form as problem (2.11) in Section 2.4. This shows that Dantzig-Wolfe reformulation corresponds to the idea of train decomposability, and can be a natural approach to modelling the TTRP.

4.2 Disjunctive Formulations

Disjunctive formulations for the TTRP are characterised by the presence of variables $t_n^k \in \mathbb{R}_+$ that each correspond to the time at which an event n pertaining to train k occurs. For example, if the event that train k arrives at track component s is denoted by n, then t_n^k is the time at which train k arrives at s.

For each train $k \in \mathcal{K}$, individual train constraints usually control the differences between the times of pairs of events using *precedence constraints* of the form

$$t^k_{n_2} - t^k_{n_1} \geq p^k_{n_1,n_2}, \tag{4.2}$$

where $p^k_{n_1,n_2} \in \mathbb{R}$ is the minimum amount of time separating the occurrence of event n_1 and event n_2 for train k. Constraints of this type can accomplish three things, depending on the sign of $p^k_{n_1,n_2}$ and the events used:

1. If $p^k_{n_1,n_2} \geq 0$, then (4.2) ensures that event n_2 occurs a minimum of $p^k_{n_1,n_2}$ later than event n_1. If events n_1 and n_2 represent arrivals at consecutive track components and $p^k_{n_1,n_2}$ is the minimum time required to traverse n_1, then this constraint establishes the order of traversal of these track components, and ensures that the minimum traversal time is respected.

2. If $p^k_{n_1,n_2} < 0$, then (4.2) ensures that event n_1 occurs a maximum of $-p^k_{n_1,n_2}$ later than event n_2. This can be used to model maximum traversal times by letting $-p^k_{n_1,n_2}$ be the maximum time. Constraints of this type are also often used to enforce maximum waiting times in platforms or stations.

3. They can be used to provide absolute upper and lower bounds on the times of events, by using a dummy event n_0 with a fixed time $t^k_{n_0} = 0$ in place of n_1 or n_2. This reduces (4.2) to either $t^k_{n_2} \geq p^k_{n_0,n_2}$ or $t^k_{n_1} \leq -p^k_{n_1,n_0}$. These bounds can be used to express constraints that arise from the timetable on arrivals and departures from stations.

Linking constraints in disjunctive models control the differences between pairs of events for different trains using constraints of the form

$$t^{k_1}_{n_2} - t^{k_2}_{n_1} \geq p^{k_1,k_2}_{n_1,n_2}, \tag{4.3}$$

where $p_{n_1,n_2}^{k_1,k_2}$ is the minimum time that must elapse between event n_1 for train k_1 and event n_2 for train k_2. Analogously to the individual constraints, these can be used to enforce minimum and maximum differences between the times of events relating to different trains. Both (4.2) and (4.3) are known as *precedence constraints*, but they differ in whether the events relate to the same train, or to two different trains.

Single occupancy track capacity constraints are commonly used to limit the occupancy of a particular track component to at most one train at any one time. Suppose it is known that train k_1 will traverse track component s before train k_2, and suppose that *enter* and *leave* are the events corresponding to entering and leaving s. Then the track capacity can be expressed using the constraint $t_{enter}^{k_2} - t_{leave}^{k_1} \geq p_s^{k_1,k_2}$. This ensures that k_1 leaves s before k_2 enters it, therefore respecting the capacity of s. The time $p_s^{k_1,k_2}$ that must elapse between these two events is often known as a *headway*. However, since k_1 and k_2 could be ordered the other way round, the *disjunctive* constraint

$$(t_{enter}^{k_2} - t_{leave}^{k_1} \geq p_s^{k_1,k_2}) \vee (t_{enter}^{k_1} - t_{leave}^{k_2} \geq p_s^{k_2,k_1}) \tag{4.4}$$

must be used. This states that either the left hand side expression must be true, or the right hand side expression i.e. at least one of the trains must traverse s first. In this case, the expressions cannot both be true because each train must enter s before leaving it. In order to prevent s from being traversed by two trains at the same time, a constraint of the form (4.4) must be used for every pair of trains k_1 and k_2 that could potentially traverse s.

4.2.1 Alternative Graph Model

The Alternative Graph (AG) model is a popular disjunctive model for timetable rescheduling. Examples of its use include D'Ariano et al. [2007a, 2008]; D'Ariano

and Pranzo [2009]; Corman et al. [2009a,b, 2011b,a, 2012a]; Kecman et al. [2013]; Samà et al. [2017]. The AG model is typically used as a microscopic model, where track components correspond to block sections. The route of each train (the sequence of block sections to be traversed) must be decided before solving the model. The set of discrete events included in the model for each train then comprises the arrivals into each of these block sections. That is, there is one variable $t_n^k \in \mathbb{R}_+$ corresponding to the time at which train k arrives at each block section on its pre-determined route. Because the sequence of track components is fixed for each train, the basic AG model can perform only retiming and not rerouting, although it has been extended to include rerouting by Corman et al. [2010a].

An *alternative graph* is a directed graph $\mathcal{G} = (N, F, A)$. The node set N is the set of discrete events, consisting of one node for every traversal of a track component by a train. In addition, N contains a dummy node fixed at time zero, and another dummy node that must occur after all other events. Precedence constraints are represented by the set of directed arcs F, called *fixed arcs*. For example, $t_{n_2}^k - t_{n_1}^k \geq p_{n_1,n_2}^k$ would be represented by a directed arc from event n_1 for train k to event n_2 for train k of length p_{n_1,n_2}^k. Each disjunctive constraint that is used for the capacity of a track component is represented by a pair of arcs in A, called *alternative arcs*. For example, the disjunction (4.4) would be represented by two arcs, one from event *leave* for train k_1 to event *enter* for train k_2 of length $p_s^{k_1,k_2}$, and the other from event *leave* for train k_2 to event *enter* for train k_1 of length $p_s^{k_2,k_1}$.

A solution to the AG model consists of two parts. First, one alternative arc out of each pair must be chosen to be kept and the other discarded. This corresponds to choosing the order of each pair of potentially conflicting trains. Second, the time t_n^k at each node must be chosen to satisfy the remaining precedence constraints. This is done to minimise the time difference between the two dummy events, that occur before and after all other events, respectively. In the parlance of machine scheduling

literature, this is minimising the *makespan*. In fact, the AG model has strong similarities to the well-known *job-shop scheduling problem*. Trains correspond to 'jobs' and track components to 'machines'. Single occupancy track capacity constraints correspond to the constraint that each machine can process at most one job at once. Trains must move directly from one track component to the next: these are known as *blocking constraints* in the machine scheduling literature.

4.2.2 Disjunctive MILP Formulations

When disjunctive formulations are solved using MILP techniques, disjunctive constraints of the form (4.4) must be linearised. This usually achieved by introducing a binary variable $x_s^{k1,k2}$ that indicates whether k_1 traverses s before k_2 or vice versa. Constraint (4.4) can then be replaced by:

$$t_{enter}^{k_2} - t_{leave}^{k_1} + M x_s^{k1,k2} \geq p_s^{k_1,k_2} \tag{4.5}$$

$$t_{enter}^{k_1} - t_{leave}^{k_2} + M(1 - x_s^{k1,k2}) \geq p_s^{k_2,k_1}. \tag{4.6}$$

When $x_s^{k1,k2} = 0$, constraint (4.5) corresponds to the left hand side expression of (4.4) and when $x_s^{k1,k2} = 1$, constraint (4.6) corresponds to the right hand side expression of (4.4). Since $x_s^{k1,k2}$ must take value either 0 or 1, the disjunction of (4.4) is enforced. The value of the constant M is chosen sufficiently large that (4.6) is redundant when $x_s^{k1,k2} = 0$ and (4.5) is redundant when $x_s^{k1,k2} = 1$. Collectively, (4.5) and (4.6) are called *big-M* constraints. Almost all disjunctive models use big-M constraints for disjunctions. A notable exception is the model of Lamorgese and Mannino [2019]. They apply a combination of Benders' reformulation and polyhedral results to replace the big-M constraints by dynamically added knapsack cover inequalities and other linear inequalities.

Several disjunctive MILP models perform rerouting in addition to re-timing. One approach, taken by Törnquist and Persson [2007] and Acuna-Agost et al. [2011a], is to use additional binary variables x_s^k that indicate whether train k uses track component s. Additional network flow constraints are necessary to ensure that the track components used by each train form a feasible sequence through the railway network. Constraints (4.5) and (4.6) can no longer be used for track capacity constraints, since it is not known that k_1 and k_2 will traverse s. Instead, the following constraints can be used:

$$t_{enter}^{k_2} - t_{leave}^{k_1} + Mx_s^{k1,k2} + M(1 - x_s^{k_1}) + M(1 - x_s^{k_2}) \geq p_s^{k_1,k_2} \tag{4.7}$$

$$t_{enter}^{k_1} - t_{leave}^{k_2} + M(1 - x_s^{k1,k2}) + M(1 - x_s^{k_1}) + M(1 - x_s^{k_2}) \geq p_s^{k_2,k_1}. \tag{4.8}$$

These constraints use additional big-M terms to ensure that the disjunction is only enforced if both trains traverse s. If either $x_s^{k_1} = 0$ or $x_s^{k_2} = 0$, then both of (4.7) and (4.8) are both redundant. Meanwhile, if $x_s^{k_1} = 1 = x_s^{k_2}$, then (4.7) and (4.8) become identical to (4.5) and (4.6).

A related approach, taken by Pellegrini et al. [2014], D'Ariano et al. [2014] and Min et al. [2011], is to instead enumerate valid subsequences of track components that make up routes r though a particular area, usually a station. Binary variables x_r^k are then used to indicate whether route r is used by train k, and a constraint is required to ensure that exactly one of these routes is selected. In this case, (4.5) and (4.6) are replaced by constraints

$$t_{enter}^{k_2} - t_{leave}^{k_1} + Mx_s^{k1,k2} + M(1 - x_{r_1}^{k_1}) + M(1 - x_{r_2}^{k_2}) \geq p_s^{k_1,k_2} \tag{4.9}$$

$$t_{enter}^{k_1} - t_{leave}^{k_2} + M(1 - x_s^{k1,k2}) + M(1 - x_{r_1}^{k_1}) + M(1 - x_{r_2}^{k_2}) \geq p_s^{k_2,k_1} \tag{4.10}$$

for each pair of routes r_1 and r_2 for the two trains that both contain track component

s. Note that (4.9) and (4.10) reduce to (4.5) and (4.6) if $x_{r_1}^{k_1} = 1 = x_{r_2}^{k_2}$. As a result, the disjunction for track component s is enforced if and only if both trains use routes that contain s.

4.2.3 Discussion

We have shown that disjunctive formulations are a convenient way of modelling the TTRP. Individual train constraints including track component order, minimum and maximum traversal times and absolute bounds on the times of events can be modelled using precedence constraints. We also showed that disjunctions can be used to model single occupancy track capacity constraints. The AG model and disjunctive MILPs were introduced, including different ways of modelling rerouting with big-M constraints.

Whilst disjunctive models excel at modelling the TTRP, solving them quickly is often more problematic. The difficulty stems from the disjunctive constraints. When disjunctions are modelled using big-M constraints in a MILP, the bounds that can be obtained by solving LP relaxations are generally not very close to the optimal objective value. Formulations with this property are known as *weak* formulations, and can be time consuming to solve to optimality. This motivates our interest in the time-indexed formulations, which are discussed next.

4.3 Time-indexed Formulations

Time-indexed formulations are characterised by partition of the time horizon into a set of discrete time intervals $\mathcal{T} = \{1, \ldots, T\}$. The discrete events considered are *timed events*, meaning that each one is associated with a time interval, as well as a

train and a track component. Popular examples of a timed event n for train k include its arrival at a track component s in time interval t, or its utilisation of s during time interval t. Since time is already part of the definitions of events, binary variables are often used to indicate which events occur. The enumeration of the timed events n that index the variables is only possible because the time horizon is represented as a finite discrete set.

4.3.1 Arc Packing

Arc packing models are typically based around representing timed events as nodes in a time-space graph (time-space graphs are explained in Section 2.6). The schedule of each train is represented as a *source-sink* path in this graph, corresponding to a sequence of timed events. Directed arcs $a = (n_1, n_2)$ are used in the time-space graph to indicate that timed event n_2 may directly follow timed event n_1. Suppose that event n_1 indicates arrival at track component s_1 at time interval t_1 and n_2 similarly with s_2 and t_2. Then the inclusion of arc $a = (n_1, n_2)$ in the time-space graph indicates that a train may consecutively traverse s_1 followed by s_2, and that $t_2 - t_1$ time intervals should elapse between arrivals. In other words, the track topology and traversal times may be represented using the time space graph.

Models such as [Caprara et al., 2002] and [Bettinelli et al., 2017] use binary variables x_a^k to indicate whether or not arc a is in the *source-sink* path of train k. The first formulation presented in Chapter 5 is also of this kind. The individual train constraints, which are implicit in the requirement that each train is assigned a *source-sink* path in the time-space graph, are enforced using network flow constraints.

Using the variables defined, single occupancy track capacity constraints can be ex-

pressed using constraints of the form

$$\sum_{k \in \mathcal{K}} \sum_{a \in A} x_a^k \leq 1, \tag{4.11}$$

where A is some set of arcs implying occupancy of a particular track component. This is where the name *arc packing* comes from: this is a set packing constraint involving variables indexed by arcs. In the simplest models, the sets A consist of just one arc. However, A can be defined in different ways to represent more complex track capacity constraints. For example, Harrod [2011] includes arcs that represent transitions between track components, rather than just occupation of track components. Our model in Chapter 5 extends constraints (4.11) in a different way to take into account the complexities of signalling in large station areas.

Some arc packing models, such as [Brännlund et al., 1998], do not explicitly use time-space graphs. Rather, they contain variables $x_{s,t}^k$ to indicate whether train k uses track component s at time interval t. Individual train constraints can be expressed directly using these variables, and single occupancy track capacity constraints can be expressed as

$$\sum_{k \in \mathcal{K}} x_{s,t}^k \leq 1 \quad \forall s, t. \tag{4.12}$$

Although variables $x_{s,t}^k$ are not indexed by the arcs of a time-space graph, this formulation is very similar to the standard arc packing formulation.

4.3.2 Path Packing

Path packing formulations use groups of timed events such as entire timed train runs i.e. timed sequences of track components. In time-space graph models, these correspond to *source-sink* paths in the graph, and the variables x_p^k indicate whether train k uses path p. In some models for station areas, such as Caimi et al. [2012],

the groups of timed events p represent train runs from the entrance of the station to a platform. Individual train constraints are often implicit in these paths, in the sense that the path p represented by x_p^k usually already represents a feasible sequence of track components and respects minimum traversal times. Each train is additionally constrained to be allocated exactly one path, using a constraint of the form $\sum_{p \in P^k} x_p^k = 1$, where P^k is a set of possible paths for train k. The set of paths P^k may be enumerated prior to solving the optimisation model, or dynamically generated — this is discussed in Section 5.3.

Track capacity constraints in path packing models can be represented using set packing constraints. Suppose that path p_1 for train k_1 and path p_2 for train k_2 conflict (i.e. both require the use of a track component during the same time interval). Zwaneveld et al. [2001] prevent them from both being selected by adding the constraint

$$x_{p_1}^{k_1} + x_{p_2}^{k_2} \leq 1. \tag{4.13}$$

However, constraints of this type generally lead to weak MILP formulations. This problem can be mitigated by using a *conflict graph*. This is an undirected graph in which the nodes are paths, and an arc is present between two nodes if and only if the paths represented by the nodes have a conflict. Instead of including a constraint for each conflicting pair (corresponding to an arc in the conflict graph), constraints of the form

$$\sum_{(k,p) \in C} x_p^k \leq 1 \quad \forall \textit{ maximal cliques } C \tag{4.14}$$

can be included. A maximal clique C in the conflict graph is a maximal subset of the nodes such that every pair of nodes in the subset is adjacent. Constraints (4.14), called *clique inequalities*, lead to stronger MILP formulations than if constraints of the form (4.13) are used (see Padberg [1973]). Methods for identifying these cliques that are specific to the TTRP have been developed by Caimi et al. [2011]. Cliques are

added dynamically during the solution process to avoid the computational burden of calculating them all in advance.

Track capacity constraints in path packing models can also be formulated by using constraints that are each linked to a specific single occupancy track component and time interval. For example, suppose that $P^k_{s,t}$ consists of all paths for train k that imply utilisation of track component s in time interval t. Then the capacity of s in time interval t can be limited by the constraint

$$\sum_{k \in \mathcal{K}} \sum_{p \in P^k_{s,t}} x^k_p \leq 1. \tag{4.15}$$

We build on this approach in the second TTRP formulation presented in Chapter 5 to make it suitable for complex station areas.

4.3.3 Resource Trees

The models of Caimi et al. [2011] and Lusby et al. [2013] both use *resource trees* in place of traditional time-space graphs. A resource tree for a particular train is a dynamically generated tree in which each node is a timed event. The branches of the tree represent the outcomes of combinations of discrete decisions for each train, such as which track component to occupy next when there is a choice, and what speed to travel at. Individual train constraints are represented in the construction of the resource trees. Paths from source nodes to sink nodes in resource trees can be represented using either arc variables x^k_a or path variables x^k_p. Track capacity constraints can therefore be modelled in the same way as arc packing and path packing formulations.

4.3.4 Path Configuration

Borndörfer and Schlechte [2007] have proposed the Path Configuration Problem formulation. This is an extended formulation of the path packing problem that affects the way that track capacity constraints are represented in the model. New binary variables y_q are introduced to indicate the selection of *configurations* q. A configuration is a set of timed arcs, corresponding to a particular track component, that do not conflict with one another. Configurations are represented as paths in a directed graph, and at most one configuration can be chosen per track component. The track capacity constraints are enforced via constraints of the form

$$\sum_{p:a\in p} x_p \leq \sum_{q:a\in q} y_p$$

that state that a path p containing particular arc a can be chosen for a train only if a configuration q containing that arc is also chosen.

4.3.5 Cumulative Flow

Meng and Zhou [2014] have proposed an extension of the arc packing problem that uses *cumulative flow variables*. These are variables $a^k_{i,t}$ and $d^k_{i,t}$ that indicate whether train k has arrived at (respectively departed from) track component i by time t. These are coupled with variables $x^k_{i,t}$ that express whether train k is using i at time t. This is achieved using constraints

$$x^k_{i,t} = a^k_{i,t+g} - d^k_{i,t-h}, \tag{4.16}$$

where g is the number of time intervals before arrival that a train starts occupying the track component, and h is the amount of time after departure that it continues

to do so. This variable coupling allows the track capacity constraints to be expressed using constraints similar to (4.12).

4.4 Discussion

Our review of TTRP formulations has focussed on underlying mathematical structures, and in particular variable definitions and the representation of single occupancy track capacity constraints. It is clear that a wide variety of different types of formulations have been proposed.

In some cases, the choice of formulation enables specific types of modelling to be carried out more easily. For example, connections are much easier to model using disjunctive formulations than they are using time-indexed formulations. On the other hand, modelling rerouting is very natural in time-indexed formulations, but requires many additional variables and constraints in disjunctive models. Because this thesis tackles the TTRP in complex station areas, rerouting is very important and so time-indexed formulations are a natural choice.

In other cases, the choice of formulation is influenced by computational factors. Disjunctive models usually use big-M constraints, which lead to weak LP relaxations and cause poor branch-and-bound performance. Whilst time-indexed models avoid big-M constraints, they can also be hard to solve because they require a large number of binary variables. This is a particular concern for microscopic models, since they often have many track components and use short time intervals, for example of length 15 seconds. In this thesis, we develop solution methods for a time-indexed model to try to overcome this difficulty. We show that time-indexed models can be suitable for solving a microscopic TTRP model.

There are many other ways in which TTRP models can be classified:

- Whilst our model uses a microscopic track topology, there are many macroscopic existing models. Further distinctions within these groups are discussed in Section 5.3.
- Whilst our model is designed for complex station areas, TTRP models can be designed for single track lines, double track lines or large networks. There are a growing number of works dedicated to high-speed lines.
- Not all models use single occupancy constraints to model track capacity. More complex track capacity constraints are introduced in Section 5.4.6.
- TTRP models can be classified by the severity of traffic perturbations they are designed to respond to. Whilst we focus on relatively small disturbances, other authors, such as Zhan et al. [2015], focus on complete blockages of the track.
- TTRP models perform different combinations of rescheduling strategies. Whilst our model performs retiming, rerouting, skipping stops and (optionally) cancellations, many macroscopic models focus on other strategies such as *short turning* [Altazin et al., 2020].
- Train speed, including acceleration and deceleration can be modelled in different ways. This is discussed in much more detail in Section 6.2.
- Additional features of some TTRP models that we have not mentioned include consideration of rolling stock considerations, train and platform length and passenger connections.

More comprehensive coverage of all of these issues is available in review papers by Cacchiani et al. [2014], Corman and Meng [2015], and Fang et al. [2015].

Chapter 5

A multicommodity flow model for rerouting and retiming trains in real-time to reduce reactionary delay in complex station areas

5.1 Introduction

Delays to passenger trains are a significant problem in Great Britain. In 2018–2019, only 65.3% of recorded passenger train stops occurred on time [Network Rail, 2020], and the time that passengers lose each year as a result of arriving late at their destinations has been estimated to be worth at least £1 billion [National Audit Office, 2008, p. 46]. In 2019, 64.35%[1] of total train delay minutes were due to *reactionary delays*, which are delays that are caused by the knock-on effect of prior delays. The prevalence of reactionary delays is therefore a very significant part of the overall problem.

[1]Statistic provided by Network Rail.

In this paper, we present a model and algorithm that can be applied to reduce the amount of reactionary delay after an initial delay incident. This is achieved by solving the Train Timetable Rescheduling Problem (TTRP) [Cacchiani et al., 2014]. The TTRP consists of calculating new schedules (called *rescheduling*) for trains in real-time that are different from those originally planned. These new schedules should maximise a measure of utility so as to represent the best possible response to the delay incident. Rescheduling consists of both *rerouting* and *retiming*, finding new routes and timings for a train, respectively. Rerouting implicitly includes *replatforming*, finding new platforms at the same station for a train to stop at. The TTRP is solved within a defined area of a railway network over a fixed time horizon. The new schedules should be practically achievable under the constraints of the signalling system.

Although the TTRP has already benefited from a significant amount of research attention, which has been well documented by Corman and Meng [2015], Cacchiani et al. [2014] and Fang et al. [2015], the successful deployment of real-time optimization algorithms as a decision support tool for timetable rescheduling has so far been limited (see Borndörfer et al. [2017] and Lamorgese et al. [2018]). One reason for this is that researchers have been unable to solve realistic instances to optimality in the strict solution time limits required in a real-time environment. Instead, some have relied on heuristic methods that offer no guarantees on the quality of solutions found. Others have proposed simplified models that either carry out rerouting and retiming separately, approximate the signalling constraints or use simplistic objective functions.

In this study, we propose a multicommodity flow model for the TTRP that performs rerouting and retiming simultaneously. The model employs a novel representation of the signalling constraints that is accurate and detailed, yet leads to a tractable formulation of the problem. Our objective function uses the concept of utility to flexibly

represent complex rescheduling preferences. We present a new set of instances based on real data from an area of the British railway network centred around Doncaster station. Finally, we show that these instances can be solved to either optimality or provably near to optimality in sufficiently short times using a tailored branch-and-price algorithm.

This research has been carried out in collaboration with Network Rail, the railway infrastructure manager in Great Britain. In 2018–2019 on their network, 1.8 billion passenger journeys were made, totalling 68 billion passenger kilometres. Network Rail is seeking to use new technology to improve their ability to manage disruption on the railway, and therefore to improve the reliability of passenger services. This represents a good opportunity for optimization-based rescheduling to have an impact.

The paper is organised as follows. In Section 5.2, we define some prerequisite railway signalling concepts and terminology before reviewing the current literature in Section 5.3. Section 5.4 introduces the model. In Section 5.5, we show how to formulate the problem as an integer program, which can be solved using the branch-and-price solution algorithm presented in Section 5.6. Section 5.7 describes the creation of the test instances, Section 5.8 presents our computational results, and finally Section 5.9 offers our conclusions and suggested directions for future research.

5.2 Railway Signalling

In order to ensure safe operation of the railway, the movement of trains is regulated by a signalling system. Signalling is a complex topic that is covered in detail by Pachl [2014]. For conciseness, in this paper we will focus only on the details relevant to our model for the TTRP. Although we describe the workings of the signalling

system in Great Britain, it is similar elsewhere. The signalling system controls train movements at stations, junctions and on the lengths of uninterrupted track that run between them, called *open lines*. Signalling for stations and junctions is different to the signalling on open lines.

The vast majority of open lines in Great Britain are signalled with single-direction *Automatic Block Signalling*. In this system, the track is divided into *block sections*. Each block section is protected by a signal that is visible to drivers in advance of arrival at the block section. Signals display a green aspect to indicate that an approaching train may pass the signal and enter the block section. In contrast, a red aspect indicates that an approaching train must stop before entering the block section. Signals protect block sections by ensuring that there is at most one train in each block section at any one time and therefore that trains cannot collide. Though we do not include them in our model, yellow and double-yellow aspects are often used to indicate caution, warning trains to slow down when the next or next-but-one signal, respectively, is red.

In contrast, stations and junctions are usually signalled with an interlocking system. This is a more complicated signalling arrangement designed to cope with the increased complexity of converging, diverging and crossing tracks within these areas. The track within an interlocked area is divided into *track circuits*. In a station, some of these track circuits contain platforms. Signals are placed both at the limits of an interlocked area and at the end of platforms. The red and green aspects of these signals have the same meanings as the red and green aspects of signals on the open line. Trains travelling through interlocked areas begin and end this traversal at signals at the limit of the interlocked area. During this traversal, they may also visit signals positioned at platforms.

In an interlocked area, the route from one signal to an adjacent signal consists of a

sequence of track circuits. The signal at the start of a route will allow a train to traverse the route only if it has first been *locked*. The interlocking system prevents a route from being locked unless all track circuits that make up the route are free, i.e. are not currently locked for another train. Therefore, when a route is locked for a train, no other route with track circuits in common can be locked. Since they cannot be locked, they cannot be traversed. In *route-release* interlocking systems, the track circuits in a route are *released* simultaneously when the entire route is cleared. In *sectional-release* interlocking systems, each track circuit is individually released when it is cleared. We will show how to model both of these release systems.

In our model, the open line is treated as an interlocked area. This is because the Automatic Block Signalling system used on the open line can be seen as a special case of interlocking, where each block section consists of a single track circuit. This observation is valid regardless of the interlocking release type.

In Great Britain, and in this paper, the word *berth* is used to refer to signals. We use the word *route* to mean the track that must be traversed to get from one berth to another. We use the terms berth and route regardless of whether the signals are on the open line or in an interlocking area. We define the *traversal time* to be the minimum amount of time required to traverse a route. We define the *headway* to be any additional time required between a train vacating a route or track circuit, and it being released and ready for use by another train. The sum of these two times is called the *blocking time*.

5.3 Literature Review

Since the literature on railway optimisation is very broad, we will focus specifically on the TTRP and its offline equivalent, the Train Routing Problem (TRP) where

relevant.

Models for the TTRP divide railway track into discrete *track components*. This facilitates the representation of both the locations of trains over time, and the signalling constraints. Track components of different types can be used, such as track circuits, block sections, platforms or even whole stations or lines. The type of the track components used affects the granularity of the model. *Macroscopic* models usually represent stations using nodes and the tracks in between stations using edges. This approach allows for large areas of a railway network to be modelled, but fails to capture signalling constraints accurately. As a result, macroscopic models are unsuitable for rescheduling in complex station areas. *Microscopic* models usually model individual track circuits or block sections. While models that use block sections are very common in the rescheduling literature, they cannot model sectional-release interlocking systems accurately. As a result, such models often underestimate the capacity of large station areas [Corman et al., 2009b; Pellegrini et al., 2014]. The capacity of station areas with sectional release interlocking systems can only be accurately represented by models that use track circuits, such as those proposed by Corman et al. [2009b], Pellegrini et al. [2014], Caimi et al. [2011] and Lusby et al. [2013]. The main disadvantage of these models is that they are often intractable for large stations due to the large number of decision variables required to model the considered track circuits.

Models for the TTRP can also be classified according to the way in which time is modelled. *Disjunctive* models use continuous variables to represent the times at which trains begin using individual track components. *Time-indexed* models, on the other hand, divide time into discrete time intervals and use binary variables to indicate whether or not each train uses a given track component during a given time interval. Disjunctive and time-indexed models represent the constraints imposed by the signalling system differently. In both cases, signalling constraints are represented

by capacity constraints on the track components. In disjunctive models, for a given track component, one capacity constraint is defined for each pair of trains that use the component. These constraints are in the form of a disjunction between the two possible orders in which the pair of trains can use the track component. In contrast, a capacity constraint on a track component in a time-indexed model can be represented by an upper bound on the number of trains that use each track component in each time interval. A time-indexed model has been developed for the TTRP in this paper. The relative merits of disjunctive and time-indexed models are discussed below.

One of the most popular models for the TTRP is the Alternative Graph model of D'Ariano et al. [2007a]. This is a disjunctive model that was originally developed by Mascis and Pacciarelli [2002] for machine scheduling. It is suitable for railway retiming because when the routes of the trains are fixed, the TTRP becomes a variant of the job-shop scheduling problem. D'Ariano et al. [2007a] present an exact branch-and-bound procedure for minimising the maximum reactionary delay, which is incorporated in a full dispatching system described by D'Ariano et al. [2008]. This branch-and bound-algorithm is further developed by Mannino and Mascis [2009], who propose stronger lower bounds to speed up the enumeration process.

The Alternative Graph model has two important shortcomings. The first is that it cannot model rerouting. The ability to perform rerouting is crucial to finding good rescheduling solutions in stations with many routing alternatives. Though the model has been extended by Corman et al. [2010a] to incorporate rerouting, the authors were only able to find heuristic solutions to the resulting problems. The second shortcoming of the Alternative Graph model is that the exact branch-and-bound algorithm of D'Ariano et al. [2007a] can only be used when the objective function is to minimise the maximum of some measure over all station stops. The advantages of instead optimising the total of some measure over all station stops is discussed in Section 5.4.8. Although Samà et al. [2015] have shown that the model

can be extended to accommodate many different objective functions, the resulting problems can only be solved with generic Mixed Integer Programming (MIP) solvers, the drawbacks of which are discussed next.

In order to overcome the shortcomings of the Alternative Graph model, Törnquist and Persson [2007], Pellegrini et al. [2015], Meng and Zhou [2014], Min et al. [2011] and Acuna-Agost et al. [2011a] have proposed disjunctive MIP formulations for the problem. All of these models allow trains to be rerouted, and they all include either total delay or total weighted delay as part of the objective function. However, the representation of disjunctive constraints in a MIP requires the use of big-M constraints. As a result, these models have weak Linear Programming (LP) relaxations. This makes them difficult to solve to optimality using LP-based branch-and-bound in the short run times required for real-time rescheduling. Although both Samà et al. [2016] and Törnquist Krasemann [2012] have proposed heuristics for these models with acceptable solution times, it is not possible to know the quality of any solutions produced by these algorithms.

Time-indexed models are an alternative to disjunctive models which avoid the computational difficulties associated with disjunctions. In time-indexed models, the time horizon is divided into discrete time intervals. This allows *time-space resources* to be defined, each of which is a pair consisting of one of track component and one time interval. Typically, a binary variable is used to indicate whether a train consumes a given time-space resource. Rescheduling can be seen as problem of assigning time-space resources to trains. The track capacity constraints can be represented by introducing one set packing constraint for each time-space resource. Constraints are also required to ensure that the assignment of time-space resources for each train represents a feasible itinerary. This can be achieved using a time-space graph, where each node represents a time-space resource and each directed arc is a feasible transition between them. An allocation of time-space resources to a train can be

represented as a path in the time-space graph.

Time-indexed models have been predominantly used in offline railway optimisation problems such as the Train Timetabling Problem (TTP) [Brännlund et al., 1998; Caprara et al., 2002; Borndörfer and Schlechte, 2007; Cacchiani et al., 2008], train routing [Zwaneveld et al., 2001; Lusby et al., 2011; Harrod, 2011; Caimi et al., 2011] and train platforming [Caprara et al., 2011]. In contrast, time-indexed models have not been widely used for the TTRP, due to a perception that they contain too many variables for the solution of real instances in the stringent time limits imposed by real-time environments. Recently, however, some authors have successfully used time-indexed models for real-time applications. Lusby et al. [2013] use a branch-and-price algorithm for the TTRP, whilst Meng and Zhou [2014] use a time-indexed formulation for rescheduling on a railway network with multiple parallel tracks, and Bettinelli et al. [2017] present an effective iterative heuristic for real-time rescheduling.

Since time-indexed models can produce very large formulations, most authors decompose the problem so that each binary variable represents a full train path. Though this has been achieved in some settings by the enumeration of train paths [Zwaneveld et al., 2001; Caprara et al., 2011], it is most common to dynamically generate each path variable. This has been achieved within both Lagrangian relaxation [Brännlund et al., 1998] and column generation [Borndörfer and Schlechte, 2007; Cacchiani et al., 2008; Lusby et al., 2013] frameworks. Both Caimi et al. [2011] and Lusby et al. [2013] generate new path variables using tree structures that capture the sequence of decisions taken for each train over time. Though this approach allows detailed train speed profiles to be considered, it is not scalable, since the size of the trees grows exponentially with the number of decisions in this sequence. Our model avoids this problem, because new path variables are generated by solving shortest path problems on a time-space graph. Since the time-space graph is directed and acyclic, the complexity of these problems is linear in the number of edges in the time-space

graph.

In this paper, we make the following contributions:

1. We present a new model for the Train Timetable Rescheduling Problem that is capable of performing rerouting and retiming simultaneously.

2. We demonstrate that it is possible to model a dense, sectional-release interlocking area accurately by using routes as the track components. This is made possible by introducing a distinction between two types of capacity consumption: *occupying* and *banning*. This distinction allows track circuits to be modelled implicitly. Our model is an improvement on models that explicitly use track circuits as the track components. Such models can be intractable due to the large number of variables.

3. We present a new objective function, which was developed in collaboration with Network Rail. This objective function uses the concept of utility to represent Network Rail's preferences more accurately than existing objective functions in the literature.

4. We present a new set of instances based on real data for an area around Doncaster station in the UK. We provide important details of how we generated the instances, the provision of which is scant in much of the other published literature.

5. We show that these instances can be solved either to optimality or provably near to optimality in times suitably short for real-time operations. We present a tailored branch-and-price algorithm for this purpose in which the subproblem for each train is a shortest path problem, and therefore efficiently solvable. We demonstrate the success of several acceleration strategies from the column generation literature.

6. We show that it is now realistic to solve time-indexed models for the TTRP to optimality in real-time environments.

7. We demonstrate, both theoretically and empirically, the relationship between the number of conflicts in an instance and its difficulty.

5.4 Modelling the Train Timetable Rescheduling Problem

5.4.1 Description of the Problem

Following a disturbance to the timetable, the problem is to find a new route and set of timings (i.e. a new schedule) for each controlled train such that there are no conflicts between trains and Network Rail's utility is maximised. Controlled trains are those that are forecast to be inside a defined area of track during a time horizon. The new route can involve stopping at different platforms from those originally planned, taking different approaches to planned platform stops, or cancelling the stop altogether. There are no conflicts between trains if and only if the new schedule can be implemented in practice under the signalling system. Network Rail's utility is modelled as the total weighted utility over all train stops which are carried out, where at each stop the utility is a function of lateness.

5.4.2 Modelling Approach

Our approach to modelling the problem is to build a directed, acyclic time-space graph $G = (N, A)$. Each arc $a \in A$ has an arc weight c_a^k for each train $k \in \mathcal{K}$. In addition, we specify two different sets of arcs, $A_n \subseteq A$ and $\bar{A}_n \subseteq A$, for each

node $n \in N$. The graph G is configured such that every finite-weight *source-sink* path describes a feasible timed train route through the modelled area. If a train k is assigned such a path, its total weight is the negative of the utility of the train carrying it out. The arc sets A_n and $\bar{A}_n$ are used in our novel representation of the track capacity constraints, which is described in Section 5.4.6.

The TTRP is formulated as follows:

TTRP. *Find a source-sink path in G for each train $k \in \mathcal{K}$ such that the total weight of all the paths is minimised, subject to the constraints that for each node $n \in N$, and across all train paths (i) at most one arc in A_n is used and (ii) if an arc in A_n is used then no arcs in $\bar{A}_n$ are used.*

The rest of this section explains how the graph G, the arc weights c_a^k and the arc capacity sets A_n and $\bar{A}_n$ are built. In order to describe how G is built in Section 5.4.5, we first explain how time is modelled in Section 5.4.3 and how the track is modelled in Section 5.4.4. The track capacity constraints are explained in Section 5.4.6. Sections 5.4.7 and 5.4.8 explain the objective function, and details about the arc weights c_a^k are given in Section 5.4.9.

5.4.3 The Time Horizon

Let the time horizon over which trains should be rescheduled be $[0, T]$. The time horizon should be long enough to allow a wide variety of rescheduling options to be considered, but not so long that new primary delays during the time horizon are likely to occur, making further rescheduling necessary. We use a time horizon of length 60 minutes in our experiments.

The time horizon is divided into a discrete set of consecutive time intervals $\mathcal{T} = \{0, \ldots, T\}$ of equal length. The number of time intervals should be large enough

to offer a reasonable approximation of continuous time, but not so large that the problem becomes intractable. We use time intervals of length 15 seconds in our experiments, resulting in 240 time intervals overall.

5.4.4 The Route Graph

Let N_b be the set of berths within the area of track being modelled, and let $A_b \subset N_b \times N_b$ be the set of permissible transitions between these berths (i.e. routes). Together, these form the directed berth graph $G_b = (N_b, A_b)$. The graph G_b is typically connected, although this is not necessary. A small example for illustrative purposes is provided in Figure 5.1(a).

From the berth graph, we can derive the route graph $G_r = (N_r, A_r)$, where $N_r = A_b$ and A_r consists of directed arcs between routes that together form a path of length 2 in G_b. This is sometimes known as the line graph of G_b, and is shown in Figure 5.1(b). A route consists of the track between two berths, or signals. A traversal of $r = (b_1, b_2)$ is defined as starting when a train passes signal b_1 and finishes when it passes signal b_2.

Each route $r \in N_r$ has data associated with it. Let L_r be the minimum number of time intervals required to traverse the route. Let h_r be the number of time intervals that should be left as headway between any two traversals, so that $L_r + h_r$ is the blocking time for route r. Let $TC_r = \{tc_r^i\}_{i=0}^{n_r-1}$ be the set of track circuits for route r, in order of traversal from tc_r^0 to $tc_r^{n_r-1}$ Our methodology for collecting and in some cases inferring this data is described in Section 5.7.

Finally, we expand G_r to take account of platforms. Let PL be the set of platforms. Each platform $pl \in PL$ occurs immediately before a berth $b^{pl} \in N_b$. This means that each platform pl lies at the end of any route that it occurs within. We truncate each

of these routes, so that they finish immediately before the platform pl. We insert two new artificial routes pl_{stop} and pl_{pass}, which both represent the track alongside the platform. Traversal of pl_{stop} corresponds to stopping at platform pl to allow passengers to alight and embark. Traversal of pl_{pass} corresponds to passing the platform without stopping. This separation allows pl_{stop} and pl_{pass} to have different traversal times. Separating platforms from the routes they lie on also allows us to measure the exact times at which trains arrive at platforms, which is important for the objective function.

In order to describe the procedure for separating these new platform routes, we first define

$$\sigma^-(b^{pl}) = \{(i,j) \in A_r : j = b^{pl}\} \text{ and } \sigma^+(b^{pl}) = \{(i,j) \in A_r : i = b^{pl}\}$$

to be the sets of routes ending and starting at berth b^{pl}, respectively. The steps for separating the new platform routes are as follows:

1. Remove all arcs in A_r that begin at a route in $\sigma^-(b^{pl})$ and end at a route in $\sigma^+(b^{pl})$.

2. Insert the new nodes pl_{stop} and pl_{pass} into N_r.

3. Insert into A_r all arcs from the sets $\{(r,r') : r \in \sigma^-(b^{pl}), r' = pl_{stop}, pl_{pass}\}$ and $\{(r,r') : r = pl_{stop}, pl_{pass}, r' \in \sigma^+(b^{pl})\}$.

4. Set the data for pl_{stop} and pl_{pass}. The traversal and headway times $L_{pl_{pass}}$ and $h_{pl_{pass}}$ for pl_{pass} are set to zero. The traversal and headway times $L_{pl_{stop}}$ and $h_{pl_{stop}}$ for pl_{stop} are set to the minimum dwell time D_{pl} and minimum headway H_{pl} for the platform pl, respectively. D_{pl} is the minimum number of time intervals required to allow passengers to alight and embark at platform pl, and H_{pl} is the minimum number of time intervals required once a train has left the

platform pl before another train can arrive. Both pl_{stop} and pl_{pass} consist of the same single track circuit.

Figure 5.1(c) shows the graph that results from following this procedure in our example.

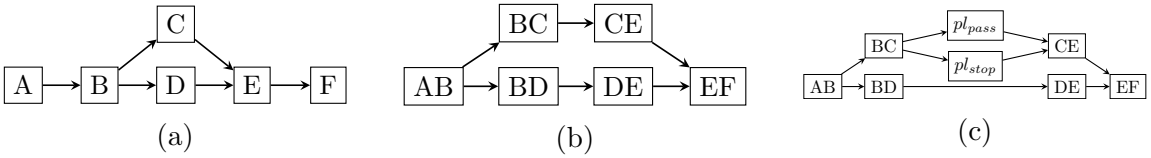


Figure 5.1: A small example to illustrate the construction of the route graph. (a) shows the berth graph G_b; (b) shows the corresponding route graph G_r before expanding the platforms. (c) shows the route graph expanded for a platform pl that exists at the end of route BC.

5.4.5 The Time-space Graph

Let the node set $N = (N_r \times \mathcal{T}) \cup \{source, sink\}$ of G consist of all of the route, time interval pairs (r, t), in addition to an artificial source and sink. For convenience, let $N_0 = N \setminus \{source, sink\}$.

The directed arc set $A = \bigcup_{i=1}^{6} A_i$ of G consists of the following groups of arcs:

(i) $A_1 = \{(source, (r_0^k, a_0^k)) : k \in \mathcal{K}\}$ corresponding to entering from the source node to the known location of the train either at the beginning of the time horizon or when the train first enters the area during the time horizon.

(ii) $A_2 = \{(source, sink)\}$ consists of a single arc, corresponding to a train cancellation. This can be omitted if desired.

(iii) $A_3 = \{((r, t), (r, t+1)) : (r, t) \in N \text{ and } (r, t+1) \in N\}$ corresponding to waiting in route r for one time interval.

(iv) $A_4 = \{((r, t), (r', t + L_r)) : (r, r') \in A_r, (r, t) \in N \text{ and } (r', t + L_r) \in N\}$ corresponding to traversing r and arriving in r' after the minimum traversal time

L_r.

(v) $A_5 = \{((r, T), sink) : r \in N_r\}$ corresponding to exiting to the sink node at the end of the time horizon.

(vi) $A_6 = \{((r, t), sink) : \sigma^+(r) = \emptyset \text{ and } (r, t) \in N\}$ corresponding to exiting to the sink node from a node at the boundary of the area of track modelled.

An example of G is given in Figure 5.2. It uses the same track layout as the example in Figure 5.1.

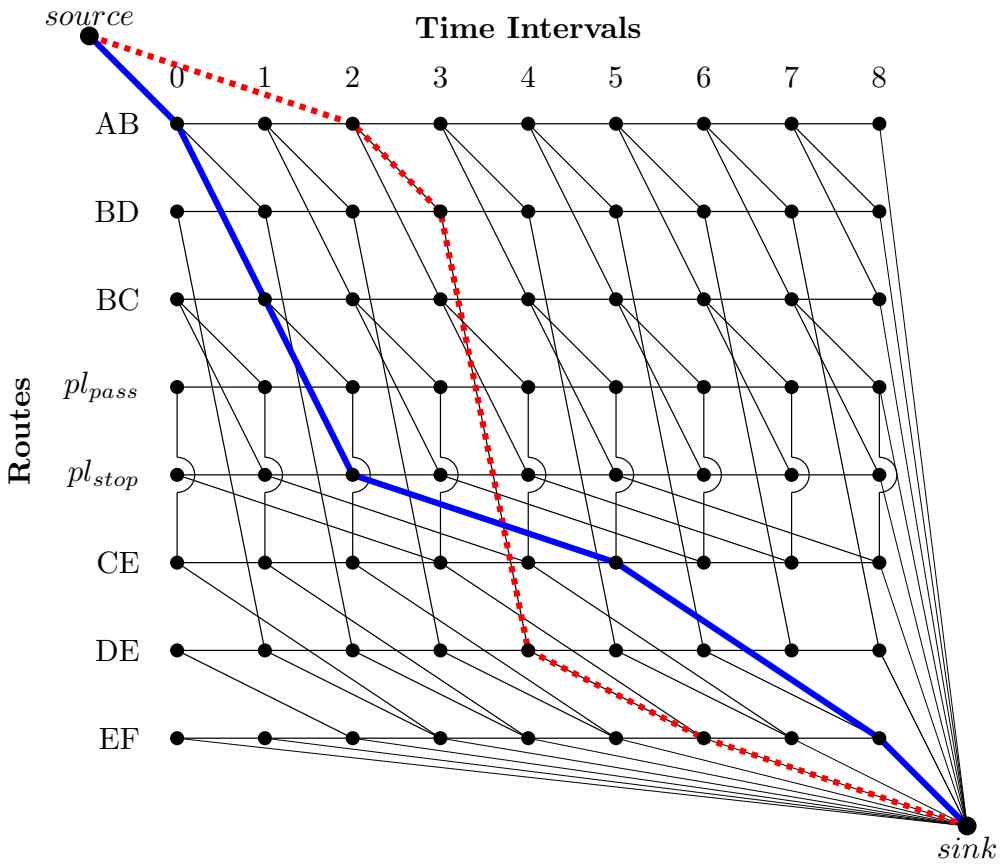


Figure 5.2: Example time-space graph G corresponding to the route graph example in Figure 5.1. Two *source-sink* paths are shown.

Note that a *source-sink* path in G specifies the routes that a train is momentarily traversing during every time interval from the current time up until the end of the time horizon. The routes to which a train is assigned also determine the platforms

that the train uses. The definition of A_4 ensures that minimum traversal and dwell times are respected. The arcs in A_3 make it possible for a train to wait for additional time at any point in its journey. Waiting should be implemented by making alterations to the speed profile of the train. These alterations should be calculated separately, and are not addressed in this paper.

The directed graph G is acyclic. It has $|N_r||\mathcal{T}|+2$ nodes and typically at least twice that many arcs depending on the track topology. In our experiments, G had 64,082 nodes and 156,594 arcs.

5.4.6 Track Capacity

For each train $k \in \mathcal{K}$, a finite-weight *source-sink* path in G corresponds to an *individually feasible* schedule over the time horizon. This means that such a path would be feasible if there were no other trains using the track. However, a set containing more than one train path is not *jointly feasible* unless it is possible for all of the train paths to be carried out together under the constraints imposed by the signalling system. The signalling system limits the capacity of the track in the interests of safety, as described in Section 5.2.

Each node $(r,t) \in N_0$ can be viewed as a time-space resource. We define two different ways in which a train can consume the capacity of a resource (r,t), called *occupying* and *banning*, in Sections 5.4.6.1 and 5.4.6.2. These definitions allow the track capacity constraints to be expressed using two constraints on the capacity of each time-space resource. These constraints are described in Sections 5.4.6.1 and 5.4.6.2. This novel representation of the track capacity constraints accurately models sectional-release interlocking, despite the fact that routes rather than track circuits are used in the time-space graph. Two sets, $A_{r,t}$ and $\bar{A}_{r,t}$, are defined for each resource (r,t) in Sections 5.4.6.3 and 5.4.6.4. These are constructed such that a train

path contains an arc in $A_{r,t}$ if and only if it occupies (r,t), and such that a train path contains an arc in $\bar{A}_{r,t}$ if and only if it bans (r,t).

5.4.6.1 Occupying

Before a train k can traverse a route r, all of the track circuits making up r must first be locked. This can only be done if all of the track circuits are free, i.e. they are not currently locked for another train. Once train k has traversed the route, the track circuits are released according to the release-type (either route-release or sectional-release). If any of the track circuits in route r that are locked for train k to traverse are still locked during time interval t, then we say that (r,t) is *occupied* by k. Occupation is one of the ways in which a train can consume the capacity of a time-space resource.

If (r,t) is occupied by train k, then it is not possible for any other train to occupy (r,t). This is because there are track circuits in r that have not yet been released by k and therefore are not free during time interval t. This gives rise to the first capacity constraint:

Capacity Constraint 1. *No $(r,t) \in N_0$ can be occupied more than once.*

5.4.6.2 Banning

Track circuits can and do belong to more than one route. For example, Figure 5.3 shows two routes, r_1 and r_2, that share track circuits 2 and 3. Suppose that train k locks the track circuits of route r_1 in order to traverse r_1. Since r_2 contains track circuits 2 and 3, which have now been locked for train k, r_2 is now unavailable to be locked by another train. This will remain true until k releases track circuits 2 and 3. If r_2 is unavailable for this reason during time interval t, we say that (r_2,t) is *banned*

by train k.

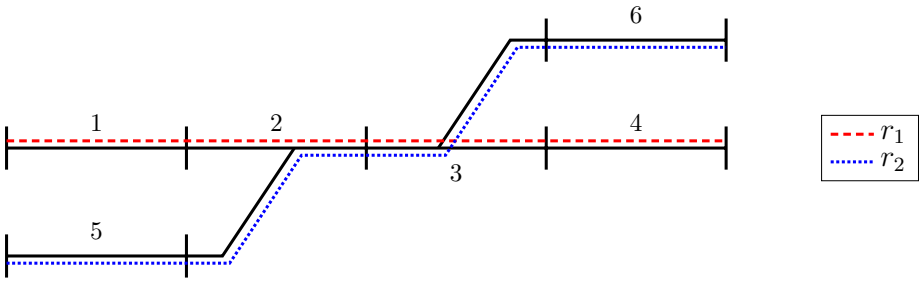


Figure 5.3: Routes r_1 and r_2 are shown by the dashed red and dotted blue lines, respectively, so that $TC_{r_1} = \{1, 2, 3, 4\}$, $TC_{r_2} = \{5, 2, 3, 6\}$. The two routes share the track circuits $TC_{r_1} \cap TC_{r_2} = \{2, 3\}$.

More generally, a train *bans* a time-space resource (r', t) if it makes route r' unavailable during time interval t as a result of occupying (r, t), where r is a route with track circuits in common with r'.

Notice that if (r', t) is banned by train k, then (r', t) cannot be occupied by k or any other train. This is the second capacity constraint:

Capacity Constraint 2. *No $(r, t) \in N_0$ can be both banned and occupied.*

There is no constraint on the number of times that a time-space resource can be banned. To understand why, consider routes r_1,r_2 and r_3 from the example in Figure 5.4. Since r_1 and r_3 share no track circuits, it is feasible for (r_1, t) to be occupied

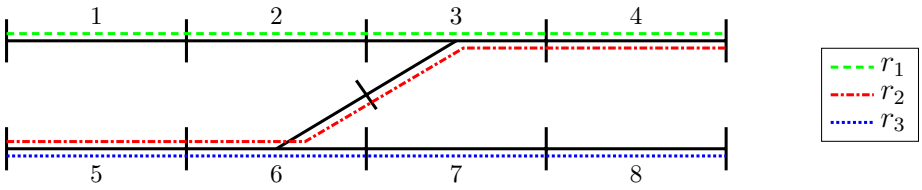


Figure 5.4: Routes r_1,r_2 and r_3 are shown by the dashed green, dot-dashed red and dotted blue lines, respectively, so that $TC_{r_1} = \{1, 2, 3, 4\}$, $TC_{r_2} = \{5, 6, 3, 4\}$ and $TC_{r_3} = \{5, 6, 7, 8\}$.

by one train, and for (r_3, t) to be occupied by a different train. Since r_2 shares track circuits with both r_1 and r_3, this may involve both of these trains banning

(r_2, t). However, no track circuit is locked more than once and therefore there is no infeasibility.

The difference between occupying and banning is subtle. Both occupying a resource and banning a resource preclude the resource from being otherwise occupied. The crucial difference is that whilst a resource cannot be occupied more than once, it may be banned more than once.

5.4.6.3 Construction of $A_{r,t}$

To enforce the capacity constraints, it is necessary to be able to determine whether a given *source-sink* path in G occupies a given time-space resource (r, t). This is achieved by constructing a set $A_{r,t}$ of arcs such that a path occupies (r, t) if and only if it contains an arc in $A_{r,t}$.

If a train occupies (r, t), then by definition there are track circuits in route r that are locked for the train during time interval t. In other words, the train is traversing r during time interval t. This traversal lasts for $L_r + h_r$ time intervals, which is the blocking time of route r. Therefore, the traversal began within the preceding $L_r + h_r$ time intervals, and the corresponding path in G must include a node in

$$W_{r,t} = \{(r, t') : t' \in \{t - (L_r + h_r) + 1, \ldots, t\}\}.$$

Since it must have entered this node via an arc, the path must contain an arc in

$$A_{r,t} = \bigcup_{(r',t') \in W_{r,t}} \sigma^-(r', t') \setminus \bigcup_{(r',t') \in W_{r,t}} \sigma^+(r', t').$$

The reverse is also true: if the path contains an arc in $A_{r,t}$, then it must contain a node in $W_{r,t}$ and therefore it must occupy the resource (r, t). Therefore, a train path

contains an arc in $A_{r,t}$ if and only if it occupies (r,t). An example of the set of nodes $W_{r,t}$ and the set of arcs $A_{r,t}$ corresponding to a time-space resource (r,t) are shown in Figure 5.5.

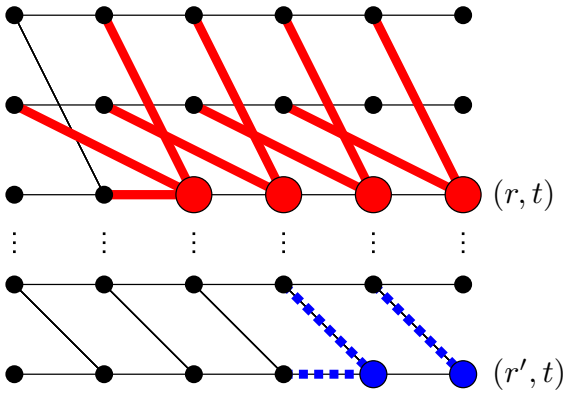


Figure 5.5: The nodes in $W_{r,t}$ and $\bar{W}_{r,t}$ are shown as large red and medium blue, respectively. The arcs in $A_{r,t}$ and $\bar{A}_{r,t}$ are shown as thick solid red and thick dotted blue, respectively. In the example, $L_r + h_r = 3$, $S_r = \{r'\}$ and $t(r', r) = 1$.

5.4.6.4 Construction of $\bar{A}_{r,t}$

We also show how to construct a set $\bar{A}_{r,t}$ of arcs such that a path bans (r,t) if and only if it contains an arc in $\bar{A}_{r,t}$. Let $S_r = \{r' \in N_r \setminus r : TC_r \cap TC_{r'} \neq \emptyset\}$ be the set of all routes with track circuits in common with r. In a route-release interlocking system, a train bans (r,t) if and only if it occupies (r',t), where $r' \in S_r$. This is because the track circuits common to r and r' are locked when r' is locked, and not released until route r' is released. Hence, r is prevented from being locked for the same time intervals as r' is locked.

Using the same reasoning as in the last section, we can say that a train bans (r,t) if and only if its path contains a node in

$$\bar{W}_{r,t} = \{(r', t') : r' \in S_r, t' \in \{t - t(r', r) + 1, \ldots, t\}$$

where $t(r', r) = L_{r'} + h_{r'}$. Therefore, a train bans (r,t) if and only if its path contains

an arc in

$$\bar{A}_{r,t} = \bigcup_{(r',t') \in \bar{W}_{r,t}} \sigma^-(r',t') \setminus \bigcup_{(r',t') \in \bar{W}_{r,t}} \sigma^+(r',t').$$

The situation is different in a sectional-release system. Because track circuits are released one by one as r' is traversed, the track circuits $TC_r \cap TC_{r'}$ common to r and $r' \in S_r$ can be released before all of the track circuits in r' are released. This means that there may be time intervals t during which a train occupies (r', t), but does not ban (r, t).

To reflect this difference, we must amend the definition of $t(r', r)$. When route $r' \in S_r$ is first locked, route r is prevented from being locked. Route r becomes available again when all of the track circuits in $TC_r \cap TC_{r'}$ are released. The last track circuit of these to be released as the train traverses r' is $tc_{r'}^{i(r',r)-1}$ where

$$i(r', r) = \min \left\{ i \in \mathbb{N} : \{tc_{r'}^j\}_{j=i}^{n_{r'}-1} \cap TC_r = \emptyset \right\}.$$

We define $\theta(r', r) \in [0, 1]$ to be the proportion of the route traversal time $L_{r'}$ after which this occurs. The time to release can therefore be written

$$t(r', r) = \lceil \theta(r', r) L_{r'} + h_{r'} \rceil.$$

Here, $\lceil \cdot \rceil$ indicates rounding up to the nearest integer number of time intervals. Rounding up ensures that the discretisation of time does not lead to an underestimation of the time for which track circuits are locked. Figure 5.5 shows an example of both $\bar{W}_{r,t}$ and $\bar{A}_{r,t}$ in a sectional-release system.

5.4.6.5 Antichain Condition

The method of representing the track capacity constraints that is described in the preceding sections relies on an important assumption. This assumption is described below and sufficient conditions for its validity are given.

Suppose that there is a *source-sink* path in G that includes arcs from both $A_{r,t}$ and $\bar{A}_{r,t}$, for some time-space resource (r,t). Any train assigned this path would, by definition of $A_{r,t}$ and $\bar{A}_{r,t}$, both occupy and ban (r,t). This would violate Capacity Constraint 2, which ensures that (r,t) cannot be both occupied and banned. Capacity Constraint 2 applies regardless of whether the resource is occupied and banned by the same train, or different trains. It is not possible within the real signalling system for a train to violate the track capacity single-handedly. Therefore, for the model to be correct, there should be no *source-sink* paths in G that include arcs from both $A_{r,t}$ and $\bar{A}_{r,t}$. This is stated mathematically in Assumption 1, which relies on Definition 1. Definition 1 is well known — for example, see [Hartzheim, 2005].

Definition 1. *An antichain in a directed graph $G = (N, A)$ is a set $Z \subseteq A$ of arcs such that each path in G contains at most one arc in Z.*

Assumption 1. *For each $(r,t) \in N_0$, $A_{r,t} \cap \bar{A}_{r,t} = \emptyset$ and $A_{r,t} \cup \bar{A}_{r,t}$ is an antichain in G.*

Proposition 1, which uses uses Definition 2, provides sufficient conditions for Assumption 1 to hold. Checking these sufficient conditions is easier than directly checking that Assumption 1 holds.

Definition 2. *For two routes $r_1, r_2 \in N_r$, let*

$$L(r_1, r_2) = \min \left\{ \sum_{r' \in p \setminus r_2} L_{r'} : p \text{ is an } r_1\text{-}r_2 \text{ path in } G_r \right\}$$

be the minimum total traversal time from r_1 up to but not including r_2. For example, if $(r_1, r_2) \in A_r$ then $L(r_1, r_2) = L_{r_1}$.

Proposition 1. *Assumption 1 is true if for each $r \in N_r$, either (i) $S_r \cup \{r\}$ is an antichain in G_r or (ii) $L(r_1, r_2) \geq L_{r_1} + h_{r_1}$ for each pair $r_1, r_2 \in S_r \cup \{r\}$.*

Proof. Let $(r, t) \in N_0$. Since $r \notin S_r$ by definition, $W_{r,t} \cap \bar{W}_{r,t} = \emptyset$ and hence $A_{r,t} \cap \bar{A}_{r,t} = \emptyset$.

Suppose for contradiction that at least one of (i) and (ii) holds, and that there is a *source-sink* path p in G that contains two distinct arcs $((r_0, t_0), (r_1, t_1))$ and $((r_2, t_2), (r_3, t_3))$ in $A_{r,t} \cup \bar{A}_{r,t}$, where without loss of generality $t_1 \leq t_3$. By the definitions of $A_{r,t}$ and $\bar{A}_{r,t}$, the routes r_1 and r_3 are in $S_r \cup \{r\}$. Since $t_1 \leq t_3$, there must be a subpath q of p from (r_1, t_1) to (r_3, t_3). If $S_r \cup \{r\}$ is an antichain in G_r, then this is a contradiction. Therefore, (ii) must hold. Subpath q starts at time t_1, ends at time t_3 and must span at least $L(r_1, r_3)$ time intervals. By definition of $A_{r,t}$ and $\bar{A}_{r,t}$, $t_1 \geq t - L_{r_1} - h_{r_1} + 1$ and $t_3 \leq t$. Putting this together, $t - L_{r_1} - h_{r_1} + 1 + L(r_1, r_3) \leq t$, which rearranges to $L(r_1, r_3) \leq L_{r_1} + h_{r_1} - 1$. This violates (ii), which proves the result by contradiction. □

It is easier to check the conditions of Proposition 1 than Assumption 1 directly. This is because they depend only on the route graph G_r, rather than the full time-space graph G. In our instances, condition (i) is satisfied for all routes, and this will be the case for most railways. Figure 5.6 shows an example of how Assumption 1 could fail to hold if the conditions of Proposition 1 are not satisfied.

5.4.6.6 Innovations of Proposed Approach

In existing approaches, occupying is the only type of capacity that is considered. Typically, the track is modelled as consisting of components that can be used by at

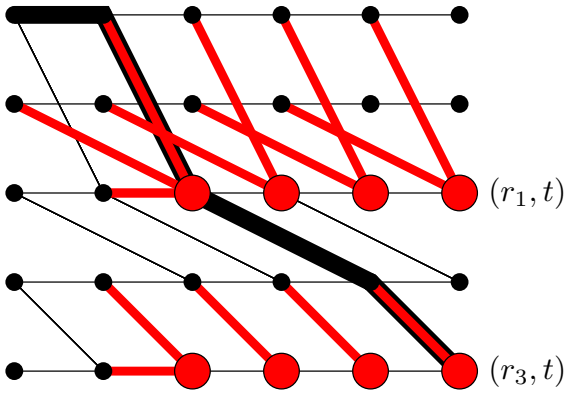


Figure 5.6: An example in which Assumption 1 does not hold. The arcs in $A_{r,t} \cup \bar{A}_{r,t}$ are shown as thick solid red lines, and $r_1, r_3 \in S_r \cup \{r\}$. The very thick black line is a path that contains two arcs in $A_{r,t} \cup \bar{A}_{r,t}$. Condition (i) does not hold, since there is an r_1-r_3 path in G_r. Condition (ii) does not hold because $L(r_1, r_2) = 2 \not> 3 = L_{r_1} + h_{r_1}$.

most one train at any one time. When these components are as large as block sections or routes, the track capacity is underestimated by the model. This is convincingly explained and empirically demonstrated by both Pellegrini et al. [2014] and Corman et al. [2009b].

In order to address this underestimation of track capacity, Pellegrini et al. [2014] and Corman et al. [2009b] have explicitly modelled track circuits. These models have decision variables that describe the arrival time of each train at each traversed track circuit. This approach is an accurate way of modelling the track capacity of a sectional-release interlocking area. However, there are two drawbacks to modelling all of the track circuits explicitly in this way. The first is that the large number of variables required to model interlocking areas with many track circuits can render models of this type intractable. The second drawback is that the solutions are difficult to implement in practice. This problem arises from the fact that track circuits are not individually signalled, and so the arrival of trains into individual track circuits cannot be controlled via signalling. If a solution specifies that a train must wait in a track circuit that does not immediately precede a signal, then a signaller cannot easily implement this.

Our model is able to accurately represent the track capacity of a sectional-release interlocking area without the drawbacks of explicitly modelling each track circuit. This is enabled by the consideration of banning as a second type of capacity in addition to occupying. Banning allows the release times of individual track circuits to be implicitly taken into account in the sense that although they could be calculated from a given solution, they are not associated with any decision variables. Using routes as the track components ensures that the model is tractable. This is because the size of the time-space graph depends on the number of routes in the area of track modelled rather than the number of track circuits, which is typically much greater. The use of routes rather than track circuits also ensures that solutions are clear and simple from an implementation perspective. The values of the decision variables in a solution indicate the time intervals in which trains should pass signals. This is something that signallers have full control over.

5.4.7 The Schedule

When railway traffic is perturbed due to primary delays, the schedule that was planned at the tactical phase may no longer be feasible. However, it is still the schedule that would ideally be carried out if possible. As a result, it is very important in determining the objective function.

Let $\mathcal{K}$ be the set of trains scheduled to pass though the controlled area during the time horizon. The schedule for each train $k \in \mathcal{K}$ can be represented by a sequence of triples $(r_j^k, a_j^k, d_j^k)_{j=0}^{J^k}$. In this notation, there are J^k *scheduled events* for train k within the area modelled during the time horizon. For event j, train k is scheduled to begin traversal of route $r_j^k \in N_r$ at time interval $a_j^k \in \mathcal{T}$, and depart route r_j^k at time interval $d_j^k \in \mathcal{T}$. Any values of a_j^k or d_j^k that lie outside the time horizon are ignored. Scheduled events can be *stopping events* or *passing events* depending on

whether or not the train is scheduled to stop to allow passengers to embark or alight. Passing events have a scheduled arrival time but do not have a scheduled departure time.

The initial route and time interval of train k are given by r_0^k and a_0^k. If the train is within the area at the beginning of the time horizon, then $a_0^k = 0$, and r_0^k is the route it is traversing at this time. Otherwise, r_0^k is the route on which train k first enters the area during the time horizon, and a_0^k is the time interval during which this occurs. In the latter case, a_0^k should be a forecast time. Using a forecast allows the model to take into account any anticipated delay in the arrival of the train into the controlled area. As a result, the model can suggest rescheduling solutions that pre-empt the effect of this anticipated delay, making rescheduling more effective.

Platform alternatives for the purpose of replatforming are inserted into the schedule. Suppose that (r_j^k, a_j^k, d_j^k) is a stop in the schedule, and that $\tilde{r}_j^k$ is an alternative platform to r_j^k of sufficient length to accommodate train k. Then $(\tilde{r}_j^k, a_j^k, d_j^k)$ is inserted into the schedule, in addition to (r_j^k, a_j^k, d_j^k). The set of all platform alternatives used for r_j^k must be an antichain in G_r. There is no constraint in the model on how many scheduled events each train carries out, but if the platform alternatives form an antichain in G_r then at most one of the platform alternative events can possibly occur.

5.4.8 The Objective Function

Most rescheduling models in the literature optimise a measure of performance that is inherent to the system. Though there are many examples (see [Samà et al., 2015]), the most prevalent objective function is to minimise a measure of train delays such as the maximum train delay. However, optimising a single inherent performance measure is inadequate because it fails to recognise that there may be many reasons for preferring

one rescheduling solution over another. This is especially true for models that allow scheduled events to be skipped, such as the one in this paper. These models require an objective function that can express the priority of minimising delay relative to minimising skipped events. In recognition of the inadequacy of minimising a single inherent performance measure, Corman et al. [2012a], Samà et al. [2015] and Binder et al. [2017] propose multicriteria methods for the TTRP. Whilst these methods take into account more than one performance measure, they are unsuitable for a real-time environment. One reason for this is that they do not achieve sufficiently fast computation times. In addition, it is not practical to require a railway signaller to choose between a set of solutions in real-time.

To overcome the challenges outlined above, the concept of utility has been used. Our objective is to maximise a utility function that models the rescheduling preferences of Network Rail. Accordingly, the objective function is the result of collaboration with Network Rail. The utility function is the total weighted utility over all trains k and scheduled events j, given by

$$U = \sum_{k \in \mathcal{K}} \alpha_k \sum_{j=1}^{J^k} \beta_k^j U_k^j(p^k).$$

In this formula, α_k is a weight reflecting the priority of train k, β_k^j is a weight reflecting the priority of event j for train k, and $U_k^j(p^k)$ is the utility accrued by train k at stop j when assigned *source-sink* path p^k in G.

Our use of total weighted utility over all train events is influenced by Network Rail's performance measure, *On-time at All Recorded Stations*. This measure is a summary of the lateness of each train event. It was adopted in 2019 in preference to the *Public Performance Measure*, which only takes into account lateness at each train's final scheduled event. Considering delay only at the final event of each train is unsatisfactory because it results in objective functions that are indifferent to delays at

intermediate stops, provided the time is subsequently recovered. It is also unsatisfactory to maximise the minimum utility over all train events, because this would correspond to indifference toward reductions in delay for all but the most delayed train. By modelling the utility as the total weighted utility over all trains and scheduled events, the objective function is never indifferent to a passenger delay reduction, and therefore reflects Network Rail's strategic priority of "putting passengers first" [Network Rail, 2019a, p. 7].

The utility $U_k^j(p^k)$ accrued by train k at event j if it is assigned *source-sink* path p^k in G is defined as

$$U_k^j(p^k) = \begin{cases} 0 & \text{if path } p^k \text{ does not visit } r_j^k \\ \gamma(t - a_j^k) & \text{if path } p^k \text{ enters } r_j^k \text{ at time interval } t, \end{cases}$$

where $\gamma : \mathbb{Z} \to \mathbb{R}^+$, depending on the number $l = t - a_j^k$ of time intervals late the train arrives, is given by

$$\gamma(l) = \begin{cases} \phi^{-\omega|l|} & \text{if } |l| \leq \Gamma \\ 0 & \text{if } |l| > \Gamma. \end{cases}$$

Negative values of l correspond to a train arriving early. The function γ is sketched in Figure 5.7, with the parameter values used in our experiments ($\phi = 1.0000001$, $\omega = 150,000$ and $\Gamma = 240$, corresponding to 1 hour).

The key features of this utility function are as follows:

- It depends on lateness, which is directly relevant to the passenger experience and therefore forms the basis of performance analysis within Network Rail. Using lateness is an improvement on the widely used measure of *consecutive delay*, defined as the additional delay created by rescheduling actions. Unlike

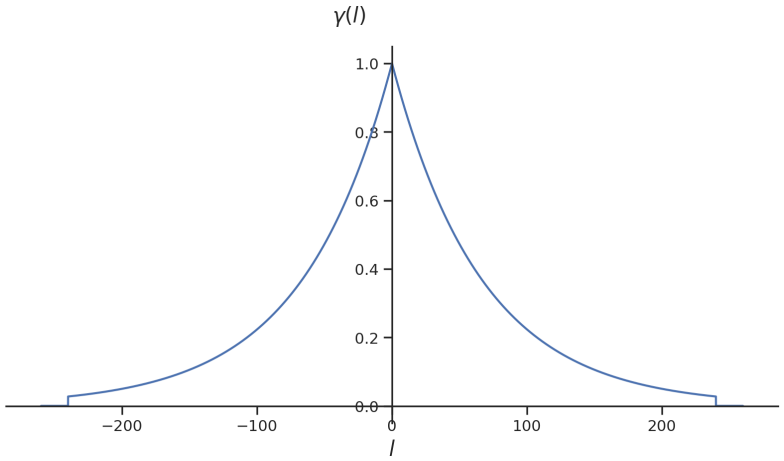


Figure 5.7: A sketch of γ.

lateness, consecutive delay is abstract to passengers.

- A preference is shown for events occurring as close to on-time as possible. This is achieved by making γ strictly increasing on $[-\Gamma, 0]$ and strictly decreasing on $[0, \Gamma]$. This preference reflects the fact that late events are bad for passengers. Whilst early events are not directly negative for passengers in the same way, they can cause unanticipated congestion at stations.

- A preference is shown for scheduled events occurring over not occuring, provided that they occur within Γ time intervals of the scheduled time. This is achieved by making $\gamma > 0$ on $[-\Gamma, \Gamma]$ and $U_k^j(p^k) = 0$ if the event does not occur. This reflects the fact that skipping scheduled stops is disruptive to passengers.

- The function γ is strictly convex on the intervals $[-\Gamma, 0]$ and $[0, \Gamma]$. This convexity reflects the idea that passengers are more sensitive to an additional time interval of lateness when a train is closer to on-time.

- Since the model is time-indexed, any set of values $\{\gamma(l) : l \in \mathbb{Z}\}$ could be used without compromising the tractability of the model. The objective function is therefore very flexible.

The value of α_k is set to 1 for express passenger (class 1) trains, and 0.4 for ordinary passenger (class 2) trains. The train weights α_k could also be used to take into account other factors affecting train priority such as the train's operator, the number of passengers or the number of remaining stops for the train once it has left the controlled area. However, these aspects are not considered in this paper.

The stop weights β_j^k are set as follows: A train's final arrival or passing event in the time horizon is given weight 0.7 to reflect the importance of trains leaving the controlled area punctually. All other arrivals excluding those representing platform alternatives receive an equal proportion of the remaining weight 0.3. Non-final passing events receive a weight of zero. When r_j^k is a platform alternative, β_j^k has the same weight as the planned platform arrival but discounted by a factor of 0.9. This reflects the disruption to passengers as a result of changing platforms. Other considerations could also be taken into account, such as the service patterns and passenger interchange importance of individual stations.

We will show in Section 5.4.9 how to set the arc weights in G to reflect the objective.

5.4.9 Arc Weights

The objective function is represented on the time-space graph G via the weights c_a^k for each train $k \in \mathcal{K}$ and each arc $a \in A$. These weights are set such that the total weight of a *source-sink* path for train k is equal to the negative of its total utility. Using the negative of the utility turns the problem into a minimisation problem. Note that we use the symbol ∞ to mean a real number sufficiently large that no good solution to the problem contains a path that uses an arc of this length — it is effectively a constraint.

The arcs in A_1 have weights $c^k_{(source,(r^j_0,a^j_0))} = \infty \mathbb{1}_{\{j \neq k\}}$ for each $k \in \mathcal{K}$ and $j = 0, \ldots, J^k$, where $\mathbb{1}$ is the indicator function. This ensures that trains cannot begin in the initial position of another train. The cancellation arc in A_2, has $c^k_{(source,sink)} = 0$ for all trains k if cancellations are to be included in the model, and $c^k_{(source,sink)} = \infty$ otherwise. All of the arcs a in A_3, A_5 and A_6 have $c^k_a = 0$ for all k, because the sink is artificial and waiting does not contribute to the modelled utility function.

Arcs a in A_4 have $c^k_a = 0$ for all k except in two very important cases:

- **Departures**:

  For $k \in \mathcal{K}$, $j = 0, \ldots, J^k$ and $t < d^k_j$, if $a = ((r^k_j, t), (r', t')) \in A_4$ then $c^k_a = \infty$. This prevents trains departing from a stop early if they visit that stop.

- **Arrivals**:

  For $k \in \mathcal{K}$ and $j = 0, \ldots, J^k$, if $a = ((r', t'), (r^k_j, t)) \in A_4$ then $c^k_a = -\alpha_k \beta^j_k \gamma(t - a^k_j)$. Such an arc is included in a train path if and only if event j for train k occurs at time t. The negative of the associated utility is accrued.

Assumption 2 must hold, so that it is never possible for a train path to collect more than one positive reward for carrying out a scheduled event more than once.

Assumption 2. $\Gamma < L(r^k_j, r^k_j)$ *for all* $k \in \mathcal{K}, j = 0, \ldots, J^k$.

This is only a relevant consideration when short cycles containing platforms exist in G_r. There are no cycles containing platform routes in our instances, so we are free to pick any value of Γ.

5.5 Integer Programming Formulation

5.5.1 Arc Formulation

The TTRP, defined in Section 5.4.2, can be formulated as an Integer Linear Program similarly to the classical multicommodity flow problem. For each train $k \in \mathcal{K}$ and arc $a \in A$, we introduce a binary variable x_a^k which takes the value 1 if and only if the path in G corresponding to train k includes the arc a. The formulation is as follows:

$$\min \quad \sum_{k \in \mathcal{K}} \sum_{a \in A} c_a^k x_a^k \tag{5.1a}$$

$$\text{s.t.} \quad \sum_{k \in \mathcal{K}} \left(\sum_{a \in A_{r,t}} x_a^k + \delta \sum_{a \in \bar{A}_{r,t}} x_a^k \right) \leq 1 \quad \forall (r,t) \in N_0 \tag{5.1b}$$

$$\sum_{a \in \sigma^+(n)} x_a^k - \sum_{a \in \sigma^-(n)} x_a^k = b_n^k \quad \forall k \in \mathcal{K}, n \in N \tag{5.1c}$$

$$x_a^k \in \{0,1\} \quad \forall k \in \mathcal{K}, a \in A. \tag{5.1d}$$

where $b_{source}^k = 1$, $b_{sink}^k = -1$ for all $k \in \mathcal{K}$, $b_n^k = 0$ for all $n \in N_0$ and $k \in \mathcal{K}$, and $\delta > 0$ is a small positive constant. The notation $\sigma^+(n)$ and $\sigma^-(n)$ denotes the set of arcs starting and ending at node n, respectively.

The objective (5.1a) minimises the total weight of the paths. The constraints have a block-diagonal structure: (5.1b) are linking constraints enforcing the track capacity, whilst the flow constraints (5.1c) form one block per train. For each train $k \in \mathcal{K}$, the corresponding block of flow constraints ensures that the solution for train k corresponds to a *source-sink* path in G.

The track capacity constraints are expressed by (5.1b). If a train k occupies a resource $(r,t) \in N_0$, then $x_a^k = 1$ for some $a \in A_{r,t}$ and the constraint ensures that (r,t) cannot otherwise be occupied or banned. If no train occupies $(r,t) \in N_0$, then up to $\lfloor \frac{1}{\delta} \rfloor$ trains may ban it. We make δ sufficiently small that this limit is never exceeded in practice. In our experiments, $\delta = 0.05$ was sufficient.

This formulation has $|\mathcal{K}||A|$ binary variables and $|N_0|+|\mathcal{K}||N|$ constraints. A typical instance in our computational experiments has around 3,500,000 binary variables and around 1,500,000 constraints. We could not even obtain feasible solutions using the generic Mixed Integer Programming solver Gurobi because the instances were too large.

5.5.2 Danzig-Wolfe Decomposition into a Path Formulation

Decomposition is a popular approach for integer programs that are prohibitively large to solve using general purpose solvers. Formulation (5.1a)–(5.1d) is particularly amenable to Danzig-Wolfe decomposition because the constraint matrix has a block-diagonal structure where each block is the constraint matrix of a shortest path problem on a directed acyclic graph.

We perform Danzig-Wolfe decomposition using the discretisation approach presented by Vanderbeck and Wolsey [2010]. For each train $k \in \mathcal{K}$, define

$$X^k = \left\{ \mathbf{x}^k \in \{0,1\}^{|A|} : \sum_{a \in \sigma^+(n)} x_a^k - \sum_{a \in \sigma^-(n)} x_a^k = b_n^k \quad \forall n \in N \right\}$$

to be the set of feasible paths, and P^k to be its finite index set so that $X^k = \{\mathbf{x}^{k,p} : p \in P^k\}$. We can rewrite variables $\mathbf{x}^k$ in terms of the constants $\mathbf{x}^{k,p}$ and new binary

variables $\lambda^{k,p}$:

$$\mathbf{x}^k = \sum_{p \in P^k} \lambda^{k,p} \mathbf{x}^{k,p}, \quad \sum_{p \in P^k} \lambda^{k,p} = 1, \quad \lambda^{k,p} \in \{0,1\} \quad \forall p \in P^k.$$

This variable transformation makes the constraint $\mathbf{x} \in X^k$ implicit, so when we substitute the new variables into formulation (5.1a)–(5.1d), constraint (5.1c) can be dropped. This results in the path formulation:

$$\min \quad \sum_{k \in \mathcal{K}} \sum_{p \in P^k} \left(\sum_{a \in A} c_a^k x_a^{k,p} \right) \lambda^{k,p} \tag{5.2a}$$

$$\text{s.t.} \quad \sum_{k \in \mathcal{K}} \sum_{p \in P^k} \left(\sum_{a \in A_{r,t}} x_a^{k,p} + \delta \sum_{a \in \bar{A}_{r,t}} x_a^{k,p} \right) \lambda^{k,p} \leq 1 \quad \forall (r,t) \in N_0 \tag{5.2b}$$

$$\sum_{p \in P^k} \lambda^{k,p} = 1 \quad \forall k \in \mathcal{K} \tag{5.2c}$$

$$\lambda^{k,p} \in \{0,1\} \quad \forall k \in \mathcal{K}, p \in P^k. \tag{5.2d}$$

Formulations (5.1a)–(5.1d) and (5.2a)–(5.2d) are equivalent. Applying the classical results of Geoffrion [1974], the linear relaxations of these formulations provide the same lower bound on their optimal values, which is the same bound provided by Lagrangian relaxation of (5.1a)–(5.1d). This is because all of the extreme points of the linear relaxation of each X^k are integral.

Formulation (5.2a)–(5.2d) has $|\mathcal{K}|+|N_0|$ constraints, which can be significantly fewer than (5.1a)–(5.1d). Although formulation (5.2a)–(5.2d) typically has many more variables, we know that each integer feasible solution will have exactly $|\mathcal{K}|$ non-zero variables. As a result, (5.2a)–(5.2d) can be effectively solved using a branch-and-price algorithm.

5.6 Branch-and-Price Algorithm

Formulation (5.2a)–(5.2d) can be solved using a branch-and-price algorithm. This is a branch-and-bound algorithm in which the LP relaxation at each node is solved by column generation. In Sections 5.6.1 and 5.6.2 we describe how column generation can be used to solve the LP relaxation of (5.2a)–(5.2d). Two acceleration strategies are described in Sections 5.6.3 and 5.6.4. Finally, in Section 5.6.5, we describe our branching scheme.

5.6.1 Column Generation

Here we outline the basic column generation algorithm (see Desrosiers and Lübbecke [2005] for a more general explanation) for solving the LP relaxation of (5.2a)–(5.2d), which is referred to as the Master Problem (MP).

In iteration n, we restrict P^k to a much smaller subset P^k_n, resulting in a smaller LP called the Restricted Master Problem (RMP). Solving the RMP yields an optimal primal solution $\boldsymbol{\lambda}^* = (\lambda^{*k,p})_{k\in\mathcal{K},p\in P^k_n}$ and optimal dual values $\boldsymbol{\rho}^* = (\rho^*_k)_{k\in\mathcal{K}}$ corresponding to constraints (5.2c), and $\boldsymbol{\pi}^* = (\pi^*_{r,t})_{(r,t)\in N_0}$ corresponding to constraints (5.2b).

These dual values can be used to find a column $p \in P^k \setminus P^k_n$ of minimum reduced cost for each $k \in \mathcal{K}$. If none of these columns have negative reduced cost, then $\boldsymbol{\lambda}^*$ is optimal for the MP, which is now solved. Otherwise, at least one column with negative reduced cost must be added to the RMP (i.e. $P^k_n \subset P^k_{n+1}$) and the algorithm proceeds to iteration $n+1$.

The reduced cost of a column corresponding to path $p \in P^k$ for train k is

$$\begin{aligned}\bar{c}^{k,p} &= \sum_{a\in A} c_a^k x_a^{k,p} - \sum_{(r,t)\in N_0} \left(\sum_{a\in A_{r,t}} x_a^{k,p} + \delta \sum_{a\in \bar{A}_{r,t}} x_a^{k,p} \right) \pi_{r,t}^* - \rho_k^* \\ &= \sum_{a\in A} c_a^k x_a^{k,p} - \sum_{(r,t)\in N_0} \sum_{a\in A} \left(\mathbb{1}_{\{a\in A_{r,t}\}} + \delta \mathbb{1}_{\{a\in \bar{A}_{r,t}\}} \right) x_a^{k,p} \pi_{r,t}^* - \rho_k^* \\ &= \sum_{a\in A} (c_a^k - \pi_a^*) x_a^{k,p} - \rho_k^*\end{aligned} \tag{5.3}$$

where

$$\pi_a^* = \sum_{(r,t)\in N_0} \left(\mathbb{1}_{\{a\in A_{r,t}\}} + \delta \mathbb{1}_{\{a\in \bar{A}_{r,t}\}} \right) \pi_{r,t}^* \tag{5.4}$$

is the dual value for arc a, and $\mathbb{1}$ is the indicator function.

The task of finding the minimum reduced cost column for train k, known as the k^{th} subproblem (SP^{k}), therefore amounts to finding a shortest *source-sink* path in G where each weight c_a^k is replaced by $c_a^k - \pi_a^*$.

5.6.2 Subproblem Solution

Solving the subproblem for each train amounts to finding a *source-sink* path in G with modified weights. Because G is directed and acyclic, we can use the classical algorithm of Kahn [1962]. This is an exact labelling algorithm that first sorts the nodes $n \in N$ into a topological order $N_{top} = (n_0, \ldots, n_{|N|})$ so that $(n_i, n_j) \in A \iff i < j$. The node labels are initialised to ∞ and the nodes are scanned once in topological order starting at the node *source*. At each node $n_i \in N$, the directed arcs starting at n_i are relaxed in turn. An arc $(n_i, n_j) \in A$ is relaxed by checking whether the sum of the label at n_i and the weight of (n_i, n_j) is smaller than the label at node n_j. If so, the label of node n_j is updated to this smaller value, and n_i is recorded as the predecessor of n_j. When the node *sink* is reached in the scan of N_{top},

the algorithm backtracks through the succession of recorded predecessors of the node *sink* to yield the shortest *source-sink* path in G. This algorithm is of complexity $\mathcal{O}(|A|)$.

Note that although the subproblem is solved multiple times in the column generation algorithm, the topological ordering N_{top} needs to be calculated only once at the beginning of the entire solve process. This ordering remains valid for every subproblem.

5.6.3 Acceleration Strategy 1: Partial Pricing

To improve the convergence of the column generation algorithm, we use partial pricing (see Desaulniers et al. [2002]). This popular technique involves solving only a subset of the subproblems in each pricing round. If no negative reduced cost columns are found from this subset then a full pricing round is used to determine whether the MP is solved. This causes the dual values to converge more quickly and hence reduces the total number of subproblems that must be solved, at the expense of potentially solving more RMPs. Empirical evidence suggests that solving subproblems in three equally sized groups achieves the best performance.

5.6.4 Acceleration Strategy 2: Reduced Cost Variable Fixing

Reduced cost variable fixing (see Irnich et al. [2010]) has also been used to accelerate the column generation algorithm. Reduced cost variable fixing is based on the observation that if the reduced cost $\bar{c}_a^k$ of an original variable x_a^k exceeds the integrality gap $UB - LB$, then arc a cannot appear in an optimal path p^k for train k. LB and UB can be any lower or upper bounds on the optimal objective value, respectively.

At each execution of the reduced cost fixing procedure, UB is set to the objective of the current incumbent primal feasible solution. At each execution, LB is set to the sum of the RMP objective value and $|\mathcal{K}| \min\{rc^k : k \in \mathcal{K}\}$, where rc^k is the reduced cost for subproblem k (see [Desrosiers and Lübbecke, 2005, p. 11]).

Any arc a that cannot appear in an optimal path p^k for train k can be eliminated from subproblem k. In practice, this is achieved by setting its weight c_a^k to ∞, where ∞ is a number sufficiently large to prevent arc a ever forming part of an optimal path p^k for train k. Eliminating these arcs prevents the generation of columns that cannot form part of an optimal integer solution. This causes the column generation algorithm to discover good quality integer feasible solutions more quickly.

Directly translating the results from Irnich et al. [2010], the reduced cost $\bar{c}_a^k$ of variable x_a^k, where $a = (i, j)$, is calculated from an optimal dual solution $(\boldsymbol{\pi}^*, \boldsymbol{\rho}^*)$ as

$$\bar{c}_a^k = c_a^k - \pi_a^* + l_i - l_j$$

where l_i is the length of the shortest *source-i* path in G when the arc weights are modified to $(c_a^k - \pi_a^*)$. These shortest path distances are available from the graph labels after solving the subproblem and can be extracted with little computational effort. Reduced cost variable fixing in a given subproblem is therefore of complexity $\mathcal{O}(|A|)$.

Our computational experiments show that reduced cost variable fixing is only effective when the integrality gap is small, for example under 5%. Therefore, the procedure is not executed until the root node is solved, and after that it is executed each time a new best primal solution is found.

5.6.5 Branching

An optimal solution to the linear relaxation of (5.2a)–(5.2d) may contain fractional variables and therefore not be integer feasible for (5.2a)–(5.2d). The column generation algorithm is therefore embedded inside a branch-and-bound algorithm, resulting in a branch-and-price algorithm.

The following result characterises the possible fractionalities by showing that a fractional variable always arises as a result of a track capacity conflict between two different trains.

Definition 3. *A solution $\boldsymbol{\lambda}$ to an RMP has a fraction-inducing conflict if there exists an active constraint $(r,t) \in N_0$ in the RMP containing both a fractional variable $\lambda^{k_0,p_0} \in (0,1)$ and a non-zero variable $\lambda^{k_1,p_1} \in (0,1]$ with $k_0 \neq k_1$. We denote the fraction-inducing conflict $(r,t,\lambda^{k_0,p_0},\lambda^{k_1,p_1})$.*

Proposition 2. *If $\boldsymbol{\lambda}$ is a fractional, basic solution of the RMP that is optimal for the MP, then $\boldsymbol{\lambda}$ has a fraction-inducing conflict.*

Moreover, if a fractional variable has minimal objective coefficient among all basic variables for its train, and these objective coefficients are not all equal, then it is part of a fraction-inducing conflict.

Proof. Suppose $\boldsymbol{\lambda}$ is fractional. Then $\exists k \in \mathcal{K}$ and $p \in P^k$ such that $\lambda^{k,p} \in (0,1)$.

Letting $\{\lambda^{k,\hat{p}} : \hat{p} \in \hat{P}^k\}$ be the set of variables for train k taking fractional values, we know by constraints (5.2c) that $|\hat{P}^k| \geq 2$.

For each $\hat{p} \in \hat{P}^k$, let $\boldsymbol{\lambda}_{\hat{p}}$ be the solution obtained from $\boldsymbol{\lambda}$ by replacing

$$\lambda^{k,\hat{p}} \leftarrow \lambda^{k,\hat{p}} + \epsilon \text{ and } \lambda^{k,q} \leftarrow \lambda^{k,q} - \frac{\epsilon}{|\hat{P}^k| - 1}$$

for all other $q \in \hat{P}^k \setminus \{\hat{p}\}$, where $\epsilon > 0$ is a small constant.

By assumption, $\boldsymbol{\lambda}$ is a basic solution and hence an extreme point of the feasible set of the RMP. However, it can also be written as a sum

$$\boldsymbol{\lambda} = \frac{1}{|\hat{P}^k|} \sum_{\hat{p} \in \hat{P}^k} \boldsymbol{\lambda}_{\hat{p}}$$

of the distinct solutions $\boldsymbol{\lambda}_{\hat{p}}$. Therefore, $\exists p_0 \in \hat{P}^k$ for which $\boldsymbol{\lambda}_{p_0}$ is infeasible.

By construction, $\boldsymbol{\lambda}_{p_0}$ satisfies (5.2c) and so $\boldsymbol{\lambda}_{p_0}$ must violate (5.2b). In other words, there is a capacity constraint $(r,t) \in N_0$ that is active for $\boldsymbol{\lambda}$ and violated by $\boldsymbol{\lambda}_{p_0}$. Constraint (r,t) would not be violated by $\boldsymbol{\lambda}_{p_0}$ if it contained only variables in $\hat{P}^k$, so it must contain another variable $\lambda^{k_1,p_1} \in (0,1]$ with $k_1 \neq k_0$, which takes the same value in $\boldsymbol{\lambda}$.

Note that if $c^{k,p} = \min\{c^{k,\hat{p}} : \hat{p} \in \hat{P}^k\}$ and $\exists q \in \hat{P}^k$ such that $c^{k,p} < c^{k,q}$, then $\boldsymbol{\lambda}_p$ as constructed above is infeasible by the optimality of $\boldsymbol{\lambda}$, and so we can set $p_0 = p$.

□

One consequence of this result is that instances with no conflicts, for example instances in which there is no primary delay, produce integral LP relaxations. More generally, we would expect the number of fractional variables to be correlated with the number of conflicts in the instance.

To devise a branching rule, we must first understand the different types of fraction-inducing conflicts that can arise:

Definition 4. *Given a fraction-inducing conflict* $(r,t,\lambda^{k_0,p_0},\lambda^{k_1,p_1})$ *for* $\boldsymbol{\lambda}$*, let* $y_{r,t}^{k_0,p_0}$ *and* $y_{r,t}^{k_1,p_1}$ *be the coefficients of* λ^{k_0,p_0} *and* λ^{k_1,p_1} *in constraint* (r,t)*. These are equal to* 1 *if the path corresponding to the variable occupies* (r,t)*, or* δ *if it bans* (r,t)*. Then the fraction-inducing conflict is, respectively, of type 'OO', 'OB', 'BO' or 'BB' if the*

pair $(y_{r,t}^{k_0,p_0}, y_{r,t}^{k_1,p_1})$ *is equal to* $(1,1),(1,\delta),(\delta,1),(\delta,\delta)$.

Proposition 3. *Under the conditions of Proposition 2,* $\boldsymbol{\lambda}$ *has a fraction-inducing conflict of type 'OO' or 'OB'.*

Proof. By Proposition 2, $\boldsymbol{\lambda}$ has a fraction-inducing conflict $(r, t, \lambda^{k_0,p_0}, \lambda^{k_1,p_1})$. It must be of one of the four types, 'OO', 'OB', 'BO' or 'BB'. If it is of type 'OO' or 'OB', then the result is immediately true.

Suppose that $(r, t, \lambda^{k_0,p_0}, \lambda^{k_1,p_1})$ of type 'BO'. Since $\delta\lambda^{k_0,p_0} > 0$, we know from constraint (r, t) that $\lambda^{k_1,p_1} < 1$. Consequently, $(r, t, \lambda^{k_1,p_1}, \lambda^{k_0,p_0})$ is a fractional-inducing conflict of type 'OB'.

Finally, suppose that $(r, t, \lambda^{k_0,p_0}, \lambda^{k_1,p_1})$ is of type 'BB'. Since constraint (r, t) is active, and δ was chosen sufficiently small that in practice fewer than $\lceil \frac{1}{\delta} \rceil$ trains can ban it, constraint (r, t) must contain a variable $\lambda^{k_2,p_2} > 0$ with $y_{r,t}^{k_2,p_2} = 1$. Therefore, $(r, t, \lambda^{k_0,p_0}, \lambda^{k_2,p_2})$ is a fractional-inducing conflict of type 'BO'. Using the argument in the preceding paragraph, $(r, t, \lambda^{k_2,p_2}, \lambda^{k_0,p_0})$ is of type 'OB'.

□

Proposition 3 establishes that the following branching is always available:

Branching Rule 1.

(L)	k_0 *cannot occupy* (r, t)	(R)	k_1 *cannot occupy* (r, t)	*if type 'OO'*
(L)	k_0 *cannot occupy* (r, t)	(R)	k_1 *cannot ban* (r, t)	*if type 'OB'*

Branching Rule 1 breaks the conflict by preventing k_0 and k_1 from occupying or banning (r, t) in the left and right hand branches, respectively, according to the type of conflict. This leaves (r, t) free for the other train in each branch. It should be noted that this branching rule is a novel adaptation of constraint branching, which

was first suggested by Ryan and Foster [1981].

This branching is *compatible* with the subproblem, meaning that the additional constraint created by each branch can be enforced in the subproblem without compromising the efficiency of its solution algorithm. To prevent a train k from occupying a resource (r,t), we set the arc weights $c_a^{k_0} = \infty$ for each $a \in A_{r,t}$, and to prevent banning we similarly set $c_a^{k_0} = \infty$ for each $a \in \bar{A}_{r,t}$. The subproblems remain shortest path problems on a directed acyclic graph.

A practical algorithm for identifying a fraction-inducing conflict is given in Algorithm 1. We choose to sort the variables by $c^{k,p}$ in ascending order for two reasons. First, variables with low objective value are more likely to have the lowest objective value among all basic variables for their train and hence, by Proposition 2, yield a fraction-inducing conflict. Secondly, variables with low objective value are likely to be more important for establishing good primal and dual bounds, because they represent the most desirable paths.

Algorithm 1 Conflict Branching

```
for (k_0, p_0) ∈ {(k,p) : λ^{k,p} ∈ (0,1)} sorted by c^{k,p} ascending do
    for (r,t) ∈ N_0 such that λ^{k_0,p_0} in constraint (r,t) do
        for λ^{k_1,p_1} in constraint (r,t) do
            if k_1 ≠ k_0 and λ^{k_1,p_1} ∈ (0,1] then
                return (r, t, k_0, k_1)
            end if
        end for
    end for
end for
```

5.7 Instance Data

We have developed a new set of 310 instances for the TTRP to test our approach. These instances are based on real data from an area of railway around Doncaster

station in the UK. It is of critical importance that instances used to test rescheduling algorithms are realistic, because the difficulty can depend heavily on the instance. For our model, Proposition 2 shows that there is a direct link between the amount of track capacity conflict in a given instance and the strength of the formulation for this instance. Important aspects of the instance generation are described below.

5.7.1 Track Data

We have chosen to use an area around Doncaster station for our instances. Doncaster station lies on the East Coast Main Line, a busy railway corridor connecting London with Leeds, York, Newcastle and Edinburgh. It is an important interchange for inter-city services and terminus for local services, with $3,857,000$ passenger entries and exits in 2017–2018 and over 30 trains per hour at peak times. Doncaster is a bottleneck which is responsible for reactionary delay to trains on the main line. This makes it ideal for testing our method.

Doncaster station, shown in Figure 5.8, has 9 platforms and 85 track circuits. It is bounded by 17 signals, and has 17 signals inside the station. In addition to the mainline that runs through the station, 4 double track lines start at Doncaster, going to Sheffield, Lincoln, Leeds and Hull, respectively. There are several heavily used routes within the station which cross the main lines and create conflicts. A good example is the route into platform 1, going from berth 0302 to berth 0278 (see Figure 5.8). The wider area covered is shown in Figure 5.9. It contains portions of each of the lines mentioned, but lies within a single area of signalling control. It consists of 225 berths with 313 valid berth transitions.

In order to infer the track data, we used 18 months of historical data from a Train Describer (TD), one of Network Rail's real-time information systems. Train Describers record, directly from the track circuits, the time at which every berth transition oc-

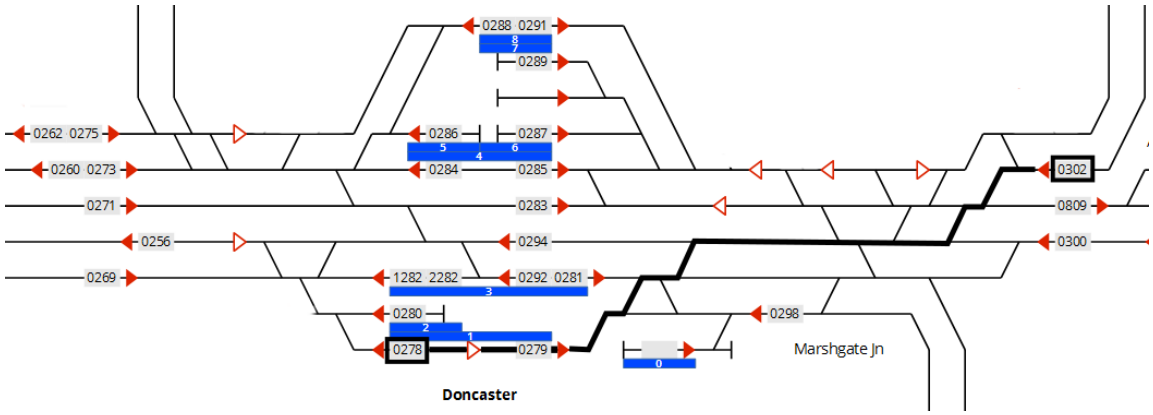


Figure 5.8: A berth diagram of Doncaster Station. The route from berth 0302 to berth 0278 is highlighted. Retrieved from `https://wiki.openraildata.com/index.php?title=DR`. Accessed 14/04/20.

curs within a particular area of signalling control. From this data, we were able to deduce the set N_b of berths within the area, and set A_b of permitted berth transitions. A significant amount of data cleaning was required to remove incorrectly recorded berth transitions, and berth transitions that are allowed only in exceptional circumstances. From N_b and A_b, we were able to build route graph G_r as described in Section 5.4.4.

Data about the track circuits TC_r included in each route $r \in N_r$ were not available digitally, so we recorded them manually from Network Rail drawings. The minimum dwell time D_{pl} was taken from timetable planning rules to be 120 seconds for all platforms. The headway times h_r and H_{pl} were estimated at 30 seconds for every route and platform, respectively.

5.7.2 Traversal Time Estimation

For each route $r \in N_r$, we estimated the traversal time $L_r \in \mathbb{Z}^+$ from the historical observations $\mathbf{y}^r$ in the TD data. These values play a crucial role in the track capacity constraints, and hence in determining how difficult the instances are to solve. By

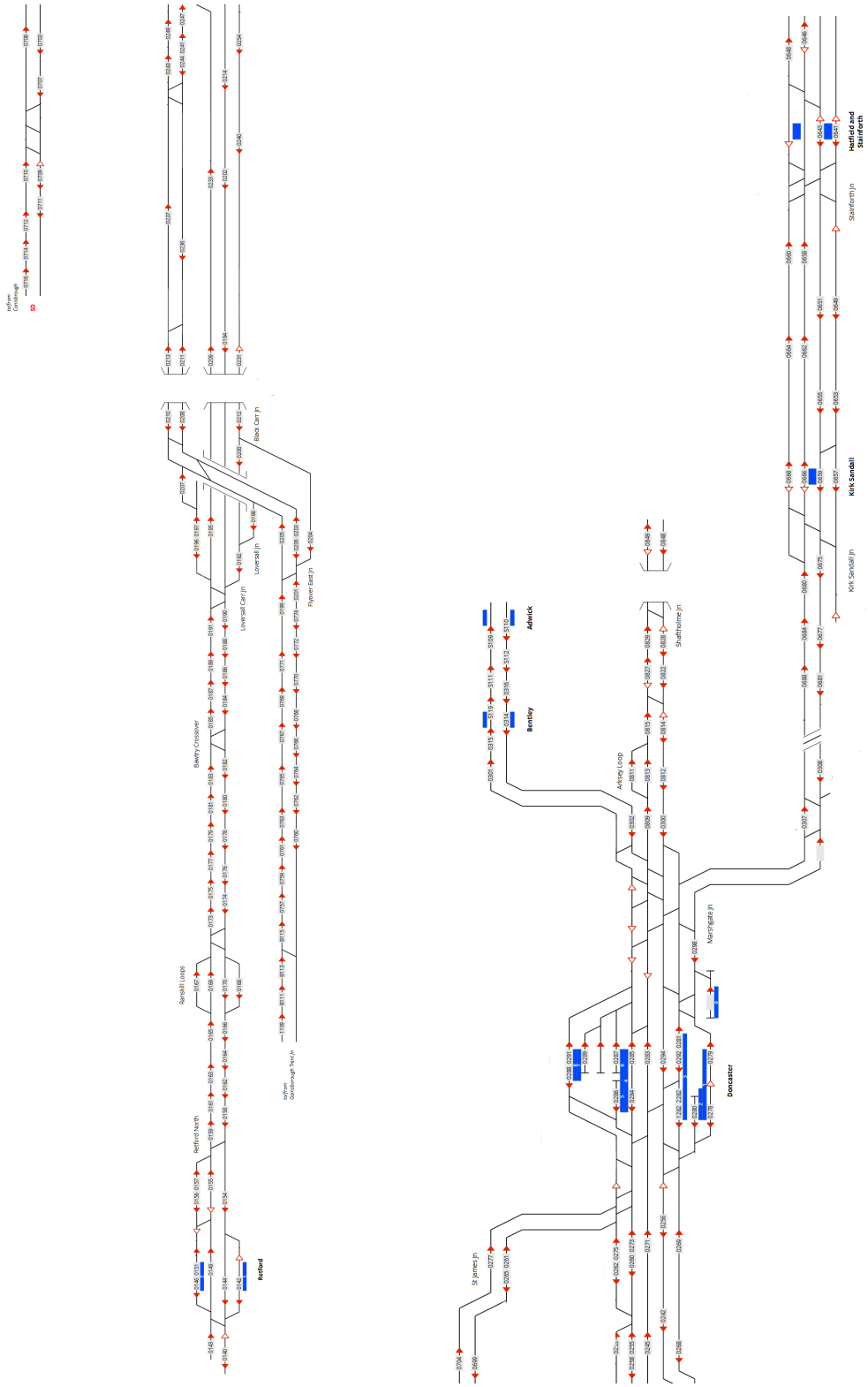


Figure 5.9: A berth diagram of the area of track modelled. Retrieved from `https://wiki.openraildata.com/index.php?title=DR`. Accessed 14/04/20.

estimating them from the historical data, we were able to overcome any need for data about distances, speed limits and rolling stock characteristics.

We first estimate the time $l_r \in \mathbb{R}^+$ in seconds, and then calculate L_r by rounding l_r up to the nearest whole number of time intervals. Although this introduces approximation error depending on the size of the time interval, rounding up ensures that solutions to the model are not unachievable in practice.

For some routes, the observed times in $\mathbf{y}^r$ can be seen to arise from two different processes: the transition times of trains travelling close to the line speed, and the transition times of trains that were not close to the line speed. This second process consists of trains which were slowed or even stopped altogether by a signal, and trains which stopped at platforms during their traversal of the route. Since we do not know which observations are from which process, we fit a Gaussian mixture model with two components to cluster the observations into these two processes. We fit the Gaussian mixture model with the EM algorithm, and take l_r to be the minimum of the means of the components. See Bouveyron et al. [2019] for an introduction to Gaussian mixture models and the EM algorithm.

Many routes, for example those on the open line, appear to have only one mixture component. To decide whether to use one component or two, we fit both models and choose the model that optimises the Bayesian Information Criterion [Bouveyron et al., 2019, p .51]. When there is just one component, we take l_r to be the mean of this component.

5.7.3 Timetable Data

There are 310 instances in the test set. Each one covers a different hour long period starting on the hour between 8am - 6pm during January 2017 (10 instances per

day over 31 days). They were created using the actual timetables and actual traffic perturbations during these times. We consider the use of real traffic perturbation scenarios to be crucial in assessing the algorithm fairly, despite the fact that it is not universal practice in the published literature.

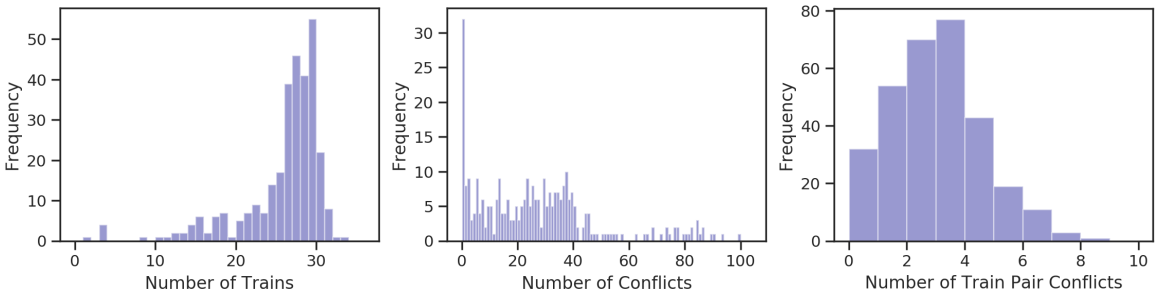
The data was collected from TRUST (Train Running Under System TOPS), another information system used by Network Rail. This system is used to compare the timetable with actual performance and is widely used within the organisation for performance analysis. Location data in the timetable, and therefore TRUST, is recorded at the level of stations and junctions (called *timing points*), with platforms and directions specified where appropriate. Directions are recorded as 'up' or 'down', and correspond to the two directions in which the most important railway line passing through a given station can be traversed.

By creating a mapping from each combination of timing point, direction and platform to the corresponding berth, we were able to reconcile our track data with the TRUST data. This mapping was created automatically by linking trains in the Train Describer (see Section 5.7.1) and TRUST datasets. This allowed us to infer the data $(r_j^k, a_j^k, d_j^k)_{j=1}^{J^k}$ for each train k, as described in Section 5.4.7.

Because the instances are drawn from real examples, the level of perturbation varies considerably between them. However, the set of instances as a whole is an unbiased sample of the daily and weekly fluctuations in traffic, and the variations in primary delay over that month.

Figure 5.10 shows some relevant characteristics of the test set. Figure 5.10(a) shows the distribution of the number of trains over the test set. The majority of instances have between 20 and 32 trains, with the most common value being 29. The 4 instances with 3 trains or fewer are taken from data corresponding to public holidays. Figure 5.10(b) shows the distribution of the number of *conflicts* in each instance

over the testset. This is calculated as the number of capacity constraints which are violated by the solution obtained by solving the problem without track capacity constraints. Finally, Figure 5.10(c) concerns the number of *train pair conflicts*. This is defined as the number of distinct pairs of trains between which there is at least one conflict. The number of conflicts is higher than the number of train pair conflicts, because one conflicting pair of trains usually causes several conflicts. It is common for these conflicts to occur on a single route over several consecutive time intervals.



(a) (b) (c)

Figure 5.10: Histograms showing the test set distribution of the number of (a) trains, (b) conflicts and (c) train pair conflicts.

5.8 Computational Results

A computational study has been carried out to evaluate the performance of our branch-and-price algorithm on our new instances. In Section 5.8.1 we investigate the run time required to solve the instances to optimality. The effects of the acceleration strategies, the relationship between conflicts and algorithm performance, and the number of reroutings in the solutions are also explored. In Section 5.8.2, we present the results of imposing a 20 second time limit. These results evaluate the suitability of the developed algorithm for a real-time environment.

For the purpose of the analysis, we have omitted the results of the 32 instances in which there were no conflicts. By Proposition 2, the LP relaxation of an instance

with no conflicts cannot have a fractional solution. As a result, all such instances were solved to optimality in less than 3 seconds, without the need for branching. Many of these instances do include small traffic perturbations, but they are not severe enough to cause conflicts.

The algorithm is implemented using SCIP 6.0.2 [Gleixner et al., 2018] as a branch-and-price framework with custom plugins written in the C language using Gurobi 9.0 [Gurobi Optimization LLC, 2020] as the linear programming solver. The experiments were carried out on a computing node equipped with an 18 core Intel Xeon E5-2699 v3 CPU with 2.30GHz and 500GB of RAM, running Ubuntu 16.04.

The full tables of results are available online (see [Reynolds, 2020]).

5.8.1 Solving to Optimality

First, we solved each instance with a time limit of 600 seconds to evaluate the time required to solve the instances to optimality. Figure 5.11 shows that across all 278 instances with at least one conflict, the median time to solve to optimality using no acceleration strategies was 12.55 seconds. It also shows that 90% of instances were solved within 45.45 seconds, and 4 instances were not solved to optimality in 600 seconds. Using acceleration strategies, we were able to reduce the median time to 10.84 seconds, the 90^{th} percentile to 31.55 seconds and solve all but 2 instances. These results demonstrate that whilst most instances can be solved in very reasonable times, it is not realistic to solve every instance to optimality in a real-time setting.

Considering the instances that were solved using acceleration methods, we find that the number of branch-and-bound nodes is highly correlated with the time to solve to optimality, with a Pearson correlation coefficient of 0.87, significant with p-value < 0.001. A mean of 88 variables per instance were generated throughout the column

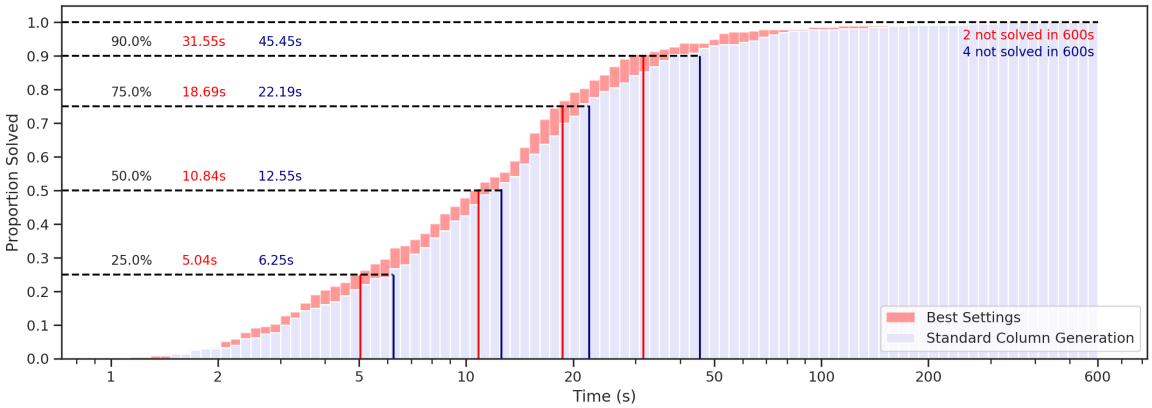


Figure 5.11: A cumulative histogram showing the proportion of instances solved for each number of seconds. Note that a log scale is used on the x-axis.

generation algorithm, demonstrating fast convergence. Although all of the instances had 64,082 constraints, solving LPs took only a small proportion of the total solve time: on average 74% of the total solving time was spent pricing variables.

5.8.1.1 Acceleration Methods

The effects of both acceleration methods were examined separately. The performance profile in Figure 5.12(a) shows that partial pricing was effective at reducing solution times, particularly by producing modest improvements across many of the instances. Partial pricing speeds up the convergence of column generation, so it has the potential to be effective on all instances. However, column generation convergence was less important than the number of branch-and-bound nodes in determining overall solution time, so its effect is limited.

Figure 5.12(b) shows that reduced cost variable fixing was also effective at reducing solution times. The performance profile reveals that it produced gains in a smaller number of instances. This might be due to the fact that it works by reducing the number of branch-and-bound nodes required to reach optimality, and therefore there was limited scope to be effective on instances with relatively few nodes. Figure

5.12(c) shows that partial pricing and variable fixing were most effective when used together. The remainder of the results analysis refers to results obtained by using both of the acceleration strategies.

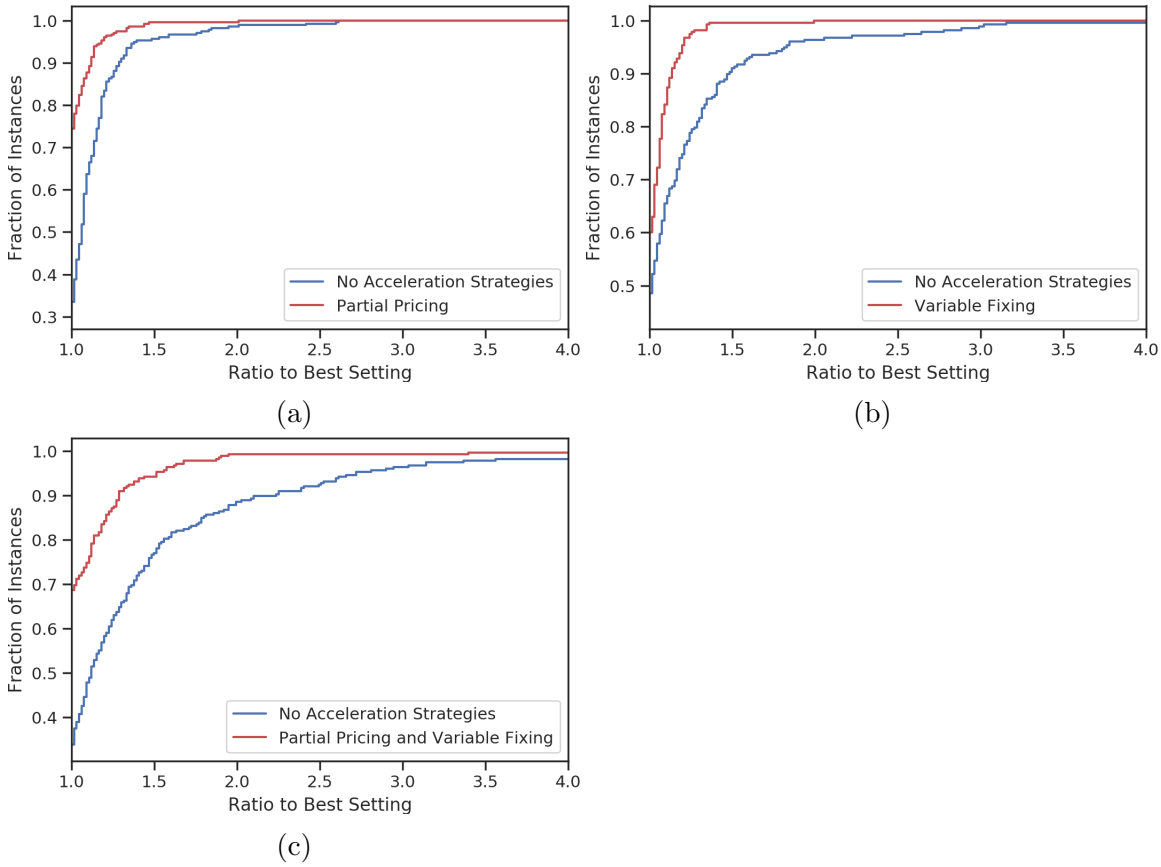
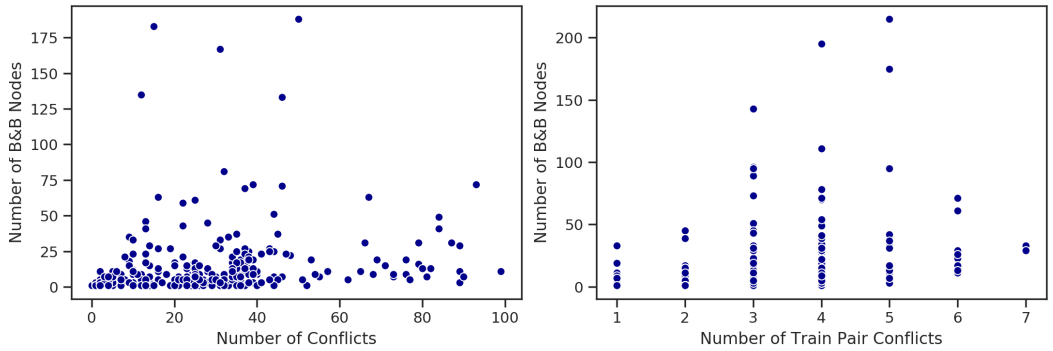


Figure 5.12: Performance profiles comparing standard column generation without any modifications with (a) partial pricing (b) reduced cost variable fixing and (c) both.

5.8.1.2 The Effect of Conflicts

In Section 5.6.5, we conjectured that there is a positive relationship between the number of branch-and-bound nodes and the level of conflict in an instance. The relationship between the number of conflicts and the number of branch-and-bound nodes is plotted in Figure 5.13(a). This plot is repeated with the number of train pair conflicts in Figure 5.13(b).



(a) (b)

Figure 5.13: Scatter plots showing the relationship between conflicts and the number of branch-and-bound nodes. Note that 3 instances with more than 250 branch-and-bound nodes have been omitted from both plots for ease of understanding.

The number of branch-and-bound nodes has a moderate positive relationship with both the number of conflicts, and the number of train pair conflicts. Because the quantities are discrete, we measure this relationship with the Spearman correlation coefficient which looks at the correlation between the ranks of the values. The Spearman correlation coefficients are 0.38 and 0.68 respectively, both with p-value < 0.001. One possible explanation for the higher correlation with the number of train pair conflicts is that branching often resolves multiple conflicts at the same time. This is especially likely to happen when conflicts occur for the same route in consecutive time intervals.

Despite this apparent relationship, a significant amount of the variation in the number of branch-and-bound nodes is unexplained by the number of conflicts. It is likely that this is accounted for by the variation in difficulty of resolving each conflict. Understanding the relationship between the difficulty of resolving conflicts and the instance difficulty would be an interesting topic for further research.

5.8.1.3 Rerouting

As part of the validation of the model, we investigated how many times a train was rerouted in each instance. Figure 5.14 shows that at least one rerouting was carried out in 48% of instances, with the most common number of reroutings being 1. Greater numbers of reroutings were less frequent, with a maximum of 5 carried out in a single instance.

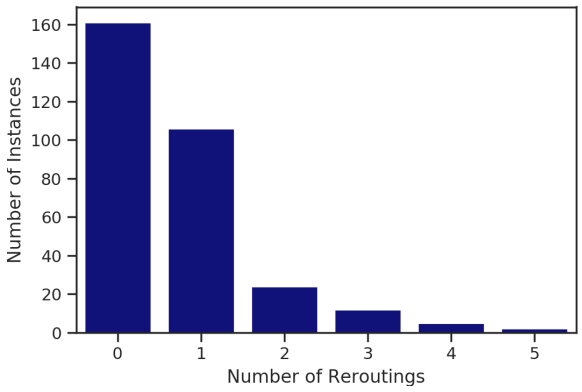


Figure 5.14: A bar plot of the number of reroutings in each solution.

The presence of rerouting decisions in the solutions justifies the inclusion of rerouting in the model. A model without the possibility of rerouting would not be able to produce these solutions, which are optimal under our objective function. In these instances, the schedule event weights β_j^k for platform alternatives were discounted by a factor of 0.9. The number of reroutings would be reduced if this parameter in the objective function were smaller.

5.8.2 Solving with a Strict Time Limit

Each instance was solved with a time limit of 20 seconds, in order to assess the performance of the algorithm when the available time is severely limited as it is in the real-time setting. We found that 81.6% of instances were solved to optimality

within this time limit. Of those that were not solved to optimality, the median gap between the best integer solution and the best bound on the optimal integer solution was 0.17% and the mean was 0.64%. A histogram of the obtained integrality gaps is shown in Figure 5.15.

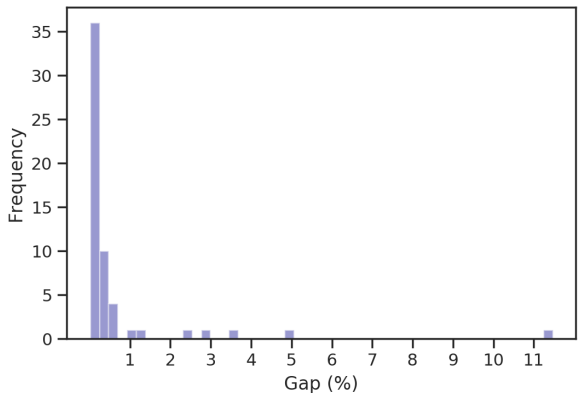


Figure 5.15: A histogram of the percentage gap between the best integer feasible solution found, and the best bound on the optimal solution.

These results demonstrate that the developed algorithm could be effectively used in a real-time setting. Even within the very stringent time limit of 20 seconds, most of the instances can be solved to optimality. Solutions within a few percentage points of optimality can be found for the remainder. The fact that solution quality can be guaranteed even when the optimal solution is not found is a significant advantage of this method over heuristic methods that cannot do the same. This is because knowledge of the solution quality would be an important factor in any decision over whether or not to accept a solution and implement it in practice.

5.9 Conclusion

In this paper, we present a new model for the Train Timetable Rescheduling Problem that is capable of performing rerouting and retiming simultaneously. The model

employs the concepts of occupying and banning to model sectional-release interlocking whilst using routes as the track components. This is a key contribution, since it shows how track capacity can be modelled in an accurate yet tractable fashion. We also present a new objective function that was developed in collaboration with Network Rail. This objective function uses the concept of utility to tractably and flexibly represent Network Rail's complex rescheduling preferences.

Our experiments, which were conducted on a new set of test instances based on real data, show that the majority of instances can be solved to optimality in 20 seconds. The remainder can be solved to provably near to optimality in this time limit, with very good guarantees on solution quality. We show that using partial pricing and reduced cost variable fixing are effective in accelerating our tailored branch-and-price algorithm. At a broader level, these results demonstrate that it is realistic to use exact time-indexed methods in a real-time environment. Finally, our results confirm the conjecture raised by our theoretical result, that instances with a greater level of conflict between train movements are more difficult to solve.

Future research will focus on simultaneously calculating train speed profiles. The assumption that all trains are capable of achieving the same traversal times may not be realistic in railways with mixed traffic. Another potential direction for further work is to refine the objective function. Economic methods could be used to estimate the utility function of Network Rail, for example by testing their preferences between solutions produced with different parameter values. The empirical relationship between these values and the resulting optimal rescheduling actions is not well understood. Finally, a big challenge for the future is to extend this model to whole regional networks of track, either by coordinating many smaller models or by improving the solution techniques so that very large instances can be solved.

Chapter 6

A data-driven, variable-speed model for the train timetable rescheduling problem

6.1 Introduction

Reactionary delays are a significant problem in railway systems. These are delays that are caused by the knock-on effect of prior delays. In 2019, reactionary delays were responsible for 64.35% of total train delay minutes in Great Britain (statistic provided by Network Rail). The problem is particularly acute in large, busy station areas, where the limited capacity of the station creates a bottleneck.

Reactionary delays can be reduced by performing *timetable rescheduling*. Timetable rescheduling involves changing the planned schedules of trains in real-time to respond to unexpected delays. The aim of solving the Train Timetable Rescheduling Problem (TTRP) [Cacchiani et al., 2014] is to find a way to reschedule the timetable that is achievable in practice and optimises some objective. TTRP models must take into

account the *speed profile* of each train, which describes how the velocity of the train changes over time. If the speed profiles of trains are not modelled with sufficient accuracy, then TTRP solutions may perform worse than expected in practice. This has been by demonstrated by Hosteins et al. [2019] using a detailed railway simulator. Lack of attention to speed profile modelling is therefore a significant risk to the validity of TTRP models.

The focus of this paper is the development of techniques for improving the modelling of speed profiles in TTRP models. In particular, the model proposed by Reynolds et al. [2020] is extended to include approximate train speed trajectories. Using a new set of realistic instances from Derby station in the UK, we show that this extended model can be solved to optimality in times comparable to the original model of Reynolds et al. [2020].

6.1.1 Problem Description

The Train Timetable Rescheduling Problem (TTRP) is solved following a disturbance to the timetable to calculate an optimal rescheduled timetable. This disturbance could consist of any set of delays that results in two trains requiring the same infrastructure at the same time (a *conflict*), making the current timetable infeasible. The rescheduled timetable — the solution to the problem — consists of a new route and set of timings (i.e. a new schedule) for each controlled train. Controlled trains are those that are forecast to be inside a defined area of track during a time horizon. New routes can involve stopping at different platforms from those originally planned, taking different approaches to planned platform stops, or cancelling stops altogether. The rescheduled timetable must contain no conflicts and therefore be capable of being carried out in practice, respecting the constrains of the signalling system. A solution is considered optimal if it maximises a utility function representing the preferences

of Network Rail, the infrastructure manager in Great Britain. This is modelled as the total weighted utility over all train stops which are carried out, where at each stop the utility disfavours lateness, platform change and cancellation.

6.1.2 Structure of the Paper

Key concepts and relevant literature are reviewed in Section 6.2. Section 6.3 provides an overview of our proposed model and approach to modelling speed profiles. Methods for estimating traversal times are developed in Section 6.4. The model is then described in full in Section 6.5. Section 6.6 presents a computational study. Section 6.7 contains our conclusions and suggestions for future research.

6.2 Literature Review

Many different variants of the TTRP have been described in the literature, and many different models have been proposed. Much of this research is detailed by the surveys of Cacchiani et al. [2014], Fang et al. [2015] and Corman and Meng [2015]. The *speed profile* of a train k is a continuous, non-linear function $v^k : [0, T] \to \mathbb{R}^+$ mapping each time t in the time horizon to the velocity $v^k(t)$ of the train at that instant. However, it is computationally impractical to optimise such a function for each train within a TTRP model. Our literature review will focus specifically on the ways in which train speed profiles have been approximated in TTRP models.

6.2.1 Fixed-speed and Variable-speed Models

Timetable rescheduling models can be classified as either *fixed-speed* or *variable-speed* models, depending on how speed profiles are modelled. An early reference to this terminology appears in [Cordeau et al., 1998, p. 393-396]. In fixed-speed models, speed profiles are implicitly modelled via the specification of a fixed minimum time that is required for each train to traverse each segment of railway track. These are called *minimum traversal times* because whilst trains are permitted to stop in any segment and hence spend longer than this minimum time, they cannot traverse the segment in a shorter amount of time. In fixed-speed models, minimum traversal times are pre-computed and apply regardless of the rescheduling actions that are proposed by the model or the speed profiles required in practice to achieve them. The fixed minimum traversal time of a segment may be the same for every train that traverses it. Alternatively, minimum traversal times may be calculated based on assumptions about the likely speed profile of a particular train carrying out its originally planned schedule. There are many examples of fixed-speed TTRP models, such as those presented by Corman et al. [2010b], Meng and Zhou [2014], Pellegrini et al. [2014], Lamorgese et al. [2016] and Reynolds et al. [2020].

Fixed-speed models can sometimes produce solutions that are not achievable in practice. The reason for this is illustrated in Figure 6.1. Consider a train traversing three segments of track r_0, r_1 and r_2 in sequence. If the train maintains a constant velocity, then the traversal times of the segments are t_{r_0}, t_{r_1} and t_{r_2}, respectively. Now suppose that the train comes to an unplanned stop in r_2 as a result of a rescheduling decision. A fixed-speed model will use the same fixed traversal times for each segment, with an additional waiting time of D in r_2. However, in reality the train's speed profile must change so that it decelerates in order to stop, and accelerates afterwards. This changes the real traversal times to $t^{var}_{r_0} > t_{r_0}$, $t^{var}_{r_1} > t_{r_1}$ and $t^{var}_{r_2} > t_{r_2}$. As a result,

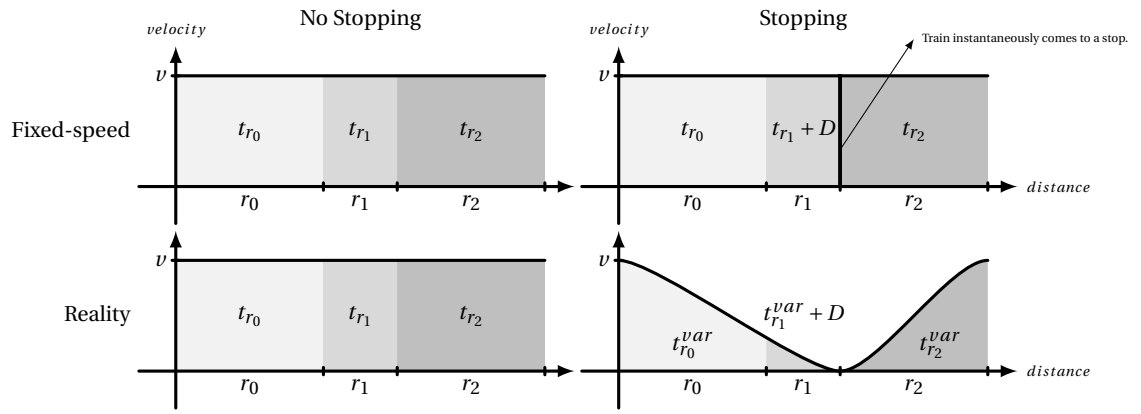


Figure 6.1: A plot showing how the speed profile assumptions made in fixed-speed models compare to actual speed profiles. The plots depict a scenario in which a train traverses r_0, r_1 and r_2 in sequence. On the left, the train does not stop; on the right, the train stops in r_2.

the disparity between actual traversal times and the traversal times in a fixed-speed model can be large. The negative effects of these disparities have been observed by Hosteins et al. [2019]. They find that high-quality solutions produced by the fixed-speed optimisation model of Pellegrini et al. [2014] do not always perform well when tested using a microscopic railway simulation. They find that the extent of the issue is dependent on the granularity of the model and the objective function used.

Variable-speed models attempt to overcome this problem by making traversal times dependent on rescheduling actions. One way of achieving this is by iterating between a fixed-speed rescheduling model and a speed profile optimisation model. Speed profile optimisation models are detailed kinetic models that optimise the speed profile of a single train carrying out a fixed schedule to minimise journey times or energy consumption. The relevant literature is summarised by Yang et al. [2016], Scheepmaker et al. [2017] and Yin et al. [2017]. The traversal times in the fixed-speed model are updated in each iteration according to the speed profiles required to achieve the rescheduling solution that it produced in the previous iteration. This iterative approach has been explored by both D'Ariano et al. [2007b] and Mazzarello and Ottaviani [2007]. Whilst iterative approaches laudably avoid the need for fundamental

changes to rescheduling models, they do not guarantee convergence to a solution that is feasible for both models. Moreover, it can be time consuming to solve a fixed-speed model multiple times, which makes the iterative approach especially unsuitable for the real-time environment in which rescheduling takes place.

To overcome these problems with the iterative approach, speed profiles can be modelled directly within the rescheduling model. Several authors have suggested simple ways to account for the disparity that arises between fixed traversal times and actual traversal times following unplanned stops (as described in the example above). Hosteins et al. [2019] suggest adding a fixed additional time to the traversal time of any segment in which a train comes to a stop. Rodriguez [2007] suggests adding an additional time to the traversal time of the subsequent segment that depends linearly on the amount of time the train stops for. Whilst these methods have the merit of being simple and leading to linear constraints, they are clearly very approximate. In reality, the additional acceleration time is not a fixed constant, and it is not linear in the amount of stopping time. Lusby et al. [2013] show that speed profiles can be modelled within the subproblem of a model solved using column generation. This allows speed profiles to be modelled in a more sophisticated way using kinematic formulae. A similar column generation approach to that of Lusby et al. [2013] is used in this paper. However, speed profiles are modelled differently to overcome some specific problems that are discussed in Section 6.4.

A different approach is to select the speed profile of each train in advance and then disallow rescheduling actions that would compromise their validity. This approach, taken by both Corman et al. [2009a] and Caimi et al. [2012], is suitable if a *green wave* policy is in operation. A green wave policy dictates that trains must only come to a stop within stations, thereby reducing unnecessary braking and acceleration and saving energy. However, it does not solve the problem of modelling speed profiles within the general TTRP, where trains may come to a stop anywhere on the track

network.

More recently, different ways of integrating the TTRP with speed profile optimisation have been proposed by Xu et al. [2017], Zhou et al. [2017] and Luan et al. [2018a,b]. These models seek to determine the actual speed profile to be used by each train simultaneously with carrying out timetable rescheduling. The principal benefit of this integration is that by solving the two problems simultaneously, solutions with better overall quality can be achieved. However, to integrate the problems, it is necessary to assume that precise real-time information about both train speed profiles and signalling states are centrally available, and that both can be centrally and automatically controlled. In other words, these models are only useful for railways in which the functions of *traffic control* and *train operation* are integrated (see [Yin et al., 2017, p. 568] for a discussion of this integration). They are typically only integrated on new and expensive high-speed lines. As a result, the integration of the TTRP with speed profile optimisation is inappropriate for practical use on the majority of railways.

6.2.2 Physics-based and Data-driven Models

Each of the models proposed by Lusby et al. [2013]; Xu et al. [2017]; Zhou et al. [2017]; Luan et al. [2018a,b] are *physics-based*. This means that traversal times of track segments are derived from kinetic speed profile modelling. Track segment distances and train speed capabilities are combined with assumptions about the tractive force applied by the driver to calculate traversal times. This kind of kinetic modelling has traditionally been used to estimate running times for timetable construction and evaluation — an introduction to this topic is provided by Brünger and Dahlhaus [2014]. Luan et al. [2018a,b] incorporate much of this kinetic modelling into a mixed-integer non-linear programming formulation, and propose and two heuristics for solving a

linearised version of the formulation. Heuristics are used because realistic instances cannot be solved to optimality in suitable computation times. Both Xu et al. [2017] and Zhou et al. [2017] discretise velocity, thus avoiding direct representation of the non-linear kinetics of train motion. Xu et al. [2017] extend the Alternative Graph model (see D'Ariano et al. [2007a]) to incorporate speed-dependent traversal times. Zhou et al. [2017] propose a time-space-speed network model for the offline train timetabling problem on a high-speed line with power supply constraints.

The physics-based approach to traversal time modelling has practical disadvantages. Detailed information about the physical properties of the track and trains is required, despite the fact that it can be hard to obtain. For example, Luan et al. [2018b] use physical parameters for the resistance between train and track that are individualised for each train and block section. Finding this information isn't simply a data collection exercise: many parameters need to first be estimated and then calibrated within the overall model. Although sophisticated methods such as those of Bešinović et al. [2013] have been developed to estimate the parameters used in physical speed profile calculations, each estimated parameter is still subject to uncertainty. The effect of this uncertainty on rescheduling models is not well understood. Our model bypasses the need for these physical parameters, and is therefore significantly less onerous to test and deploy. It is also easier to adapt to changes in infrastructure than physics-based models, since any changes will be reflected in the data and can be used to update traversal times. These are significant advantages given that practical implementations of the TTRP are still rare (see Lamorgese et al. [2018] for a description of the state of implementation).

There are also important modelling disadvantages to using the physics-based approach. In a physics-based model, the modeller must make assumptions about which speed profiles should be feasible. For example, Lusby et al. [2013] and Zhou et al. [2017] assume constant rates of acceleration over each section of track and each time

interval, respectively. Lusby et al. [2013] allow trains to travel at any real-valued speed below the speed limit, whilst Zhou et al. [2017] and Xu et al. [2017] allow trains to travel only at one of a few pre-defined speeds in each block section. Zhou et al. [2017] allow trains to transition between any two speed levels provided limits on the acceleration and deceleration capabilities of the train are respected, whilst Xu et al. [2017] allow trains to transition only between adjacent speed levels. All of these approaches to constraining speed profiles run the simultaneous risks of both eliminating perfectly reasonable speed profiles from the feasible space, and including many speed profiles that are very unlikely to arise in practice.

In this paper, we avoid these problems by taking a *data-driven* approach. This terminology refers to the fact that we infer traversal times from historical observations, leading to an innovative synthesis of statistical techniques with optimisation. Our data-driven approach provides a natural way to model traversal times that reflects speed profiles that have actually arisen in the past on the parts of track that are modelled. Rather than making assumptions about how to constrain speed profiles, our approach allows the data to speak for itself. Results for our application show that the use of more than one traversal time is justified by the data on only a subset of the routes. This highlights an additional advantage of our data-driven approach. It is able to target increased model complexity (a higher number of possible traversal times) at parts of the track where it can be best justified by the data. The result is a significantly more parsimonious model than any of the physics-based models that have been mentioned.

6.2.3 Contributions

The contributions made by this paper can be summarised as follows:

1. We propose a new variable-speed model for the TTRP in complex station areas.

The model utilises a time-space-type graph to approximate train speed profiles.

2. We show how the application of statistical methods to historical data can be used to model traversal times in a variable-speed model. This data-driven approach results in a parsimonious model that is less onerous to test and deploy than existing physics-based models. Moreover, it avoids the need to make restrictive assumptions about speed profiles.

3. We present a new set of instances based on real data from Derby station in the UK.

4. We show that these instances can be solved to optimality or provably near to optimality in times suitably short for real-time operations. In particular, the solving times are comparable with the fixed-speed model proposed by Reynolds et al. [2020].

6.3 Model Overview

This paper presents a model for the TTRP that is based on a time-space-type (TST) graph. This is a directed graph $G = (N_0 \cup \{source, sink\}, A)$ that models the state of each train over the time horizon using a path from the *source* node to the *sink* node. Each node in N_0 represents a state in which a train could be, whilst each arc represents a possible transition between these states.

Each node in N_0 corresponds to a combination (r, t, v) of a route r, a time interval t and a speed profile type v. These are defined as follows:

- A *route* is a short length of track that runs from one railway signal to another. The controlled area of track is formed of a set of routes N_r, that can be modelled as a route graph $G_r = (N_r, A_r)$, where the arcs in A_r represent feasible route

transitions. Figure 6.2 shows the route graph for an area centred on Derby station.

- *Time intervals* are periods of time, each of length 10 seconds, that together form a partition $\mathcal{T}$ of the time horizon. The time horizon begins at the time the model is solved and lasts for one hour.

- Each traversal of a route $r \in N_r$ by a train is modelled as having a *speed profile type*. The minimum number of time intervals L_r^v required to traverse r is dependent on the speed profile type v that is used. Whilst there are three general types $N_v = \{v_0, v_1, v_2\}$, the possible speed profile types for any given route r are $N_r^v \subseteq N_v$. The definition of these speed profile types and the calculation of the traversal times is the subject of Section 6.4.

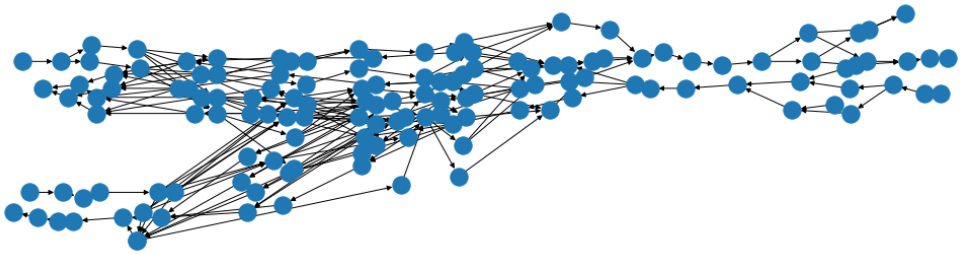


Figure 6.2: The route graph G_r for the area centred on Derby station that is used in our computational experiments.

6.3.1 Example TST Graph

A small explanatory example of a TST graph is depicted in Figure 6.3. In this example, the route graph is given by

$$N_r = \{AB, BC, CD, DE\} \text{ and } A_r = \{(AB, BC), (BC, CD), (CD, DE)\}$$

and the time horizon $\mathcal{T} = \{0, \ldots, 7\}$ consists of eight time intervals. There are three speed profile types, v_0, v_1 and v_2, although type v_2 is not possible in routes CD or DE ($N_v^{CD} = \{v_0, v_1\} = N_v^{DE}$). Type v_0 corresponds to stopping, since the arcs join nodes corresponding to the same route at consecutive time intervals. The traversal times of type v_1 are $L_{AB}^1 = 3$, $L_{BC}^1 = 2$ and $L_{CD}^1 = 2$. The traversal times of type v_2, $L_{AB}^2 = 1$ and $L_{BC}^2 = 1$, are shorter. The graph G contains arcs allowing movement between speed profile types, but movement between types v_0 and v_2 is not possible. The thick blue line is an example *source-sink* train path in G. It corresponds to a sequential traversal of all routes, with AB and CD traversed with speed profile type v_1, BC with type 2, and the train coming to a stop in DE.

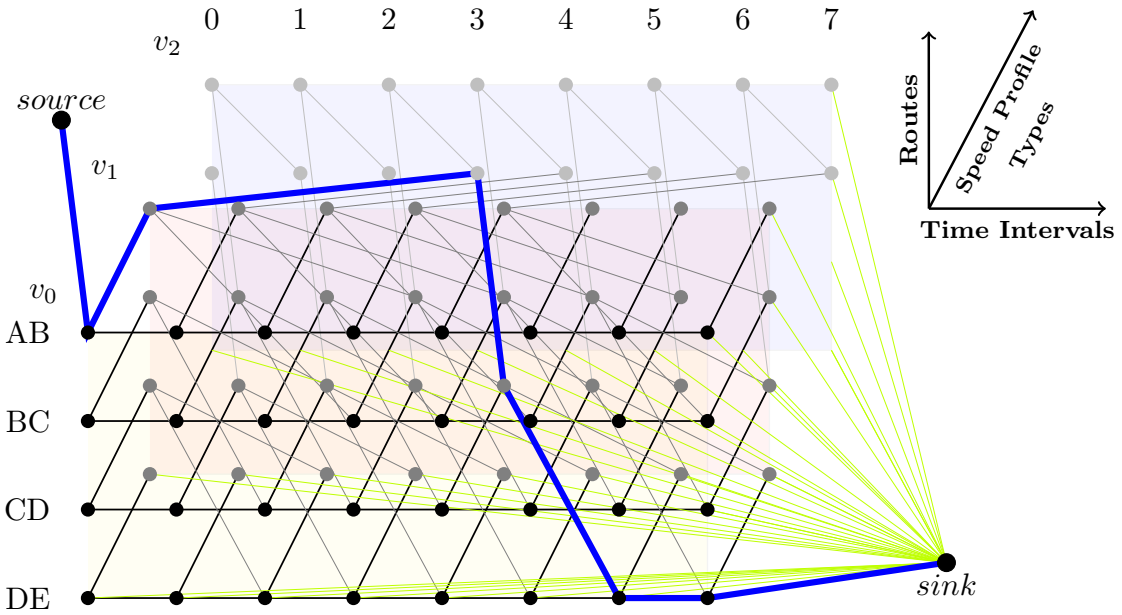


Figure 6.3: An example of a TST graph.

A solution to the problem (i.e. the new schedule) comprises one *source-sink* path in G for each controlled train $k \in \mathcal{K}$. This path completely describes the sequence of routes to be traversed by k, the time intervals in which each route traversal begins and the speed profile type of each traversal.

6.3.2 Speed Profile Types

In fixed-speed models, the traversal time of a route $r \in N_r$ is represented by a single value L_r. Trains are usually permitted to come to a stop on any route. As a result, L_r is merely the minimum traversal time because a train may spend longer than L_r in route r by stopping in it. Whilst L_r may depend on the train, route or pre-planned speed profile, it does not depend on the speed profile required to achieve the rescheduled solution. We refer to stopping (denoted by v_0) and ordinary traversal (denoted by v) as *speed profile types*. Figure 6.4 visualises the fact that both stopping and ordinary traversal are possible on any route, regardless of which type occurred on the previous route.

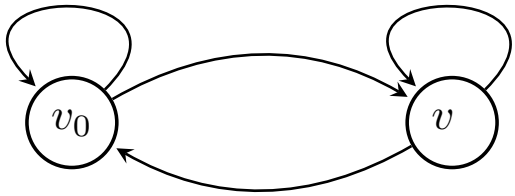


Figure 6.4: Speed profile type graph for a fixed-speed model.

Our approach is a natural extension of this idea to use two different traversal times L_r^1 and L_r^2 (with $L_r^1 > L_r^2$) corresponding to two speed profile types v_1 and v_2, respectively. The traversal time that is required for a train to traverse r will be either L_r^1 or L_r^2, depending on the speed profile required to perform the rescheduled solution. The resulting model is, therefore, a variable-speed model. In particular, the speed profile type used by a train on a given route is constrained to be adjacent in the type graph G_v (see Figure 6.5) to the type used on the preceding route. Because $(v_0, v_2), (v_2, v_0) \notin A_v$, this constraint prevents trains from using the faster speed profile type v_2 and stopping v_0 on consecutive routes. The rationale is that fast speeds and stopping must be separated by periods of acceleration or deceleration that involve slower speeds. Note that this assumption is somewhat similar to the

constraint of Xu et al. [2017] stating that the speed levels on consecutive block sections must be at most one level apart. However, we use speed profile types instead of physically defined speed levels (e.g. 250–300 km/h).

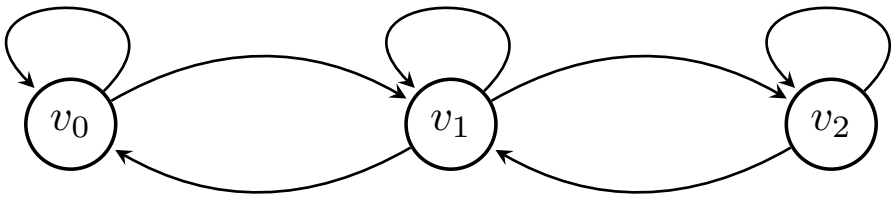


Figure 6.5: Speed profile type graph $G_v = (N_v, A_v)$ for our approach, where $N_v = \{v_0, v_1, v_2\}$ and $A_v = (N_v \times N_v) \setminus \{(v_0, v_2), (v_2, v_0)\}$.

6.4 Traversal Time Estimation

It is important to address the question of how to calculate the variable traversal times L_r^1 and L_r^2 for each route r. This is because the extent to which these values are representative of the actual traversal times of trains with different speed profiles is a major determinant of whether the solutions produced by the model are achievable in practice. The calculation of traversal times is therefore an inextricable part of the modelling methodology.

In fixed-speed models, traversal times can be estimated using classical running time estimation techniques — see [Brünger and Dahlhaus, 2014] for a review. Stochastic running time estimation methods have also been proposed for use in timetable simulation [Yuan and Medeossi, 2014]. However, the question of how to calculate traversal times for variable-speed TTRP models hasn't been adequately addressed. This task poses several unique challenges that straddle the topics of running time estimation and timetable rescheduling. In our case, two times L_r^1 and L_r^2 that represent trains with different speed profiles are required. These must be meaningful in

the sense that it should be rare or impossible for a train to use types v_0 and v_2 in consecutive routes, but possible for any other combination to occur. An additional challenge is posed by the discrete nature of time in the model, meaning that times must correspond to a whole number of time intervals. Finally, routes in station areas are often much shorter than the distances over which running times are typically calculated.

Our approach involves estimating traversal times based on historical data. The data that is used was collected by a *Train Describer* over a seven month period. A Train Describer is a real-time information system that records the sequence of routes traversed by each train and the number of seconds spent in each one.

6.4.1 Method 1: Estimating a Single Time (Fixed-Speed)

First we summarise the method employed by Reynolds et al. [2020] for estimating a single traversal time L_r for a given route $r \in N_r$ in a fixed-speed model. The data used is $\mathbf{y}^r = (y_1^r, \dots y_{n_r}^r)$, a vector containing the number of seconds spent in r by n_r different trains. Unimodal statistical distributions are generally not appropriate for modelling $\mathbf{y}^r$. This is because the times arise from different processes according to the speed profile used by a train in r. By modelling each speed profile type as a different process, we can model $\mathbf{y}^r$ as a mixture of unimodal distributions. The speed profile type of each observation is unobserved, but this can be estimated by fitting a mixture model. Specifically, we fit a Gaussian Mixture Model (GMM) to each $\mathbf{y}^r$ using the Expectation-Maximisation (EM) algorithm. This is a model-based clustering technique that identifies groups of historical times (called *clusters*) that approximately follow a Gaussian distribution. An introduction to GMMs and the EM-algorithm is provided by Bouveyron et al. [2019].

To fit a GMM to $\mathbf{y}^r$, the number of clusters (each corresponding to a speed profile

type) must be specified in advance. The appropriate number is different for each route. For example, on the open line where trains rarely travel below the line speed there is often only one process occurring and therefore a single cluster is appropriate. Conversely, up to three different speed profile types are discernable on many routes within stations. To decide the number of clusters, we fit three models with 1, 2 and 3 clusters respectively, and choose the model that optimises the Bayesian Information Criterion [Bouveyron et al., 2019, p. 51].

In the fixed-speed model, a single time L_r must be chosen for each route from the fitted mixture distribution. The cluster with the smallest mean is selected, since this reflects the times of trains travelling close to the speed limit. From this component, the mean is rounded up to the nearest number of whole time intervals to produce L_r. Whilst rounding introduces approximation error, rounding up ensures that the feasibility of a solution to the model is not compromised. An example of the result of this clustering process is shown in Figure 6.6.

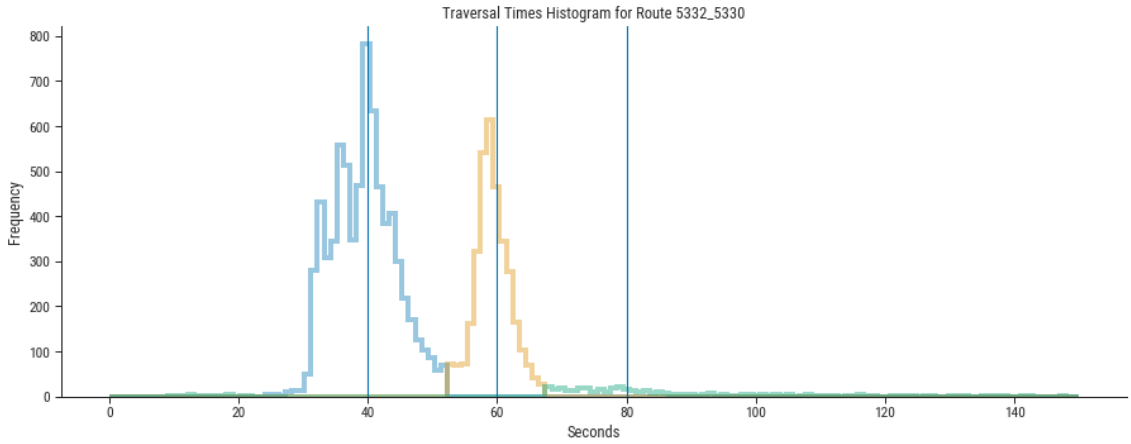


Figure 6.6: The histogram of historical traversal times for route 5332_5330. The three clusters identified are shown in different colours. The rounded mean of each cluster is shown as a vertical line. The left-most vertical line (40 seconds, or L_r=4) is used as the final estimate.

6.4.2 Method 2: Extension to Multiple Times (Variable-Speed)

The method presented in Section 6.4.1 can be extended to produce two traversal times, L_r^1 and L_r^2, for routes with three clusters. The traversal times L_r^1 and L_r^2 are calculated by taking the smallest and second smallest means of the clusters, respectively, and rounding up to the nearest whole number of time intervals. This allows more speed profile types to be represented in the model. Trains travelling close to the speed limit have a time of L_r^1, whilst trains that accelerate/decelerate or coast below the speed limit have a time of L_r^2. Observations in the third cluster are discarded, since these correspond to stopping, and that is not modelled using a traversal time.

Given the assumptions made about how clusters correspond to speed profile types, we expect to observe that when trains traverse two routes consecutively, the respective traversals are rarely of types $v_0 \to v_2$ or $v_2 \to v_0$. This is because these transitions represent an abrupt change between stopping and a fast speed within the space of a single route. To test whether this is the case, a model of transitions between different clusters is created using historical data from the Train Describer. For each historical train journey j, this transition data consists of both the sequence $(r_1^j, \dots, r_{n_j}^j) \in (N_r)^{n_j}$ of routes traversed, and the corresponding sequence $(t_1^j, \dots, t_{n_j}^j) \in \mathbb{R}_+^{n_j}$ of traversal times for each route in seconds. Since the clustering process classifies each traversal time t_i^j as belonging to a particular speed profile type, the journey can also be represented as a sequence $(i_1^j, \dots, i_{n_j}^j) \in \{v_0, v_1, v_2\}^{n_j}$ of speed profile types.

In order to test the suitability of the clustering method, we model the speed profile type sequences probabilistically using a discrete Markov chain $(X_t)_{t\in\mathbb{N}}$ with state space $N_v = \{v_0, v_1, v_2\}$. This means that for each time step $t \in \mathbb{N} = \{0, 1, 2, \dots\}$, X_t is a discrete random variable taking values in N_v, such that

- **Markov Property**

  $\mathbb{P}(X_{t+1} = i_{t+1} | X_t = i_t, \ldots, X_0 = i_0) = \mathbb{P}(X_{t+1} = i_{t+1} | X_t = i_t)$ for any $t \geq 1$ and $i_0, \ldots, i_{t+1} \in N_v$.

  In our application, the Markov property means that the conditional probability of a train using a speed profile type, given the types used on all previous route traversals, only depends on the type used on the most recently traversed route.

- **Time homogeneous**

  $\mathbb{P}(X_{t+1} = j | X_t = i) = \mathbb{P}(X_t = j | X_{t-1} = i)$ for any $t \geq 1$ and $i, j \in N_v$.

  This means that the probability of transitioning from state i to j is independent of t, and we denote this transition probability by p_{ij}.

The transition probabilities of this Markov chain can be inferred from the data. If n_{ij} is the number of times j immediately follows i in a speed profile type sequence for a train journey in the data, then p_{ij} is estimated by

$$p_{ij} = \frac{n_{ij}}{\sum_{m \in N_v} n_{im}}. \tag{6.1}$$

The clustering and transition probabilities were calculated using data from Derby station, and the estimated transition probabilities are shown in Figure 6.7. Whilst $p_{v_0 v_2} = 0.01$ is small as expected, $p_{v_2 v_0} = 0.21$ is higher than expected. It would be problematic to disallow transitions from v_2 to v_0 in our optimisation model when the historical data shows that this occurs after as many as 21% of type v_2 traversals.

The higher than expected value of $p_{v_2 v_0}$ could result from inadequacies in the clustering methodology that lead to misclassification of traversal times. One particular concern is that the Gaussian distribution may not be appropriate to model clusters. Another possibility is that distinct and meaningful clusters do not exist for some routes, or do not strongly correspond to speed profile types. A third possibility is

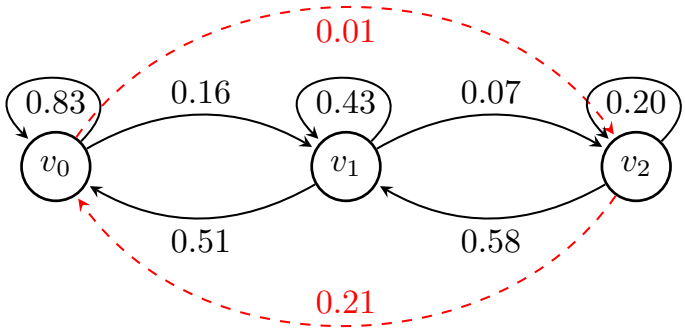


Figure 6.7: The Markov chain $(X_t)_{t\in\mathbb{N}}$, with estimated transition probabilities shown. Probabilities are rounded to two decimal places.

that $p_{v_2v_0}$ is a poor estimate of the true transition probability. This could have arisen because for some routes, the cluster representing type v_2 has only a small number of data points. Finally, it may be inappropriate to assume that the Markov property holds. For example, the distribution of X_t given the values at each previous time step might be dependent on the value of X_{t-2} in a way that our probabilistic model has failed to reflect.

The potential problems that have been identified may affect different route transitions to different extents. It is possible to estimate transition probabilities $p_{ij}^{r,r'}$ for each route transition $(r, r') \in N_r$ separately. The probabilities $p_{ij}^{r,r'}$ are calculated in the same way as p_{ij} using formula (6.1) with one difference. The difference is that n_{ij} is replaced by n_{ij}^r, the number of observations of $(r, t) \to (r', t')$ in the transition data such that t is classified as type i and t' is classified as type j. The distributions of $p_{v_0v_2}^{r,r'}$ and $p_{v_2v_0}^{r,r'}$ over all route transitions $(r, r') \in N_r$ are shown in Figure 6.8. There is considerable variation across different route transitions, with $p_{v_2v_0}$ being acceptably low for some, and unacceptably high for others.

A severe limitation of this approach is that the transition data is used only for validation and not for the clustering, which is performed separately for each route. The transition data potentially contains useful information that could be used when per-

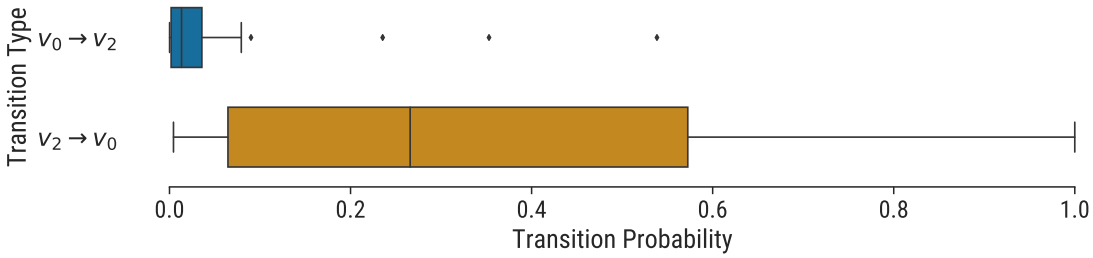


Figure 6.8: Boxplots showing the distribution of transition probabilities by type.

forming the clustering. For example, a transition from a cluster of type v_2 to a cluster of type v_0 might indicate that the second observation is misclassified and should be in cluster v_1. Transition patterns also contain useful information for selecting the number of clusters. This limitation is addressed in the next section, where the full transition data is utilised.

6.4.3 Method 3: Transition-based Multiple Traversal Time Estimation

This method for estimating differentiated traversal times is designed to address the problems identified with Method 2. It uses the full transition data to estimate the traversal times, and it does not rely on the Markov property or assume that the data follows a Gaussian distribution.

There are four main steps involved in Method 3. The first step is to identify, for each route $r \in N_r$, which observations from $\mathbf{y}^r$ arise from a train coming to a complete stop in r (type v_0). This is performed using traversal time data alone, without transition data. The distributions of times recorded for each route are very heterogeneous, which makes using standard parametric distribution fitting challenging. We therefore take an *ad hoc* approach. Any observation that is longer than 120 seconds, or in the top 10% of observations is assumed to arise from a train coming to a stop. These values (120 seconds and 10%) are selected using our familiarity with the specific area

being modelled.

The second step is to classify the remainder of the traversals as either type v_1 or type v_2. This is performed using the transition data alongside our classification of type v_0 transitions from the first step. Specifically, a route traversal is classified as type v_1 if it occurs immediately before or after another route traversal that has been classified as type v_0. Conversely, route traversals that are not adjacent to traversals of type v_0 are classified as type v_2. This completes the classification of each route traversal into a speed profile type.

The third step is to calculate the traversal times L_r^1 and L_r^2 using the classification from step two. For $i = 1, 2$, the median l_r^i (in seconds) of type v_i observations is divided by 10 (the length of a time interval) and rounded up to the nearest whole number to obtain L_r^i. Medians are used because they are not unduly influenced by more extreme observations in each group. The rounding process is necessary as a result of the discretisation of time in the TST graph.

The fourth and final step is to decide, for each route, whether the data supports using two different traversal times or whether a single traversal time is more appropriate. Due to rounding, routes r for which $l_r^1 - l_r^2 < 10$ have $L_r^1 = L_r^2$, so these should have only one traversal time i.e. $N_r^v = \{v_0, v_1\}$. For other routes, we check for statistical evidence that the true values $\check{l}_r^1$ and $\check{l}_r^2$ of the medians of groups v_1 and v_2 are significantly different (l_r^1 and l_r^2 are estimates of $\check{l}_r^1$ and $\check{l}_r^2$, respectively). Mood's test for a difference in medians [Mood, 1950] is used to assess this. A one-tailed test is performed at significance level 95% to test the null hypothesis that $\check{l}_r^2 = \check{l}_r^1$ against the alternative hypothesis that $\check{l}_r^2 > \check{l}_r^1$. When the null hypothesis is rejected, both of the traversal times, L_r^1 and L_r^2, are used. For the remaining routes, only L_r^1 is used because there is a lack of statistical evidence that using a second traversal time is justified. An illustration of the method for a particular route that is given two

traversal times is shown in Figure 6.9. The results of carrying out this method on all routes for Derby station are described and discussed in Section 6.6.2.

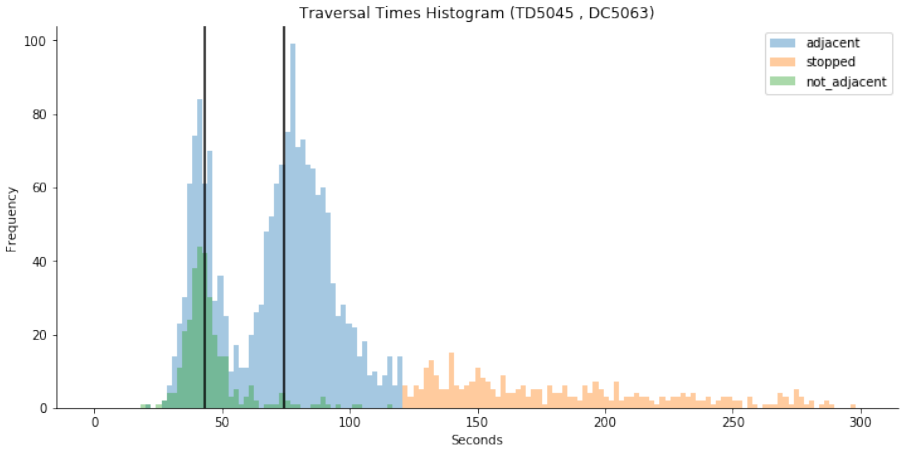


Figure 6.9: A histogram of historical traversal times for a particular route. The three estimated groups are shown in different colours, and estimates for L_r^1 and L_r^2 are marked vertical lines.

6.5 Model Description and Solution Method

The results obtained from traversal time estimation are used to determine both the TST graph, and sets of graph arcs that are used to model track capacity constraints. These objects, in turn, are used to define a Mixed Integer Programming model for the TTRP that is solved using a branch-and-price algorithm.

6.5.1 TST Graph

Recall from Section 6.3 that $G_r = (N_r, A_r)$ is the route graph and that $G_v = (N_v, A_v)$ is the speed profile type graph. The set of discrete time intervals is denoted by $\mathcal{T}$, whilst $\mathcal{K}$ is the set of trains. For each route r, $N_r^v \subseteq N_v$ is the set of possible speed profile types for route r.

The TST graph is given by

$$G = (N_0 \cup \{source, sink\}, A),$$

where, in addition to the artificial source and sink, the nodes of G are given by

$$N_0 = \{(r, t, v) \in N_r \times \mathcal{T} \times N_v : v \in N_r^v\}.$$

The directed arc set $A = \bigcup_{i=1}^{6} A_i$ of G consists of six different arc types. These six types and their interpretations are given below. Figure 6.10 shows an example arc of each different type in a small artificial example of G.

- $A_1 = \{(source, (r_0^k, a_0^k, v_0^k)) : k \in \mathcal{K}\}$

  Entering from the source node to the first known position r_0^k of train k within the time horizon and modelled area, at time interval a_0^k, and speed profile type v_0^k.

- $A_2 = \{((r, t, v_0), (r, t+1, v_0)) : (r, t, v_0) \in N_0 \text{ and } (r, t+1, v_0) \in N_0\}$

  Waiting in route r for one time interval when speed profile type is 0 (i.e. train is stopped).

- $A_3 = \{((r, t, v_0), (r, t, v_1)) : (r, t, v_0) \in N_0 \text{ and } (r, t, v_1) \in N_0\}$

  Transitioning from speed profile type 0 (stopped) to type 1 so that the train can begin traversing r again.

- $A_4 = \{((r, t, v), (r', t + L_r^v, v')) : (r, r') \in A_r, (v, v') \in A_v,$
  $(r, t, v), (r', t + L_r^v, v') \in N_0, v \neq v_0\}$

  Traversing r with seed profile type v, and arriving in a successive route r' with speed profile type v' after a traversal time of L_r^v.

- $A_5 = \{((r, T, v), sink) : r \in N_r, v \in N_v\}$

Exiting to the sink node at the end of the time horizon.

- $A_6 = \{((r,t,v), sink) : \sigma^+(r) = \emptyset \text{ and } (r,t,v) \in N, v \neq v_0\}$

  Exiting to the sink node from a node at the boundary of the area of track modelled. The train cannot exit whilst stationary.

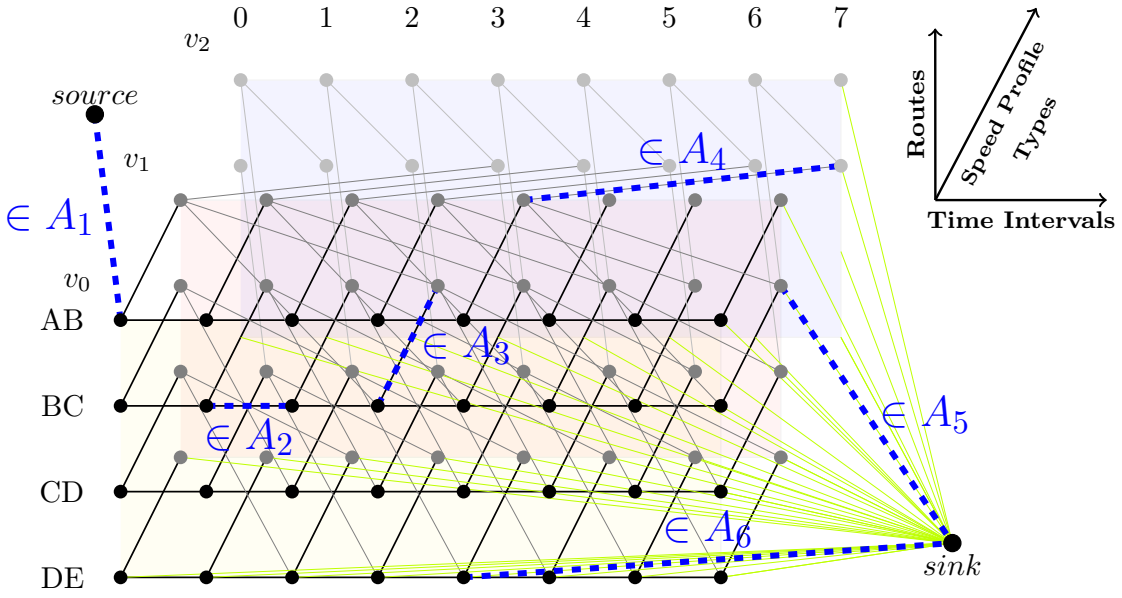


Figure 6.10: An example TST graph, with examples of arcs from different sets A_1–A_6 labelled.

6.5.2 Track Capacity Constraints

A feasible solution to the TTRP is a set of *source-sink* paths in G, one for each train, that collectively satisfy the constraints of the signalling system. The signalling system is a safety system that limits track capacity and regulates train movements to avoid physical collisions. The type of signalling system modelled is a *sectional-release interlocking* system. This takes into account track subdivisions smaller than routes, called *track circuits*. All of the track circuits in a route are *locked* before a train may enter the route, and then *released* individually once the train has vacated each track

circuit. Between locking and release, a track circuit cannot be locked by any other train. Because track circuits can form part of more than one route, the capacities of routes are not always independent.

The constraints induced by the signalling system can be modelled as capacities in the time-space-speed graph. Each pair $(r, t) \in N_r \times \mathcal{T}$ of one route and one time interval is regarded as a *time-space resource*. The capacity of these resources can be consumed by trains in two different ways: *occupying* and *banning*. Resource (r, t) is *occupied* by a train if and only if it has traversed r and at least one of the track circuits in r is still locked during time interval t. By contrast, a train *bans* a time-space resource (r', t) if it makes route r' unavailable during time interval t as a result of occupying (r, t), where r is a distinct route with track circuits in common with r'. These terms are illustrated in more detail by Reynolds et al. [2020].

The capacity of each time-space resource (r, t) is subject to two constraints:

1. (r, t) cannot be occupied more than once; and
2. (r, t) cannot be both banned and occupied.

To enforce these constraints using the capacities of arcs in G, two sets of arcs, $A_{r,t}$ and $\bar{A}_{r,t}$, are defined for each time-space resource (r, t). These are defined such that a train path contains an arc in $A_{r,t}$ if and only if it occupies (r, t), whilst a train path contains an arc in $\bar{A}_{r,t}$ if and only if it bans (r, t). Reynolds et al. [2020] show how these sets can be constructed on a time-space graph without speed profile types. We extend this to show how they can be constructed on the TST graph G. Figure 6.11 visualises an example of a set $A_{r,t}$.

A train occupies (r, t) if and only if the *source-sink* path in G assigned to that train

contains a node from the set

$$W_{r,t} = \left\{(r, t', v') : v' \in N_v^r, t' \in \{t - (L_r^{v'} + h_r) + 1, \ldots, t\}\right\},$$

where h_r is headway time left for the release of route r to occur. To see this, suppose that a train path contains such a node (r, t', v'). Then the track circuits of r cannot not all be released until the train has traversed r, and the headway time h_r has elapsed. This cannot occur before time interval $t' + L_r^{v'} + h_r \geq t - (L_r^{v'} + h_r) + 1 + (L_r^{v'} + h_r) = t + 1$, meaning that at least some track circuits are still locked during time interval t.

A train bans (r, t) if and only if the *source-sink* path in G assigned to that train contains a node from the set

$$\bar{W}_{r,t} = \{(r', t', v') : r' \in S_r, v' \in N_v^r, t' \in \{t - t(r', v', r) + 1, \ldots, t\}\},$$

where S_r is the set of routes distinct from r that share at least one track circuit with r. The quantity $t(r', v', r)$ is the minimum number of time intervals between the track circuits of route r' being locked, and all of the track circuits common to r and r' being released, when r' is traversed using speed profile type v'. This is given by

$$t(r', r) = \lceil \theta(r', r) L_{r'}^{v'} + h_{r'} \rceil,$$

where $\theta(r', r)$ is the proportion of the traversal of r' after which r is released. That quantity is calculated from the track circuit data as described by Reynolds et al. [2020].

Using the definitions of $W_{r,t}$ and $\bar{W}_{r,t}$, $A_{r,t}$ and $\bar{A}_{r,t}$ can be defined as:

$$A_{r,t} = \bigcup_{n \in W_{r,t}} \sigma^-(n) \setminus \bigcup_{n \in W_{r,t}} \sigma^+(n)$$
$$\bar{A}_{r,t} = \bigcup_{n \in \bar{W}_{r,t}} \sigma^-(n) \setminus \bigcup_{n \in \bar{W}_{r,t}} \sigma^+(n),$$

where $\sigma^-(n)$ and $\sigma^+(n)$ are used to denote the set of directed arcs of G entering a node n and leaving a node n, respectively. A path in G contains an arc in $A_{r,t}$ (respectively $\bar{A}_{r,t}$) if and only if it contains a node in $W_{r,t}$ (respectively $\bar{W}_{r,t}$). Therefore a path in G contains an arc in $A_{r,t}$ (respectively $\bar{A}_{r,t}$) if and only if it occupies (respectively bans) time-space resource (r, t).

The track capacity constraints described above can be formulated in the following way. A set of *source-sink* paths in G is a feasible solution to the problem if and only if among all of the paths:

1. At most one arc in $A_{r,t}$ is used; and
2. If an arc in $A_{r,t}$ is used then no arcs in $\bar{A}_{r,t}$ are used.

6.5.3 Antichain Condition

Analogously to the model presented by Reynolds et al. [2020], our formulation of the track capacity constraints is correct if and only if it is not possible for a train to violate these constraints single-handedly. We give conditions under which this is true below.

Definition 5. *An antichain in a directed graph $G = (N, A)$ is a set $Z \subseteq A$ of arcs such that each path in G contains at most one arc in Z.*

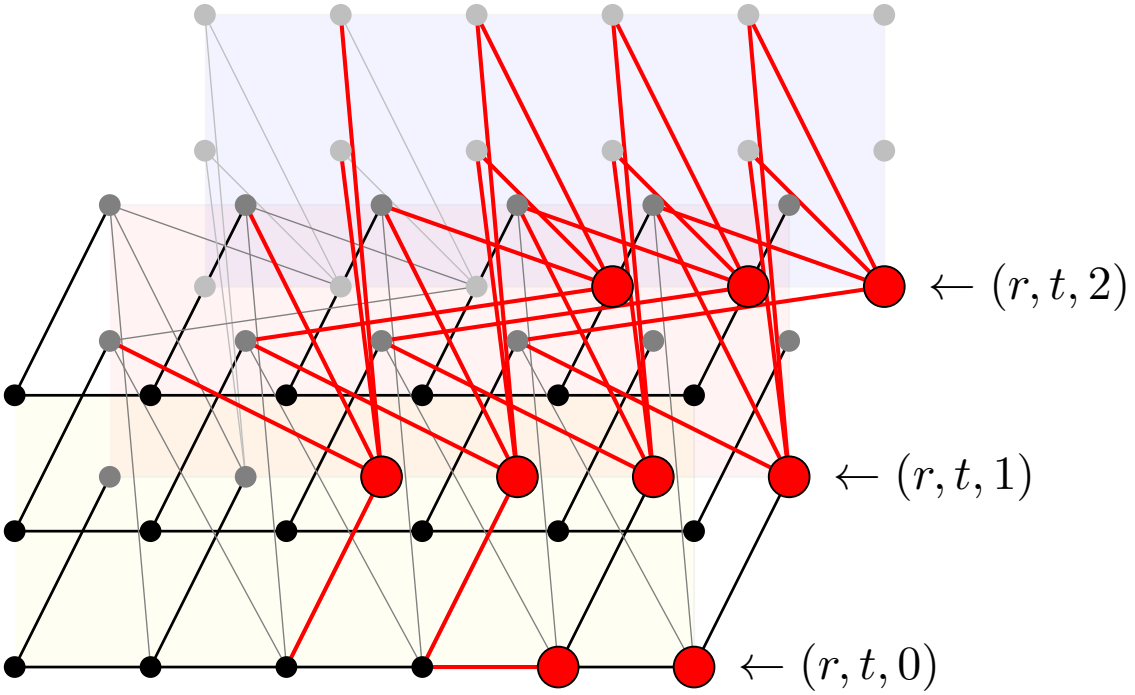


Figure 6.11: An illustration of $W_{r,t}$ and $A_{r,t}$ for a specific time-space resource $(r, t) \in N_r \times \mathcal{T}$ in an example graph G. The nodes in $W_{r,t}$ and arcs in $A_{r,t}$ are highlighted in red. The relevant times are $h_r = 1$, $L_r^0 = 0$, $L_r^1 = 2$ and $L_r^2 = 1$.

Definition 6. *For two routes $r_1, r_2 \in N_r$, let*

$$L(r_1, r_2) = \min \left\{ \sum_{r' \in p \setminus r_2} \min_{v=1,2} L_{r'}^v : p \text{ is an } r_1\text{-}r_2 \text{ path in } G_r \right\}$$

be a lower bound on the minimum total traversal time from r_1 up to but not including r_2. For example, if $(r_1, r_2) \in A_r$ then $L(r_1, r_2) = \min\{L_{r_1}^1, L_{r_1}^2\}$.

Assumption 3. *For each $(r, t) \in N_0$, $A_{r,t} \cap \bar{A}_{r,t} = \emptyset$ and $A_{r,t} \cup \bar{A}_{r,t}$ is an antichain in G.*

Proposition 4. *Assumption 3 is true if for each $r \in N_r$, either (i) $S_r \cup \{r\}$ is an antichain in G_r or (ii) $L(r_1, r_2) \geq \max_v L_{r_1}^v + h_{r_1}$ for each pair $r_1, r_2 \in S_r \cup \{r\}$.*

Proof. Let $(r, t) \in N_0$. Since $r \notin S_r$ by definition, $W_{r,t} \cap \bar{W}_{r,t} = \emptyset$ and hence $A_{r,t} \cap \bar{A}_{r,t} = \emptyset$.

Suppose for contradiction that at least one of (i) and (ii) holds, and that there is a *source-sink* path p in G that contains two distinct arcs $((r_0, t_0, w_0), (r_1, t_1, w_1))$ and $((r_2, t_2, w_2), (r_3, t_3, w_3))$ in $A_{r,t} \cup \bar{A}_{r,t}$, where without loss of generality $t_1 \leq t_3$. By the definitions of $A_{r,t}$ and $\bar{A}_{r,t}$, the routes r_1 and r_3 are in $S_r \cup \{r\}$. However, since $t_1 \leq t_3$, there must be a subpath q of p from (r_1, t_1, w_1) to (r_3, t_3, w_3), and hence a path from r_1 to r_3 in G_r, contradicting (i). Furthermore,

$$\begin{aligned} t - (L_{r_1}^{w_1} + h_{r_1}) + 1 + L(r_1, r_3) &\leq t_1 + L(r_1, r_3) && (r_1, t_1, w_1) \in W_{r,t} \cup \bar{W}_{r,t} \\ &\leq t_3 && q \text{ must span at least } L(r_1, r_3) \text{ time intervals} \\ &\leq t. && (r_3, t_3, w_3) \in W_{r,t} \cup \bar{W}_{r,t} \end{aligned}$$

This rearranges to $L(r_1, r_3) \leq L_{r_1}^{w_1} + h_{r_1} - 1$, which contradicts (ii). Therefore neither (i) nor (ii) holds, and the result is proved by contradiction. □

Note that the conditions of Proposition 4 can be checked much more easily than checking Assumption 3 directly. Condition (i) is satisfied for all routes in our instances for Derby station.

6.5.4 Objective Function and Arc Weights

The quality of a given solution to the TTRP is measured as the sum of the weights of the paths in the solution. A smaller sum corresponds to a better solution, so the problem is a minimisation problem. The weight of a given path for train k is the sum of the weights c_a^k of the arcs a that make up the path. These weights depend on both the arc and the particular train, so each arc a in G has a set of weights $\{c_a^k : k \in \mathcal{K}\}$.

The weights are selected such that the negative of the sum of the weights of the paths in a solution corresponds to the utility associated with the solution by Network Rail.

This utility is modelled as the total weighted utility over all trains k and scheduled events j, given by

$$U = \sum_{k \in \mathcal{K}} \alpha_k \sum_{j=1}^{J^k} \beta_k^j U_k^j.$$

Train priorities and event priorities are controlled by parameters α_k and β_k^j, respectively. The set of events J_k for each train k includes all of its station stops and its exit from the modelled area. Each event j specifies both a route r_j^k and a time interval a_j^k in which the train should arrive into the route. The quantity U_k^j is the utility accrued by train k at stop j. It is equal to zero if train k does not visit r_j at all, and $\gamma(t - a_j^k) = \phi^{-\omega|t-a_j^k|}$ if train k enters r_j in time interval t (and hence $t - a_j^k$ time intervals late).

To facilitate the representation of this objective function, the weights c_a^k take the following values:

- $c_a^k = 0$ if $a \in A_2, A_3, A_5, A_6$ for all $k \in \mathcal{K}$.
- $c_a^k = \infty \mathbb{1}_{\{j \neq k\}}$ for all $k \in \mathcal{K}$ and $a = (source, (r_0^j, a_0^j, v_0^j)) \in A_1$. This ensures that trains begin in their initial position, rather than the initial position of a different train.
- Let $j \in J^k$ be a scheduled event for train $k \in \mathcal{K}$ that requires train k to arrive into route r_j^k at time interval a_j^k. Let $t \in \mathcal{T}$ be a time interval. Then:
  - If the event requires the train to stop at a platform, then all arcs $a \in A_4$ that enter node (r_j^k, t, v_0) have weight $c_a^k = -\alpha_k \beta_k^j \gamma(t - a_j^k)$.
  - If the event doesn't require the train to stop (e.g. passing a junction), then the weight $c_a^k = -\alpha_k \beta_k^j \gamma(t - a_j^k)$ applies to arcs in $a \in A_4$ that enter any of the nodes (r_j^k, t, v) where $v \in N_v^{r_j^k}$.
  - If the event has a departure time that is later than t, then all arcs $a \in A_4$

starting at any of the nodes (r_j^k, t, v), where $v \in N_v^{r_j^k}$, have $c_a^k = \infty$. This prevents train k from leaving r_j^k for another route before the scheduled departure time.

All other arcs $a \in A_4$ have weight $c_a^k = 0$ for each k.

Note that weights c_a^k with value ∞ ensure that an optimal solution will never contain a path for train k that contains arc a. In computations, a sufficiently large floating point value is used to represent ∞.

6.5.5 MIP Formulation and Solution

Although the model presented in this paper differs from that presented by Reynolds et al. [2020], it can be formulated as a Mixed Integer Program analogously and therefore solved using the same branch-and-price algorithm. This is because the differences between the models are expressed in the definitions of both the graph G on which the problem is defined, and the sets $A_{r,t}$ and $\bar{A}_{r,t}$.

The formulation is a path-based flow formulation with binary variables $\lambda^{k,p}$ used to indicate which *source-sink* path $p \in P^k$ in the TST graph is selected for each train k. Thus, the formulation of the TTRP is given by

$$\min \quad \sum_{k \in \mathcal{K}} \sum_{p \in P^k} \left(\sum_{a \in A} c_a^k x_a^{k,p} \right) \lambda^{k,p} \tag{6.2a}$$

$$\text{s.t.} \quad \sum_{k \in \mathcal{K}} \sum_{p \in P^k} \left(\sum_{a \in A_{r,t}} x_a^{k,p} + \delta \sum_{a \in \bar{A}_{r,t}} x_a^{k,p} \right) \lambda^{k,p} \leq 1 \quad \forall (r,t) \in N_r \times \mathcal{T} \tag{6.2b}$$

$$\sum_{p \in P^k} \lambda^{k,p} = 1 \quad \forall k \in \mathcal{K} \tag{6.2c}$$

$$\lambda^{k,p} \in \{0,1\} \quad \forall k \in \mathcal{K}, p \in P^k, \tag{6.2d}$$

where $\delta > 0$ is a small positive constant, and each $x_a^{k,p}$ is a constant that is equal to 1 if path $p \in P^k$ for train k contains arc $a \in A$, and equal to 0 otherwise.

The objective (6.2a) is to minimise the total weight of all of the selected train paths. This corresponds to maximising Network Rail's utility. Constraint (6.2b) ensures that the track capacity constraints outlined in Section 6.5.2 are respected. For example, if a time-space resource (r, t) is occupied twice, then the left hand side of (6.2b) is 2, violating the constraint. Similarly, if (r, t) is both occupied and banned, then the left hand side of (6.2b) is $1 + \delta$, which also violates the constraint. However, the constant δ is given a small value so that the resource can be banned multiple times without violating the constraint. In our instances, a value of $\delta = 0.05$ was sufficient to ensure that there was no practical constraint on the number of times a resource could be banned. Constraint (6.2c) ensures that exactly one path is selected for each train. Finally, constraint (6.2d) states that the variables are all binary.

This problem can be solved using the branch-and-price algorithm described by Reynolds et al. [2020]. Column Generation is used to solve the LP relaxation at each node of the branch-and-bound tree. Variables generated in each column generation iteration by the solution of one subproblem for each train. These subproblems are shortest-path problems on G, which is a directed acyclic graph. Both partial pricing and reduced cost variable fixing are used as column generation acceleration strategies. A customised branching rule is used that branches on conflicts between pairs of trains over resources (r, t) in the track capacity constraints.

6.6 Computational Study

A computational study has been carried out to compare two models:

(FS) The fixed-speed model proposed by Reynolds et al. [2020], with traversal

times calculated using Method 1 (see Section 6.4.1).

(VS) The variable-speed model proposed in this paper, with traversal times calculated using the transition-based Method 3 (see Section 6.4.3).

Section 6.6.1 describes the new set of instances for Derby station in the UK that is used for our computational study. Section 6.6.2 compares the traversal times produced by the different estimation methods. Finally, Section 6.6.3 compares the performance of the branch-and-price solution algorithm for the two models. Full tables of results are available online (see [Reynolds, 2020]).

Traversal time estimation methods were implemented in the Python 3.6 language, using the package Scikit-learn 0.20.3 [Pedregosa et al., 2011] for fitting Gaussian mixture models, and the package SciPy 1.2.1 [Virtanen, 2020] for performing Mood's test. The branch-and-price algorithm is implemented using SCIP 6.0.2 [Gleixner et al., 2018] as a branch-and-price framework. Custom plugins for SCIP are written in the C language, and Gurobi 9.0 [Gurobi Optimization LLC, 2020] is used as the linear programming solver. The experiments were carried out on a computing node equipped with an 18 core Intel Xeon E5-2699 v3 CPU with 2.30GHz and 500GB of RAM, running Ubuntu 16.04.

6.6.1 Instance Data

A new set of 310 instances has been created for the computational study. These instances are based on real data from an area of railway centred around Derby station in the UK. Derby station lies on both the Midland Main Line and the Cross Country Route, two heavily used, double-track, inter-city lines connecting London with Leeds, and Bristol with York via Birmingham, respectively. The station also hosts local services to Nottingham and Matlock. In 2018–2019, Derby station had

3,902,000 passenger entries and exits and 619,000 passenger interchanges. Lying at the confluence of several lines, Derby Station is widely regarded as a traffic bottleneck. This makes it suitable for testing our model.

The area modelled is shown in Figure 6.12. Derby station has 6 bidirectional platforms and 204 track circuits. The area as a whole contains portions of three double track lines, and consists of 142 routes in total, with 228 valid berth transitions. The area includes five small stations in addition to Derby: Spondon, Peartree, Duffield, Belper and Ambergate.

Each one of the 310 instances covers a different hour long period. Ten instances, covering between 8am and 6pm, are used from each of the 31 days in January 2020. They were created using real timetables, and real data about traffic perturbations. The level of traffic perturbation varies considerably between the instances. However, they provide a representative sample of traffic conditions at Derby over January 2020.

The number of trains in each instance is shown as a histogram in Figure 6.13. The most common number of trains is 19, whilst the largest number is 23. The number of *conflicts* in each instance for both (FS) and (VS) is shown in Figure 6.14. This is defined as the number of track capacity constraints that are violated by the solution obtained from solving each model without any track capacity constraints. The number of conflicts in a given instance can be different for models (FS) and (VS) because the differing traversal times result in trains reaching different parts of the track at different times. Nevertheless, we see that there is a strong positive relationship because although the models and traversal times differ, the TTRP instances used are identical. Table 6.1 compares the number of *train pair conflicts* in the instances when using (FS) and (VS). This is the number of unique train pairs involved in at least one conflict together. Measuring train pair conflicts can be more informative

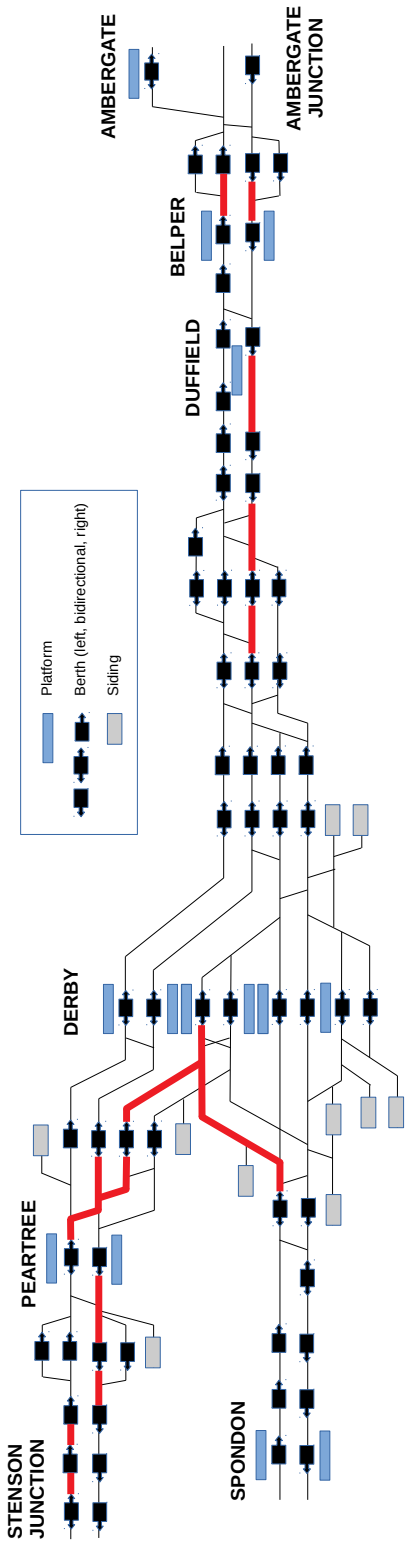


Figure 6.12: A berth diagram of the modelled area (own image). Routes with multiple traversal times in (VS) are highlighted with thick red lines (see Section 6.6.2).

than measuring conflicts. This is because whilst several conflicts can occur for the same train pair in consecutive time intervals over the same route, each train pair can only have at most one train pair conflict. Both the number of conflicts and the number of train pair conflicts have been shown by Reynolds et al. [2020] to be correlated with the number of branch-and-bound nodes required to solve instances to optimality. Discrepancies between (FS) and (VS) can therefore cause differences in the performance of the solution algorithm.

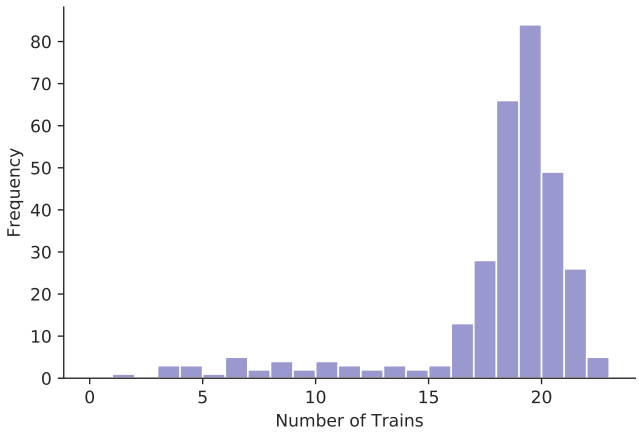


Figure 6.13: A histogram showing the distribution of the number of trains in the instances.

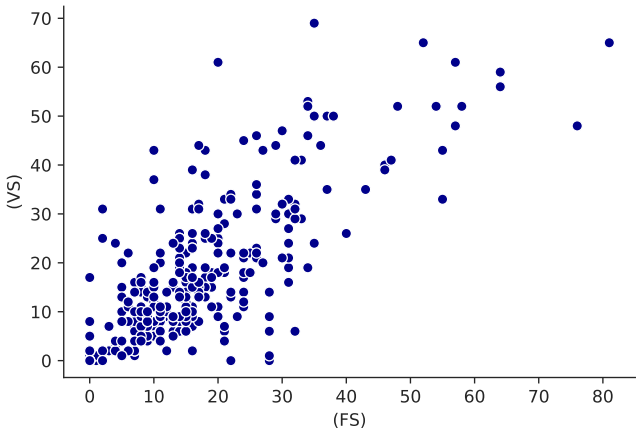


Figure 6.14: A comparison between (FS) and (VS) of number of conflicts.

		(VS)								
		0	1	2	3	4	5	6	7	8
	0	16	6	1	0	0	0	0	0	0
	1	2	17	10	4	1	0	0	0	0
	2	1	10	40	26	6	3	0	0	0
(FS)	3	1	5	22	33	18	9	0	1	0
	4	0	0	8	10	14	9	1	1	0
	5	0	1	0	1	4	6	7	2	0
	6	0	0	0	0	2	2	4	0	1
	7	0	0	0	0	1	0	1	2	1
	8	0	0	0	0	0	0	0	0	0

Table 6.1: A comparison between (FS) and (VS) of train pair conflicts. Values are frequencies of instances.

6.6.2 Traversal Time Estimation

To test (VS), historical Train Describer data was used to estimate traversal times for routes in Derby station using method 3 (see Section 6.4.3). The results show that 15 out of the 140 routes were given two different traversal times. For these 15 routes, sufficient evidence was found for a difference of at least one time interval between the traversal times of speed profile types v_1 and v_2 — sufficient evidence was not found for the remainder of the routes. This shows that our statistical method requires a high standard of evidence for using two traversal times for a route. The effect of this is that nodes of speed profile type v_2 in the TST graph are included sparingly, resulting in a parsimonious optimisation model. Including two traversal times for a route introduces additional complexity and increases the size of the model, so it is important to do this only where it can be shown statistically to be most important.

The routes with two traversal times are highlighted in Figure 6.12. Some of these routes are adjacent to routes that frequently have conflicts, such as the routes entering platform 3 (the third platform from the top in Figure 6.12) at Derby station. This is likely to arise from the fact that some trains stop in preceding routes as a result of

conflicts, whilst others do not and this affects the speed at which they traverse the route. Other routes with two traversal times are adjacent to stations at which only some trains stop, such as Peartree, Duffield and Belper. Trains that stop at these stations must decelerate or accelerate through adjacent routes, whilst trains that don't can continue at the line speed. These patterns conform to our expectations and therefore give us confidence that the traversal time estimation method is performing well.

The traversal times of routes that have three speed profile types in the (VS) model (using Method 3) are shown in Figure 6.15 alongside the corresponding traversal times for (FS) (which uses Method 1). It shows that traversal times in (FS) are often similar to or the same as for traversal type v_2 for (VS). This is because the smallest cluster mean for each route is used as the traversal time in (FS), and the observations in that cluster overlap strongly with observations that are categorised as type v_2 in (VS). Whilst the difference between traversal times for v_1 and v_2 in (VS) is not large for most routes, even a single time interval (10 seconds) difference can be enough to affect the optimal rescheduled timetable. The traversal times $L_r^1 = 12$ and $L_r^2 = 6$ for route $r = \texttt{DY551\_DC5108}$ differ by a whole minute. This is a relatively long route on the open line in which trains stopping at Duffield station and trains not stopping there are likely to be travelling at very different speeds.

6.6.3 Algorithmic Performance

To understand how our branch-and-price algorithm performs when solving both models (FS) and (VS), each of the real instances from Derby station were solved with a time limit of 600 seconds. Within this time limit, 282 of the 310 instances were solved to optimality with (FS), and 287 were solved to optimality with (VS). This is a good result because it means that all but the most difficult instances can be

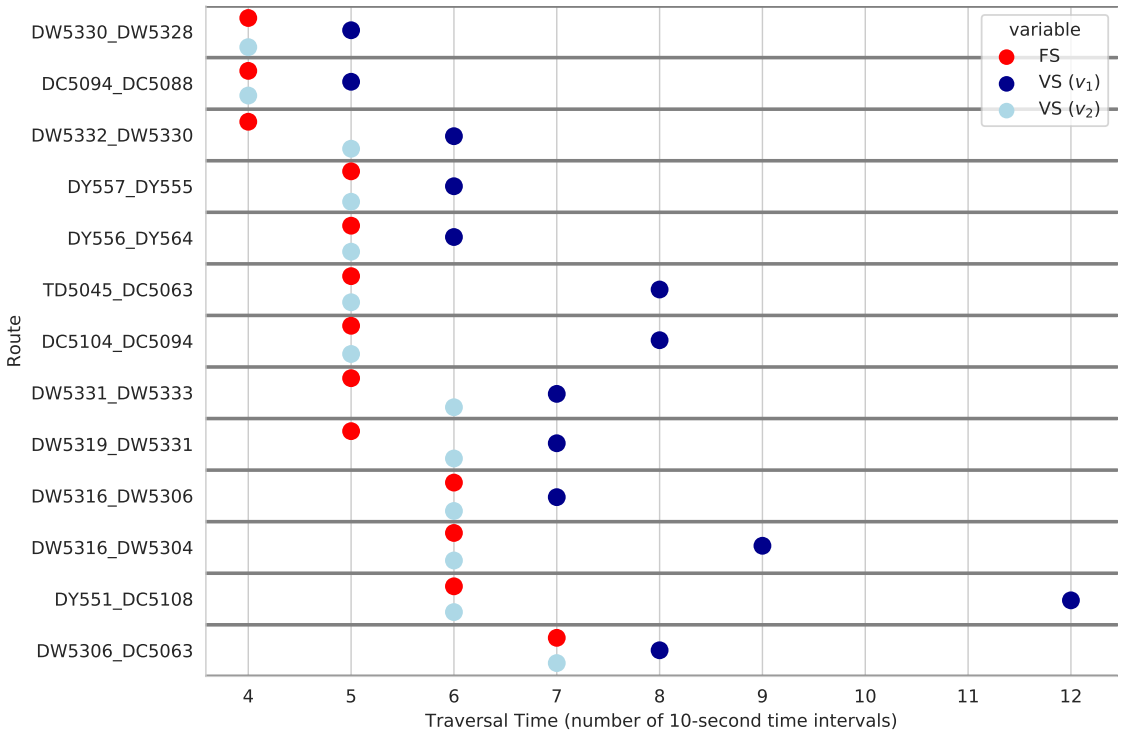


Figure 6.15: A comparison between (FS) and (VS) of traversal time lengths for routes that have two traversal times in (VS).

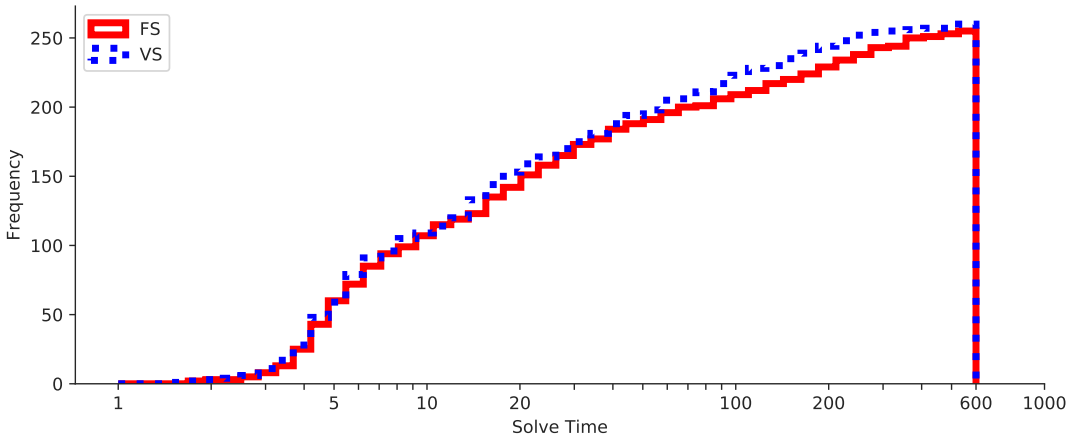
solved. The solving times of the instances that could be solved to optimality are shown in Figure 6.16(a) (note that the x-axis is logarithmic with base 2). This figure demonstrates that the distribution of solving times over the instance set are very similar for (FS) and (VS). However, the higher blue line towards the right hand side of the plot indicates that (VS) performed slightly better than (FS) among models that took over 100 seconds to solve.

The explanation for the slightly better performance of (VS) on difficult instances can be seen by studying Figures 6.16(b) and 6.16(c). These show the number of branch-and-bound nodes explored, and the number of variables (columns) generated, respectively, during the solving process. Figure 6.16(c) shows that the number of columns generated was almost identical, indicating that convergence of the column generation algorithm for solving LP relaxations was not affected by differences between (FS) and (VS). On the other hand, Figure 6.16(b) shows that slightly higher

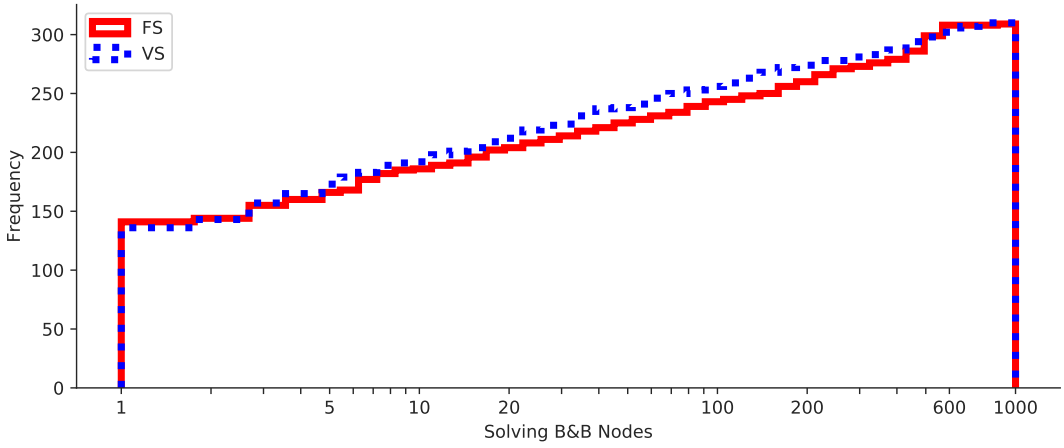
numbers of branch-and-bound nodes were generated with (VS), particularity on instances with between 20 and 600 nodes. This is the likely to be the reason for the slightly better solution times with (VS) on difficult instances. It is conjectured that this small difference in the numbers of branch-and-bound nodes arises from differences in the quantity and quality of conflicts due to differing traversal times.

Both of the models were also evaluated using a time limit of 20 seconds. This reflects the short amount of time that is typically available for solving TTRP instances in practical, real-time environments. Of the 310 instances, 174 were solved to optimality by both models within this time limit. The optimality gaps for the remaining 136 instances are plotted in Figure 6.17. It shows that with both (FS) and (VS), high-quality solutions that are provably within 10% of optimality were found for the vast majority of instances. For each, only two instances had remaining optimality gaps exceeding 20%. This reinforces the evidence found by Reynolds et al. [2020], using instances for a different station, that the algorithm used is suitable for real-time operations. Figure 6.17 also shows that whilst the distribution of remaining optimality gaps is similar for (FS) and (VS), (VS) has a slightly larger concentration of instances with very small gaps. This corroborates our observation of better performance for (VS) when the time limit was 600 seconds. However, the difference is very marginal in both cases.

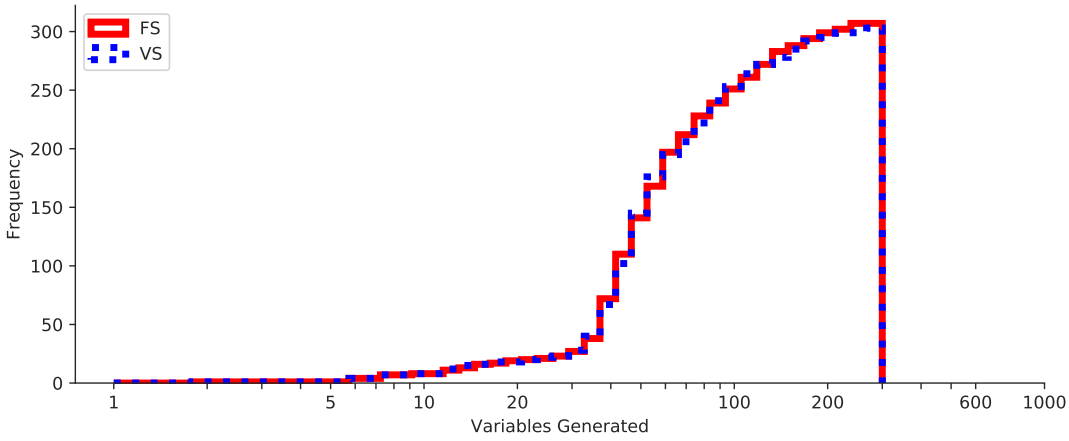
Based on the results presented, we conclude that the differences in algorithmic performance between (FS) and (VS) are negligible. If anything, the performance of (VS) slightly exceeds that of (FS) on difficult instances. This means that the benefits of variable speed modelling in (VS) compared with fixed speed modelling in (FS) come at no price for computational performance. That is a striking result, because one might expect solving variable-speed models to be more time consuming than solving fixed-speed models.



(a)



(b)



(c)

Figure 6.16: Cumulative histograms showing (a) solution time (seconds), (b) number of branch-and-bound nodes and (c) number of variables generated, over the set of instances.

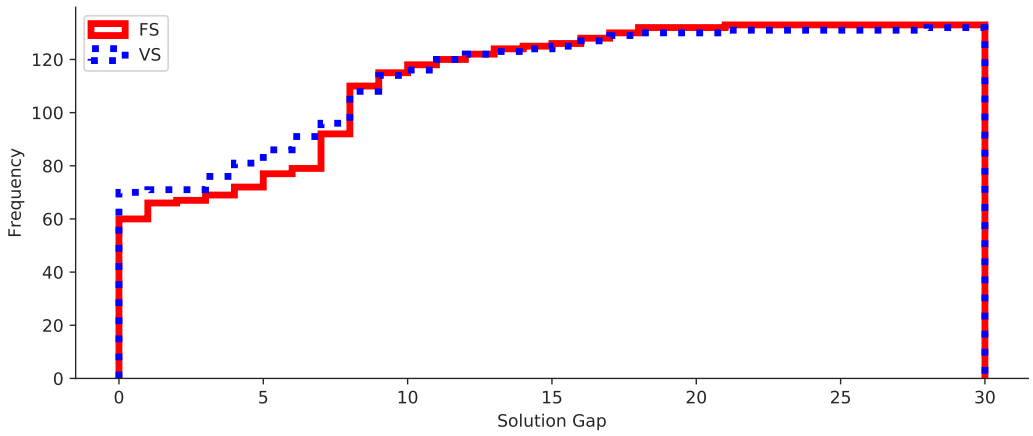


Figure 6.17: Cumulative histogram of optimality gaps after 20 seconds for (FS) and (VS).

6.7 Conclusion

In this paper, we present a new variable-speed model for the TTRP that uses a time-space-type graph to approximate train speed profiles. To achieve this, the notion of a discrete speed profile type is introduced, and techniques for estimating traversal times based on historical Train Describer data are developed.

Our approach is tested using a new set of real instances for Derby station in the UK. These tests show that our data-driven approach results in a parsimonious model that is able to improve speed profile modelling relative to fixed-speed models in some parts of the track network. In addition, these modelling enhancements come at no cost, in the sense that they do not lead to longer solving times in comparison with the fixed-speed model of Reynolds et al. [2020]. This is a major benefit of our approach to modelling speed profiles.

Further work is needed to quantify the inaccuracies in speed profile modelling that still remain. For example, a simulation study using a microscopic railway simulator could be used to evaluate solutions produced by our model and compare them to solutions produced by different variable speed models. Our data-driven modelling

approach also creates exciting opportunities to improve the statistical methodology for estimating traversal times based on speed profile types. One possibility is to use a Hidden Markov Model (see [Zucchini et al., 2016] for an introduction), in which speed profile types are the 'hidden' states, and traversal times are the observed data. This might make it possible to fit mixture models for each route (as in Method 2) whilst additionally using the full transition data.

Chapter 7

An evaluation of the fairness of railway timetable rescheduling in the presence of competition between train operators

7.1 Introduction

Effective real-time management of railway traffic is crucial to delivering good railway performance. In particular, making changes to the timetable in response to an initial delay can help to reduce the amount of additional delay caused to other trains as a result of the initial incident. This practice is known as *timetable rescheduling*. The Train Timetable Rescheduling Problem (TTRP) [Cacchiani et al., 2014] can be solved in order to determine the optimal way to reschedule the timetable. A large number of different TTRP problem variants, models, objective functions and solution methods have been studied.

However, the implications for TTRP models of economic competition between rail-

way operators has not been considered. In recent decades, different forms of competition have been introduced in several European railway systems, such as those of Germany, Great Britain and Sweden [IBM, 2011]. Where trains are operated by more than one different company over the same tracks, timetable rescheduling has the potential to impact these operators unequally. In order to be perceived as fair, a TTRP model must not systematically favour some operators over others. A perception of unfairness would be a serious barrier to the practical deployment of TTRP models in competitive railway systems. Therefore, it is essential that the fairness characteristics of such models are understood.

This study investigates the fairness of solutions obtained from solving the TTRP. We use a case study of Doncaster railway station in Great Britain during January 2017, during which time trains from seven different private passenger operators regularly used the station. The TTRP model proposed by Reynolds et al. [2020] is used to calculate the rescheduled timetables that are evaluated. First, appropriate definitions of fairness and efficiency are proposed for the TTRP. Second, the fairness of efficiency-maximising solutions is analysed. Third, the sources of unfairness in efficiency-maximising solutions are elucidated via an investigation of the competitive relationship between each pair of operators. Finally, the changes in efficiency and fairness that arise from varying the priority given to class 1 (express passenger) trains over class 2 (ordinary passenger) trains is examined. Class 1 trains are often prioritised in current manual rescheduling practice because they typically carry more passengers and travel longer distances.

This research has been carried out in partnership with Network Rail, a public sector company that owns and operates railway infrastructure in Great Britain. The railway system in Great Britain is vertically separated, meaning that passenger and freight services are operated by companies that are separated from Network Rail. These companies are called Train Operating Companies (TOCs) and Freight Oper-

ating Companies (FOCs), respectively — we will simply call them *operators*. Most passenger operators run services under competitively tendered franchise agreements. These franchises each confer the right to operate a set of services grouped by type or region over a period of several years. Open-access operators, on the other hand, run services in purchased timetable slots. The organisational structure of the system, which is likely to change in the future, is described by Williams [2019a].

Fairness to operators is taken into consideration by Network Rail during the timetable rescheduling process. Rescheduling is currently carried out in accordance with local *Route Regulating Policies*. One of the stated objectives of these policies is to "minimise delay to all [operators] ... in the best interests of all" — unfair treatment of an operator would contradict this. Moreover, delay management teams in Network Rail's Rail Operating Centres (where timetable rescheduling decisions are made) include representatives from operators working collaboratively to achieve this goal. The introduction of TTRP models to these settings relies on gaining the acceptance of key stakeholders such as operators, which is unlikely unless any proposed model can be shown to be fair.

The railway system in Great Britain has a performance-based financial compensation mechanism called Schedule 8 (see [Network Rail, 2019b]). Under this system, all delays are recorded and attributed to a particular cause. Every four weeks, this information is used to calculate the value of monetary payments to be made from Network Rail to operators and vice versa. This mechanism partially protects operators from losses resulting from events that are outside of their control. For example, if an incident caused by one operator causes delay to the train of a second operator, this second operator should receive financial compensation. However, the extent to which cash payments can compensate for delays is limited. This is partly because accurate valuation of the losses resulting from reputational damage to the operator is challenging. It is also unsatisfactory for the passengers of operators, who may

derive social benefits from train punctuality that cannot be financially valued. It is still preferable, therefore, to carry out timetable rescheduling as fairly as possible, and reduce reliance on financial compensation.

This paper is organised as follows. In Section 7.2, the relevant literature is reviewed. The case-study is introduced in Section 7.3. Our definitions of fairness and efficiency are described in Section 7.4. In Section 7.5, an analysis of the fairness of efficiency-maximising TTRP solutions is given. This is supplemented in Section 7.6 by an analysis of the interactions between pairs of operators. Finally, we consider the fairness-efficiency trade-off in Section 7.7. Section 7.8 offers conclusions and suggests some possible avenues of further research.

7.2 Literature Review

The literature addressing the TTRP is already very well documented by a number of recent reviews [Cacchiani et al., 2014; Corman and Meng, 2015; Fang et al., 2015; Lamorgese et al., 2018]. Our literature review will focus specifically on fairness, and objective functions. An overview of relevant literature on the topic of fairness is given in Section 7.2.1. This is followed up in Section 7.2.2 by a discussion of how these concepts apply to objective functions that have been used in the TTRP literature.

7.2.1 Fairness

The concept of fairness has been extensively studied by economists. It most often arises in the context of allocating or sharing limited benefits or resources between distinct entities. This is precisely the scenario faced during timetable rescheduling.

We can think of the distinct entities as the different operators, and the resources to be allocated as the available track capacity. Track capacity consists of segments of track over time, and is limited by the available infrastructure.

Much of the classical literature considering fairness considers the social welfare problem — how a central planner should allocate goods in an economy. In the framework developed by Bergson [1938] and Samuelson [1947], the preferences of each entity $i \in \{1, \ldots, n\}$ can be represented by a cardinal utility $u_i \geq 0$ that depends on the allocation that entity i receives. The set $\mathcal{U} \subset \mathbb{R}^n_+$ of vectors of utilities $(u_1, \ldots, u_n)$ that arise from feasible allocations is called the *utility possibility set*. The problem for the central planner can be formulated as choosing a point from this set. One way of choosing a point is to find a point that maximises a *social welfare function* $f : \mathcal{U} \to \mathbb{R}^+$ that encodes value judgements about the size and distribution of the utilities of the entities.

Social welfare functions can be used to model different attitudes about inequality and fairness. For example, the utilitarian function $f(u) = \sum_{i=1}^n u_i$ is completely indifferent to inequality — it values improvements in the utility of each entity equally, regardless of how high the utility already is. At the other end of the scale, the minimax function $f(u) = \min_{i=1,\ldots,n} u_i$ always seeks improvement in the utility of the worst-off entity, and is indifferent to improvements in the utility of any other entity. Atkinson [1970] proposed maximising the α-fairness welfare function

$$W_\alpha(u) = \begin{cases} \sum_{i=1}^n \frac{u_i^{1-\alpha}}{1-\alpha} & \alpha \geq 0, \alpha \neq 1 \\ \sum_{i=1}^n \log(u_i) & \alpha = 1, \end{cases}$$

which uses the parameter α to interpolate between these two extreme attitudes to inequality. When $\alpha = 0$, the function becomes the utilitarian function and when $\alpha \to \infty$, it converges to the minimax function. Another special case, when $\alpha = 1$,

corresponds to the rule proposed by Nash [1950]. For any value of α, the function W_α has the property of constant elasticity: given a fixed proportional increase in an entity's utility, the proportional increase in the welfare of that entity does not depend on the size of their utility. A detailed introduction to the theory of economic inequality is provided by Foster and Sen [1997].

Another framework in which fairness has been studied is the bargaining problem. In this problem, distinct entities must agree on how to share a jointly generated surplus. Although this is not exactly the situation that arises in timetable rescheduling, it is still an interesting way to understand fairness. For the two-player version, both Nash [1950] and Kalai and Smorodinsky [1975] have proposed axioms that must be satisfied by solutions to this problem before they can be considered fair. Each of these two studies additionally proposes solutions that satisfy these axioms. However, there is no general consensus on which set of axioms should be used. Moreover, axiomatic approaches are not useful in this application because they cannot be used to evaluate the fairness of a solution that does not satisfy the axioms.

In most applied decision making scenarios in which fairness is important, it is nevertheless not the only consideration. The fairness of any solution must usually be considered alongside its overall efficiency. The efficiency can be defined as the sum of the utilities of the entities (the utilitarian function), or in a problem specific way if this is more appropriate. Unsurprisingly, a trade-off has been observed between efficiency and fairness in many problems. In other words, greater fairness can often only be achieved by sacrificing some efficiency, whilst greater efficiency entails a reduction in fairness. This trade-off has been characterised theoretically by Bertsimas et al. [2011] and Bertsimas et al. [2012] using the concept of the price of fairness. This is defined as the proportion of total system efficiency lost by maximising a fairness objective compared with maximising efficiency. They argue that analysing the price of fairness is a useful way to navigate the fairness-efficiency trade-off.

7.2.2 Fairness and the TTRP

As far as the authors are aware, no previous studies have analysed the fairness of a TTRP model with respect to different operators. One potential reason for this is that only a few railway systems are sufficiently liberalised to have multiple operators regularly running trains over the same track. Moreover, fairness and competition in railway markets are often less important to policy makers than overall system performance.

However, fairness has been studied for similar applications of optimisation to transport planning problems, including railway planning problems. For example, Gestrelius et al. [2020] consider the evaluation of railway timetables from a competition management perspective. They report that research literature and guidelines for facilitating competition between operators in timetable design are scarce. In a more quantitative study, Li et al. [2019] investigate the fairness-efficiency trade-off for a train timetabling application, in which α-fairness between passengers was considered. Other examples of similar applications in which fairness has been considered include railway crew scheduling [Jütte et al., 2017], railway crew rostering [Breugem et al., 2019], air traffic flow management [Bertsimas and Gupta, 2016] and airport slot allocation [Fairbrother et al., 2020].

Many different objective functions have been used for the TTRP. Almost all of these can be written in the form $\max f(u_1(x), \ldots, u_n(x))$, where x is the vector of variables in the optimisation problem, each u_i is a function measuring the quality of the solution for train i, and f is a real-valued function. Examples of quantities measured by u_i (or $-u_i$ in the case of minimisation) include consecutive delay (delay experienced as a result of interaction with other trains during the rescheduled period) [Corman et al., 2010a; Pellegrini et al., 2014], deviation from planned schedule [Meng and Zhou, 2014; Lusby et al., 2013], a piecewise linear function of delay [Lamorgese and

Mannino, 2015], and the cost of delays [Törnquist and Persson, 2007]. In many studies, u_i aggregates several measures, either through development of a utility function [Harrod, 2011; Binder et al., 2017; Reynolds et al., 2020] or otherwise [Bettinelli et al., 2017; Caimi et al., 2012; Acuna-Agost et al., 2011b].

From a fairness perspective, the choice of function f is more important. This is because it can be interpreted as a social welfare function where the entities are the trains, and the welfare of each train is a proxy for the welfare of the passengers on the train. In other words, f can encode a particular level of aversion to inequality between the performance of individual trains. The most popular choice in the literature is to use a sum, which corresponds to the utilitarian function [Törnquist and Persson, 2007; Meng and Zhou, 2014; Pellegrini et al., 2014; Lamorgese and Mannino, 2015]. Weighted sums have also been used to establish train priorities [Lusby et al., 2013; Pellegrini et al., 2014]. Since this can lead to large inequalities in the delays experienced by different trains, some authors such as D'Ariano et al. [2007a] and Corman et al. [2010b], have used a minimum function, corresponding to minimax fairness. A computational comparison between minimising the total and maximum consecutive delay, respectively, has been carried out by Pellegrini et al. [2014]. Despite the variety of functions f used and their implications for fairness, there has been very little discussion of fairness in the literature. As a result, the effect of using objective functions with different levels of inequality aversion is not well understood.

Several researchers have used multi-criteria methods to solve instances of the TTRP. Whilst fairness has not been explicitly formulated, some of these studies use objectives that could help to achieve fairness. For example, minimising the number of cancelled trains or missed connections helps to achieve fairness because cancelled trains or those sufficiently late to break connections are likely to be the worst affected trains. A weighted-sum approach is used by Caimi et al. [2012] to optimise

the number of scheduled trains, the weighted delay and the number of connections maintained. Corman et al. [2012a] use a specialised metaheursitc to consider the trade-off between total delay and the number of missed connections. Samà et al. [2015] present a Data Envelopment Analysis methodology for evaluating solutions under a large number of different punctuality measures, including objectives of minimax type. Corman et al. [2011a] propose a lexicographic objective, in which trains are divided into different priority classes. Other studies adopting multi-criteria approaches include [Ginkel and Schöbel, 2007; Cavone et al., 2017; Binder et al., 2017; Shakibayifar et al., 2018; Josyula, 2019; Altazin et al., 2020]. None of these studies analyse the effect of using multi-criteria methods on fairness.

In this paper, we make the following contributions:

1. We propose measures for both the efficiency and fairness of solutions to a set of TTRP instances.
2. We identify the relative weighting of class 1 and 2 trains in the objective function as a key driver of unfairness between operators.
3. We then investigate the trade-off between efficiency and fairness that is created by changing this parameter using a case study based on real data from Doncaster station in the UK.

7.3 The Case Study

An area of railway around Doncaster station has been used as a case study for evaluating fairness. Definitions of any railway signalling terminology used below can be found in [Reynolds et al., 2020]. Doncaster station (see Figure 7.1) lies on the East Coast Main Line, a busy railway corridor connecting London with Leeds, York,

Newcastle and Edinburgh. The wider area covered (see Figure 7.2) also contains portions of four double track lines that all begin at Doncaster and go towards Sheffield, Lincoln, Leeds and Hull, respectively. The area lies within a single area of signalling control, and contains 225 berths with 313 valid berth transitions. The station itself has 9 platforms and 85 track circuits. Doncaster station is an important interchange for a variety of inter-city and local services operated by seven different operators. It is also a busy bottleneck, with over 30 trains per hour at peak times. This makes it ideal for investigating the interactions between different operators.

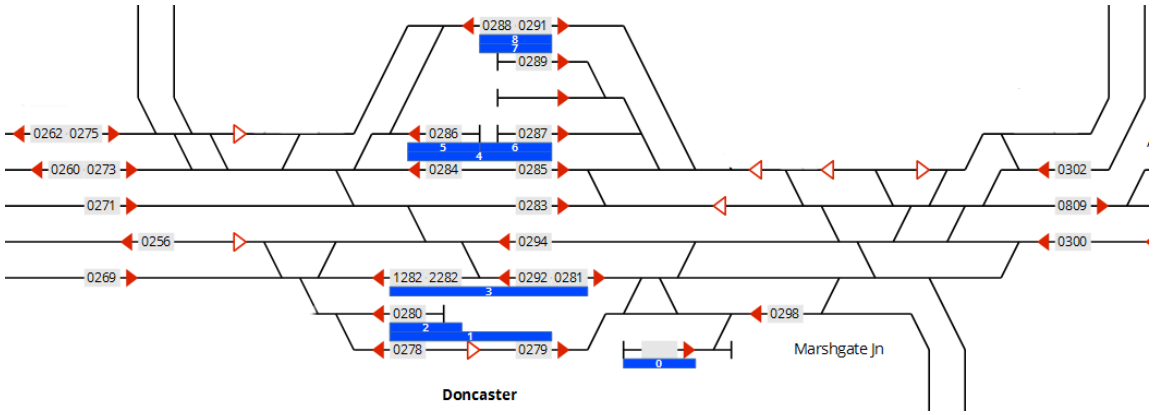


Figure 7.1: A berth diagram of Doncaster Station. Retrieved from `https://wiki.openraildata.com/index.php?title=DR`. Accessed 14/04/20.

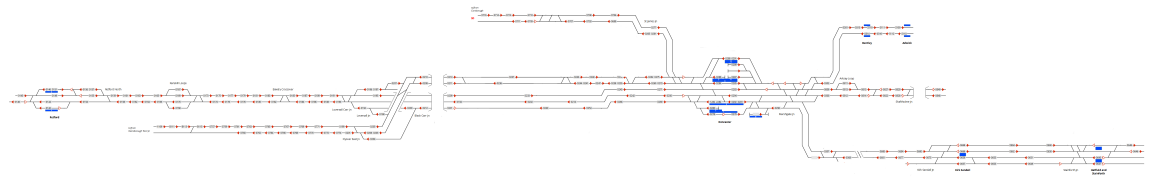


Figure 7.2: A berth diagram of the area of track modelled. Retrieved from `https://wiki.openraildata.com/index.php?title=DR`. Accessed 14/04/20.

The data for the case study comes from January 2017. The seven different passenger operators running services through the area during this month are displayed in Table 7.1. Abbreviations will be used to refer to the operators throughout. Freight operators have been excluded from the analysis because freight services are often scheduled during less busy periods and are affected by rescheduling in different ways

to passenger operators. Table 7.1 shows that NT and LNER operated the most services during the period. Whilst most of the operators run inter-city (class 1) services, NT and East Midlands Railway run mostly local (class 2) services. NT, in particular, is responsible for the vast majority of class 2 trains. During this period, Grand Central and Hull Trains were open-access operators, while the remainder were operating franchises.

Operator	Abbr.	# Trains	# Class 1	# Class 2
Northern Trains	NT	4105	730	3375
London North Eastern Railway	LNER	3492	3492	0
Transpennine Express	TPE	825	825	0
Cross Country	XC	737	737	0
Grand Central	GC	496	496	0
Hull Trains	HT	354	354	0
East Midlands Railway	EMR	275	18	257

Table 7.1: The passenger operators in the case study, including the total number and the number of class 1 and class 2 trains run through the modelled area during January 2017.

The month of January 2017 is split into 310 non-overlapping hour-long instances of the TTRP (between 8am and 6pm each day, for 31 days). These instances are created from real historical data about the timetable, and the traffic perturbations that actually occurred. The number of operators running trains in each instance ranges from two to six, with the most common number being five. By using instances that cover a whole month, we are able to understand fairness over the whole month, rather than on an instance-by-instance basis.

7.4 Efficiency and Fairness

In order to evaluate the fairness of railway timetable rescheduling with respect to efficiency, these two terms must first be defined. As far as the authors are aware, no definition of fairness has been proposed specifically for the TTRP.

7.4.1 Efficiency

Our measure of the overall system efficiency was developed with Network Rail (see Reynolds et al. [2020]). It is designed to model the utility of Network Rail, which can be seen as the central decision maker for rescheduling decisions. It is designed to take into account the overall quality of service provided to passengers.

Given a feasible solution x to a given instance, the efficiency is defined as

$$U(x) = \sum_{k \in \mathcal{K}_1} U_k(x) + w \sum_{k \in \mathcal{K}_2} U_k(x), \tag{7.1}$$

where $\mathcal{K}_1$ and $\mathcal{K}_2$ are sets containing the class 1 and class 2 trains, respectively, $U_k(x)$ is the utility accrued from train k, and $w = 0.4$ is a weight that controls the priority given to class 1 trains in comparison to class 2 trains. The priority given to class 1 trains by the value of w reflects the fact that class 1 trains typically carry more passengers. Furthermore, class 1 trains usually complete longer journeys and hence delays to class 1 trains can have a greater impact in terms of reactionary delay outside the geographical scope of the TTRP instance.

The utility $U_k(x)$ accrued from train k is calculated as a weighted average across the set J^k of timetabled events for train k within the area and time horizon modelled:

$$U_k(x) = \sum_{j \in J^k} \beta_k^j U_k^j(x). \tag{7.2}$$

Each event $j \in J^k$ in the timetable for train k corresponds to a particular part of the track, and a time that train k is due to enter it. These can include arrival events at platforms and passing events at junctions or key points along a route. The values of β_k^j ensure that more important events, such as an arrival into Doncaster station or exiting the modelled area, are weighted more highly than events at minor stations. When alternative platforms are available (such as at Doncaster station),

these are separate events $j \in J^k$. For events j at platforms other than the originally scheduled platform, β_k^j is 0.9 times the weight at the originally scheduled platform. This discourages platform changes.

The utility $U_k^j(x)$ accrued by train k at event j is equal to zero if the event j is not carried out. This could occur if the train passes through a timetabled platform without stopping to let passengers alight or depart (a cancelled stop) or if the route of a train is changed so that it does not visit the location specified by event j. Otherwise, $U_k^j(x)$ is equal to

$$\gamma(l) = \begin{cases} \phi^{-\omega|l|} & \text{if } |l| \leq \Gamma \\ 0 & \text{if } |l| > \Gamma, \end{cases} \tag{7.3}$$

where l is the number of 15 second time intervals late that the event occurs. This lateness l is negative if the train is early, and this is penalised equally to positive lateness to discourage station congestion. The parameter values used are $\phi = 1 + (1 \times 10^{-7})$, $\omega = 1 \times 10^5$ and $\Gamma = 240$.

The efficiency $E(\mathbf{x})$ of a set of solutions $\mathbf{x} = (x^i : i \in I)$ to the whole set of instances I can be calculated by summing the individual efficiency of each solution. Denoting the efficiency function U when applied to each instance i as U_i, this can be written as

$$E(\mathbf{x}) = \sum_{i \in I} U_i(x^i). \tag{7.4}$$

7.4.2 Fairness

For a given instance, let the set of operators be O, and let $\mathcal{K}_o \subset \mathcal{K}$ be the set of trains that are operated by operator o. The efficiency function $U(x)$ can be rewritten

as

$$U(x) = \sum_{o \in O} U_o(x), \tag{7.5}$$

where

$$U_o(x) = \sum_{k \in \mathcal{K}_1 \cap \mathcal{K}_o} U_k(x) + w \sum_{k \in \mathcal{K}_2 \cap \mathcal{K}_o} U_k(x) \tag{7.6}$$

is the part of the efficiency arising from trains operated by o. Since $U_o(x)$ includes only trains from operator o, it can be used to measure the utility of operator o.

The utilities $U_o(x)$ are difficult to compare because there might be different numbers of trains with different weights in the instance. For each operator $o \in O$, let x_o^* be an optimal solution when the objective is to maximise $U_o(x)$. Each solution x_o^* represents the best solution operator o can hope for, and $U_o(x_o^*)$ provides an upper bound for $U_o(x)$. This allows us to calculate a normalised utility for each operator

$$\hat{U}_o(x) = \frac{U_o(x)}{U_o(x_o^*)}. \tag{7.7}$$

These values can be compared between operators. A value of $\hat{U}_o(x) = 1$ indicates that operator o realises their maximum possible utility in the rescheduled solution x — all events for all trains are due to be carried out on time and as planned. Conversely, $\hat{U}_o(x) < 1$ indicates that one or more events have been cancelled or rescheduled to occur on a different platform, or late.

A social welfare function (such as α-fairness) could be applied to the set of utilities $\{\hat{U}_o(x) : o \in O\}$ to measure the fairness of the solution x for a single instance. This would allow fairness to be formulated as an objective function so that fairness-maximising solutions to individual instances could be computed to solve the TTRP. It would also open the possibility of using multi-criteria methods to balance the objectives of maximising fairness and maximising efficiency within each instance.

However, when considering operator fairness for timetable rescheduling, it is prob-

lematic to focus on single hour-long instances separately. This is because operators experience fairness and unfairness over a much longer period of time. The operation of a TTRP algorithm on a railway is likely to involve solving hundreds of different instances, involving repeated allocations of track capacity between the same sets of operators. Instead, it is much more appropriate to consider many consecutive instances of the problem as a single, combined allocation problem. That is the approach taken in this paper.

Considering fairness over a whole instance set rather than on an individual instance basis has important implications for fairness. It means that each individual instance need not be fair, provided any operators that loose out can be compensated in other instances. This is crucial when one considers that a typical TTRP instance considers changes to the timetable over a time horizon of only one hour. Many instances involve only a small number of decisions such as which train should go ahead of the other out of a pair of conflicting trains. It may be impossible to resolve such problems in a fair way, or doing so may require a large degradation in efficiency. Our approach of considering fairness over the whole instance set overcomes this issue.

Although fairness is measured over a whole set of instances, each instance must still be optimised individually, as and when delays arise on the railway. This makes it impossible to optimise our measure of fairness. As a result, we do not suggest the measure as a potential objective function, but rather as a tool for fairness evaluation. Unfortunately, this also precludes calculation of the price of fairness [Bertsimas et al., 2011].

Consider a set of solutions $\mathbf{x} = (x^i : i \in I)$ to a set of instances $i \in I$. We index the previous notation by i so that $O_i, U_{i,o}, \hat{U}_{i,o}$ and $x_o^{i,*}$ correspond to the notation $O, U_o, \hat{U}_o$ and x_o^*, respectively, when applied to each instance i. Note that the set of operators O_i can be different across instances.

The normalised aggregated utility for operator o over I can be calculated as

$$\hat{U}_o(\mathbf{x}) = \frac{\sum_{i \in I: o \in O_i} U_{o,i}(x^i)}{\sum_{i \in I: o \in O_i} U_{o,i}(x_o^{i,*})}. \tag{7.8}$$

These values are then used in the α-fairness welfare function to produce our measure of fairness:

$$F_\alpha(\mathbf{x}) = \begin{cases} \sum_{o \in O} \frac{\hat{U}_o(\mathbf{x})^{1-\alpha}}{1-\alpha} & \alpha \geq 0, \alpha \neq 1 \\ \sum_{o \in O} \log \hat{U}_o(\mathbf{x}) & \alpha = 1. \end{cases} \tag{7.9}$$

7.5 Evaluation of Fairness under Maximal Efficiency

The approach taken by Reynolds et al. [2020] and in many other TTRP studies is to maximise the efficiency of the system. It is therefore important to evaluate the fairness of these solutions. For each of the 310 instances, an efficiency-maximising solution was calculated using a solving time limit of 600 seconds. Only 12 instances were not solved to optimality within this time limit. For these instances, the best solution found during the time limit was selected, which was less than 1% away from optimality in all cases.

Figure 7.3 shows three quantities for each operator: the normed aggregated utility $\hat{U}_o(\mathbf{x})$, the proportion of instances for which $\hat{U}_{o,i}(x^i) = 1$, and the number of instances $|\{i \in I : o \in O_i\}|$ in which the operator runs at least one train. This shows that NT had the smallest normed aggregated utility, followed by LNER. Other operators had higher figures, showing that there is inequality in the normed utility of operators. NT and LNER also run trains in the greatest number of instances and run the most trains

in total. It can be seen that the normed utility of NT is only equal to 1 ($\hat{U}_{o,i}(x^i) = 1$) in 41.6% of instances, which is significantly lower than for other operators.

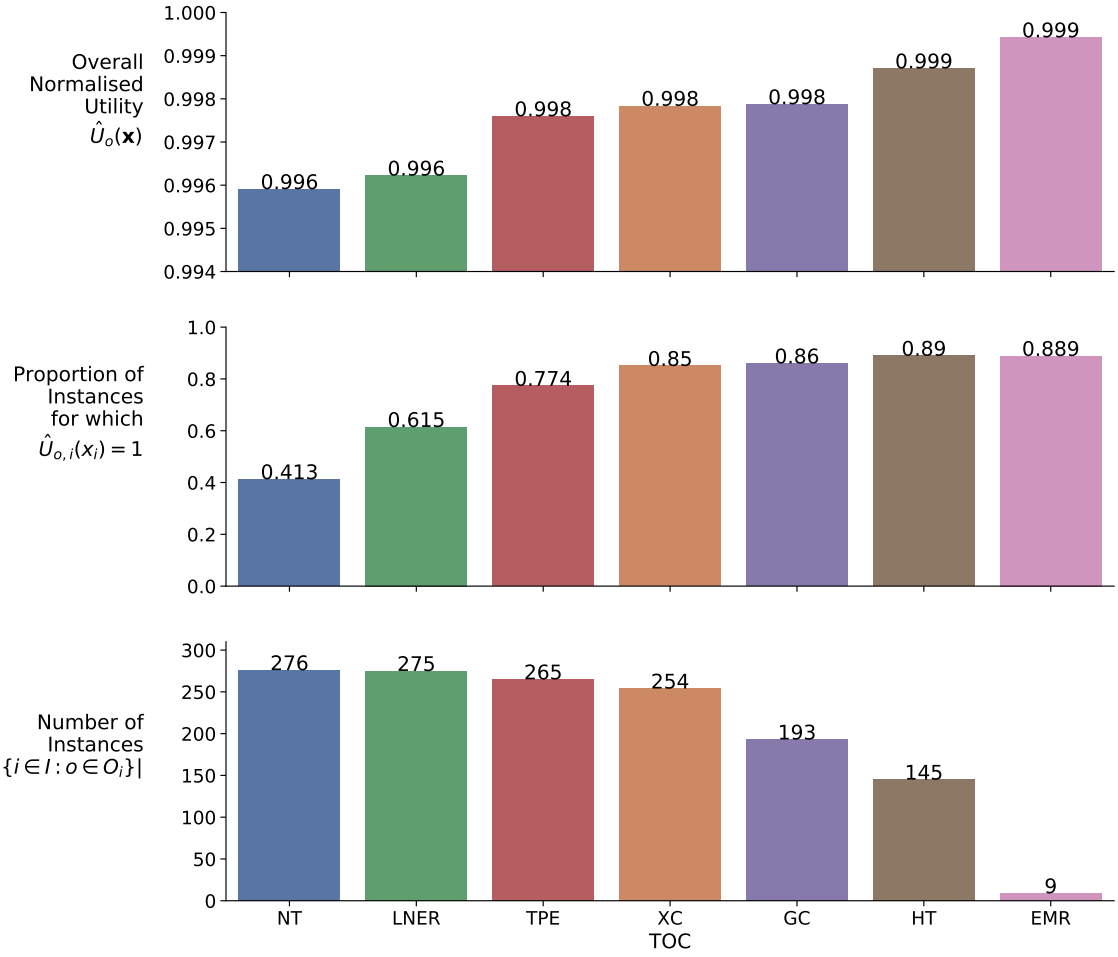


Figure 7.3: Normed aggregated utility, number of instances with utility of 1, and number of instances for each operator.

The overall figure for the α-Fairness over the full test set is -0.01689 (5 decimal places) with $\alpha = 1$. Figure 7.4 shows the α-Fairness of each instance separately. From this it can be seen that the set of solutions as a whole is more fair than many of the individual instances in the test set. It is also apparent that fairer instances are more numerous than less fair instances.

The distribution of normed utility over the instances by operator is further visible in Figure 7.5. For each operator, a boxplot is shown of $\hat{U}_{o,i}(x^i)$ for all of the instances

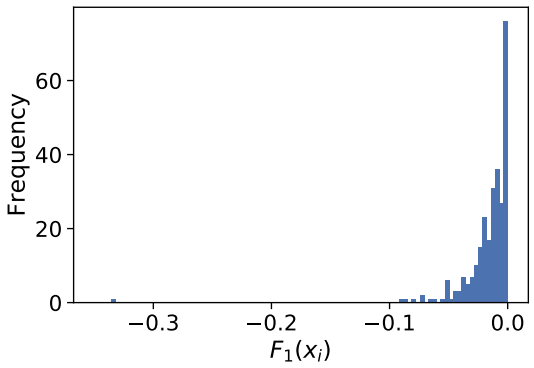


Figure 7.4: Histogram of α-fairness scores of instances, with $\alpha = 1$.

$i \in I$ such that $\hat{U}_{o,i}(x^i) \neq 1$. These are instances in which the normed operator utility was less than the maximum possible amount available for that operator. It shows that while NT was affected a greater number of instances, the values of $\hat{U}_{o,i}(x^i)$ in these instances were on average closer to 1 than for most other operators. This might be because $\hat{U}_{o,i}(x^i)$ is measured as a proportion of the total possible utility, and therefore when an operator runs more trains in an instance, a delay to any one of them causes less proportional impact.

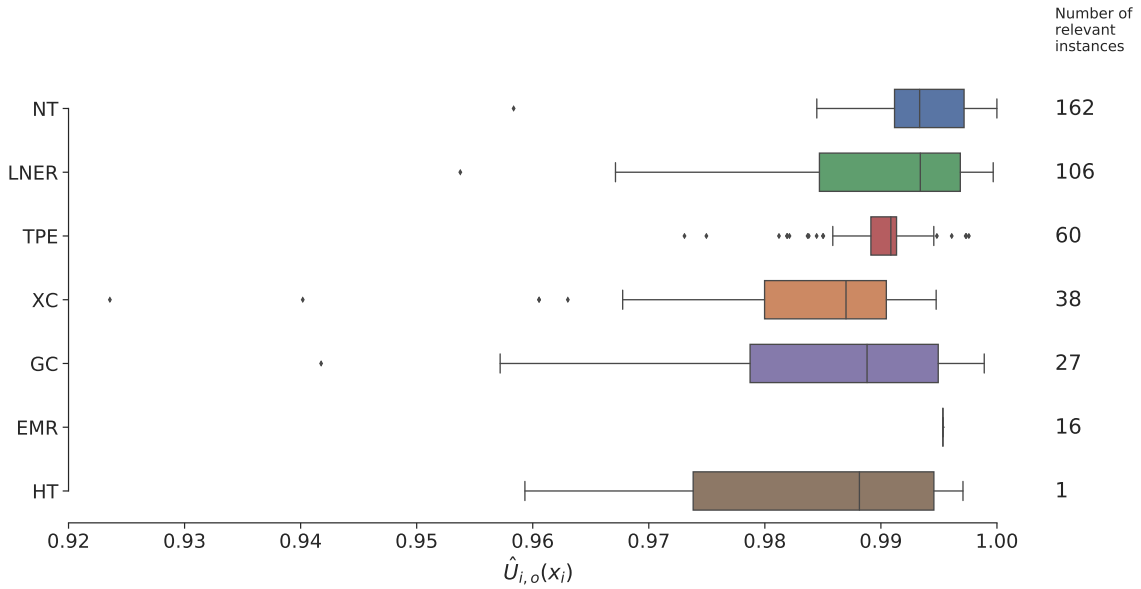


Figure 7.5: A boxplot showing the distribution of $\hat{U}_{i,o}(x_i)$ over the set of instances $i \in I$, by operator. Instances i for which $\hat{U}_{i,o}(x_i) = 1$ are excluded. The number of instances that are included is shown on the right.

To summarise, the results in this section indicate that unfairness is present in efficiency-maximal timetable rescheduling solutions. NT achieves a smaller proportion of its total possible utility than any other operator, although this is due to small losses over many instances. The results also show that fairness varies between instances.

7.6 Identifying Sources of Unfairness

Values of $\hat{U}_{o,i}(x^i)$ that are less than 1 occur because TTRP solutions must allocate scarce track capacity to some operators in preference to others. TTRP instances contain trade-offs between the interests of different operators that must be resolved according to the objective function used. We examine these trade-offs in order to more fully understand the variation in normalised aggregated utility across different operators that leads to unfairness.

For a given instance, let $x^*_{\setminus o}$ be the optimal solution when it is solved with objective function

$$\sum_{o' \in O \setminus \{o\}} U_{o'}(x), \tag{7.10}$$

for each operator $o \in O$. This objective function is obtained from the efficiency function by removing the utility accrued from any trains operated by o. For each remaining operator $o' \in O \setminus \{o\}$, the proportional gain in their utility is

$$\hat{U}_{o' \setminus o}(x) = \frac{U_{o'}(x^*_{\setminus o}) - U_{o'}(x)}{U_{o'}(x)}, \tag{7.11}$$

where x is an efficiency-maximising solution. Note that $o' \in O$ is necessary for this value to be defined, but that if $o \notin O$, we simply have that $\hat{U}_{o' \setminus o}(x) = 1$ (an operator that does not run trains in the instance is removed, so the utility of o' is

not affected).

We can aggregate this over the whole instance set I. By letting $O_i, U_{i,o}, x^i$ and $x^{i,*}_{\setminus o}$ denote O, U_o, x and $x^*_{\setminus o}$ for instance $i \in I$ (as in Section 7.4.2), the proportional gain in aggregated utility can be written as

$$\hat{U}_{o'\setminus o}(\mathbf{x}) = \frac{\sum_{i\in I: o'\in O_i} U_{i,o'}(x^{i,*}_{\setminus o}) - \sum_{i\in I: o'\in O_i} U_{i,o'}(x^i)}{\sum_{i\in I: o'\in O_i} U_{i,o'}(x^i)}. \tag{7.12}$$

These values make it possible to identify the pairwise trade-offs between operators. A large value of $\hat{U}_{o'\setminus o}(\mathbf{x})$ means that operators o and o' are in greater competition for track capacity, whereas $\hat{U}_{o'\setminus o}(\mathbf{x}) = 0$ means that there is no competition.

The values of $\hat{U}_{o'\setminus o}(\mathbf{x})$ were calculated for each pair of operators using all 310 instances. They are displayed in Figure 7.6 and represented visually in Figure 7.8 as a directed graph in which the width of each directed arc (o', o) represents the magnitude of $\hat{U}_{o'\setminus o}(\mathbf{x})$. Figure 7.7 shows the number of instances $i \in I$ for which $\hat{U}_{i,o'\setminus o}(x^i) > 0$ for each pair o, o' of operators.

Figure 7.6 shows that that LNER experienced a gain in utility of 0.2957% when NT was removed from every instance, the largest proportional increase of any pair of operators. Figure 7.7 shows that when LNER was removed from the instances, NT improved its utility in 129 of the 310 instances, the largest number of instances of any pair of operators. From Figure 7.8 it is apparent that LNER and NT both have significant trade-offs with a wide variety of different operators. Some pairs of operators that might have been expected to have significant trade-offs, because they both run large numbers of trains, did not have large trade-offs. TPE and XC are a good example of this, affecting the utility of each other only in 3 and 6 cases, respectively. This probably reflects characteristics of the timetable, in which their

trains are rarely scheduled to pass through Doncaster Station at similar times. EMR experiences no utility trade-off, but this is not a significant finding since they operate trains in only 9 of the 310 instances.

	NT	LNER	XC	TPE	GC	HT	EMR
NT	0	18.26	13.06	-0.49	0.63	0.2	0
LNER	29.57	0	4.33	5.83	3.57	1.37	0
XC	16.17	10.1	0	1.69	0.94	-0.08	0
TPE	-2.2	17.75	1.27	0	9.39	0.19	0
GC	9.58	17.89	0.73	3.66	0	0.32	0
HT	4.74	15.87	3.38	-1.66	0	0	0
EMR	5.74	0	0	0	0	0	0

Figure 7.6: The values of $\hat{U}_{o'\setminus o}(\mathbf{x})$ multiplied by 100 (therefore given in 100^{ths} of a percent) and rounded to 2 decimal places. Rows correspond to o'; columns to o.

	NT	LNER	XC	TPE	GC	HT	EMR
NT	0	129	115	12	16	3	0
LNER	91	0	37	40	34	13	0
XC	33	26	0	6	5	1	0
TPE	7	48	3	0	22	1	0
GC	12	27	2	7	0	2	0
HT	8	14	3	2	0	0	0
EMR	1	0	0	0	0	0	0

Figure 7.7: The number of instances out of 310 for which $\hat{U}_{o'\setminus o}(x^i) > 0$. Rows correspond to o'; columns to o.

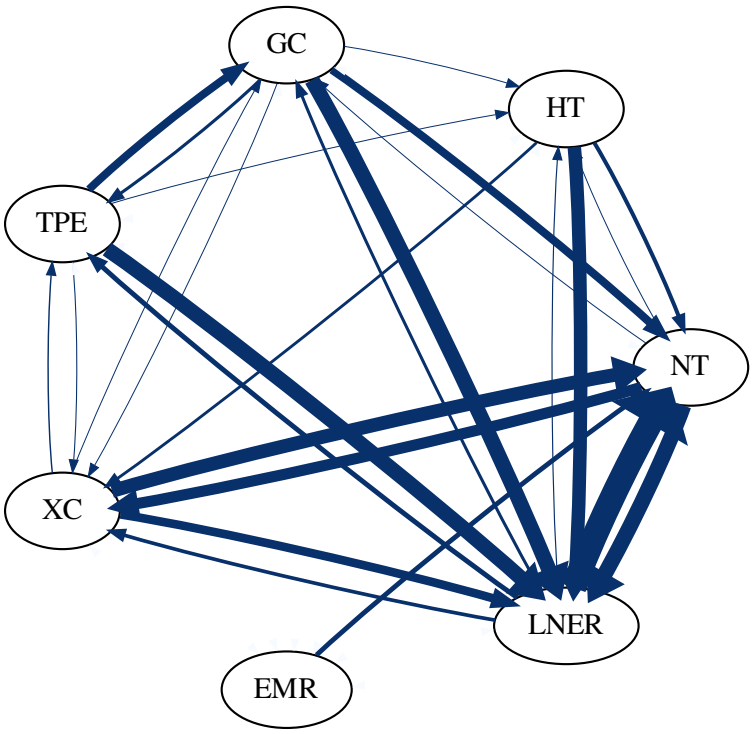


Figure 7.8: A graph of the operators in which the width of arc (o', o) corresponds to $\hat{U}_{o' \setminus o}(\mathbf{x})$.

An interesting aspect of the results displayed in Figure 7.6 is that for some pairs of operators the value of $\hat{U}_{o'\setminus o}(\mathbf{x})$ is negative, although none of these are large in absolute value. Specifically, this is observed for (o', o) =(TPE, NT), (NT, TPE), (XC, HT) and (HT, TPE). Negative trade-offs over the whole instance set are observed because in 62 of the instances, there is some pair of operators for which $\hat{U}_{i,o'\setminus o}(x^i) < 0$. This means that when operator o was removed from the instance, the utility of operator o' decreased.

This phenomenon is known as *resource non-monotonicity*, and observing it shows that maximising the efficiency is a *resource non-monotonic* strategy. In this case, the resource in question is track capacity and the entities in question are the operators in the set $O \setminus \{o\}$. When operator o is removed from an instance, the collective availability of track capacity to the entities is increased since o is no longer competing for it. Formally, removing o from an instance weakly decreases the dual price of each *time-space resource* (see Reynolds et al. [2020] for an explanation of these terms). As a result, the entities are always collectively better off, i.e.

$$\sum_{o' \in O\setminus\{o\}} U_{o'}(x^{i,*}_{\setminus o}) \geq \sum_{o' \in O\setminus\{o\}} U_{o'}(x^i).$$

However, this collective improvement in utility is not shared equally under an allocation strategy of efficiency maximisation. Rather, we observe that whilst some operators benefit disproportionately, others experience a decrease in utility. This occurs when the removal of trains run by o from an instance causes an improvement for trains run by some operator $\hat{o}$ that is only possible with a degradation for trains run by o'.

7.7 The Effect of Train Class Weight on Efficiency and Fairness

The evidence from Section 7.5 and Section 7.6 shows that in our case study, NT experiences the lowest normalised aggregated utility and that this arises as a result of trade-offs with the utility of LNER in particular. It is notable that NT also run the majority of class 2 trains in the instance set, and that these are down-weighted in the efficiency measure. Specifically, the train class weight is $w = 0.4$, meaning that the efficiency contribution of a class 2 train that runs exactly as planned is worth 0.4 of what it would if it were class 1. This suggests that an improvement in fairness might be achieved by using an objective function in which class 2 trains are down-weighted less severely.

To test this, all 310 instances were solved using the objective function U with three different values $w = 0.4, 0.7$ and 1.0. The scenario in which $w = 0.4$ is equivalent to maximising the efficiency, whereas equal weight is given to class 1 and class 2 trains in the $w = 1$ scenario. The final scenario, $w = 0.7$, was selected because it is halfway between these two extremes. For each value of w, the aggregated efficiency $E(\mathbf{x})$ and the aggregated fairness $F_\alpha(\mathbf{x})$ were calculated and are plotted in Figure 7.9. Figure 7.10 compliments the information in Figure 7.9 by showing the normalised aggregated utilities of the operators in each of the three scenarios.

These results show that as w is increased from 0.4 to 0.7, there is a trade-off between fairness and efficiency. The solutions obtained with $w = 0.7$ are more fair than $w = 0.4$, but less efficient. It is not surprising that the efficiency decreases, since when $w = 0.4$ the objective is to maximise efficiency. Figure 7.10 shows that the increase in fairness comes principally from an increase in utility for NT that is associated with a small cost for LNER and TPE. It is likely that this results in some of the conflicts between NT and LNER and between between NT and GC from being decided in

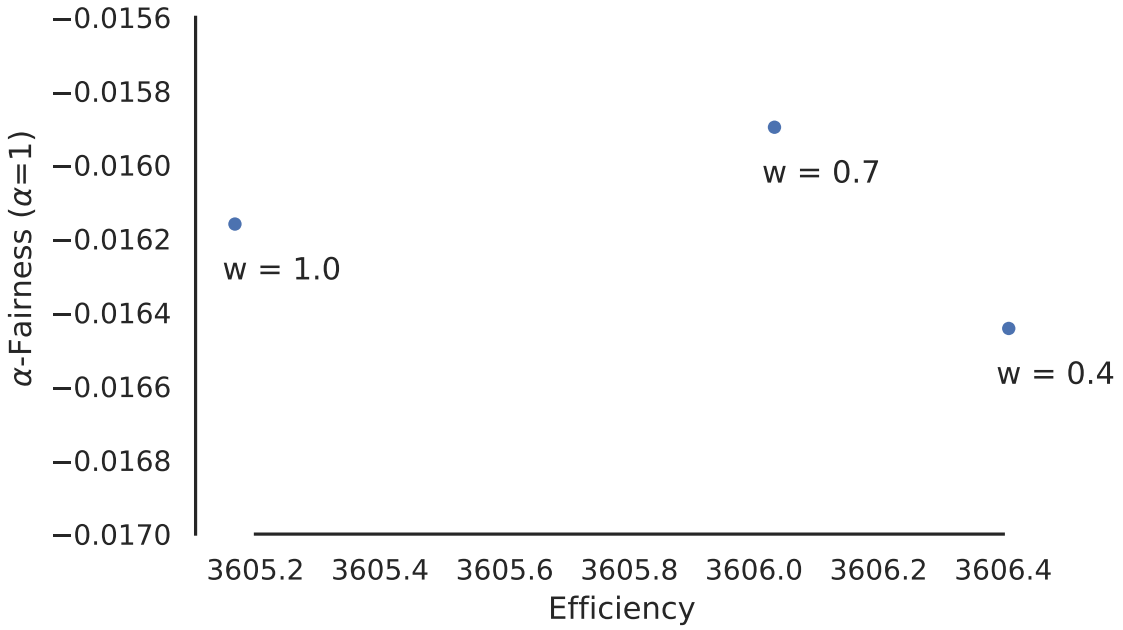


Figure 7.9: Efficiency and fairness for different values of w.

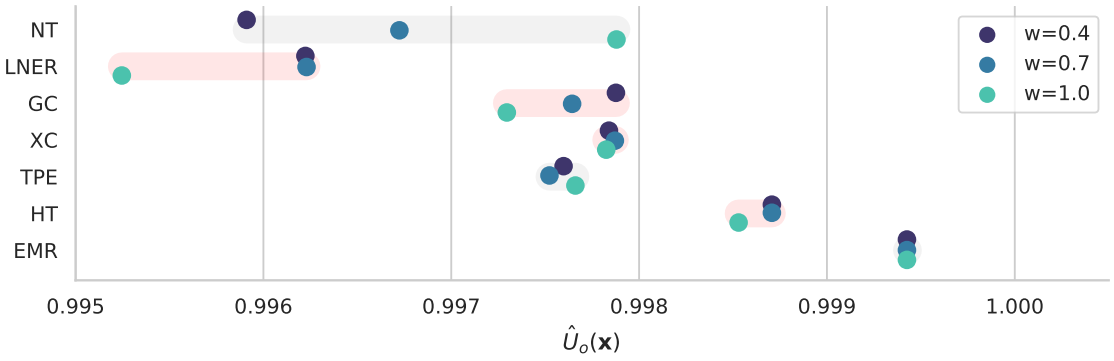


Figure 7.10: The change in normalised aggregated operator utility for different values of w. Grey lines indicate that $\hat{U}_o(\mathbf{x})$ increases as w increases; red lines show the opposite.

favour of NT as a result of the higher value of w.

A different pattern is observed as w is increased from 0.7 to 1.0. The efficiency continues to deteriorate as w deviates more from 0.4, and therefore the objective function deviates more from the efficiency measure. However, the rescheduling outcomes are less fair for $w = 1.0$ than for $w = 0.7$. Figure 7.10 shows that the increase in w once again causes a substantial improvement to the utility of NT. However, this time a more significant decrease in the utilities of LNER and GC are observed. Since for $w = 0.7$ LNER already has a lower normed aggregated utility than NT, this results in a decrease in fairness. The point for $w = 1.0$ is dominated by the point for $w = 0.7$, since it has both lower fairness and lower efficiency. This highlights that the points depicted in Figure 7.9 are not (necessarily) Pareto optimal — bi-objective optimisation techniques would be required to find Pareto optimal points. However, as pointed out in Section 7.4.2, it isn't possible to optimise fairness since this is measured over the whole instance set.

Figure 7.10 shows that the largest changes in utility in response to changing w are experienced by NT, LNER and GC. These findings are consistent with the evidence from Section 7.6 that competition between NT and LNER is high. It is interesting to note that XC does not experience a significant fall in normalised aggregated utility, despite the fact that competition between NT and XC was also found to be high. This could indicate the importance of factors other than train weight in deciding how conflicts involving trains operated by NT and XC are resolved. Another possible explanation is that conflicts involving these two operators primarily relate to the small number of Class 1 trains operated by NT.

7.8 Conclusions

In this paper, we evaluate the fairness of a TTRP algorithm in the presence of competition between train operators. This is achieved by analysing results from a case study of Doncaster station during the month of January 2017. Measures for both efficiency and fairness are proposed and calculated both for each instance individually and over the whole month. Efficiency-maximising solutions are studied both to investigate fairness and to analyse pairwise trade-offs between operators. Finally, the effect of changing the train class weighting in the objective function is studied with respect to fairness and efficiency.

The results show that some unfairness is present in efficiency-maximising solutions. That unfairness is principally to the detriment of NT, the operator running most of the class 2 trains. Further, the largest trade-off in normalised aggregated utility is between NT and LNER. Increasing the train class weighting from 0.4 to 0.7 increases fairness by improving the utility of NT, but decreases efficiency by harming the utility of operators running class 1 trains. However, increasing w from 0.7 to 1.0 decreases both efficiency and fairness by harming the utility of LNER. This shows that the value of w needs to be carefully considered in any future deployment of optimisation-based timetable rescheduling.

Future research should focus on making direct comparisons between the fairness of TTRP solutions and the fairness of historical manually decided rescheduling actions. This would add to the evidence presented in this paper about the likely impact of deploying optimisation-based timetable rescheduling. Another direction of further study would be to identify how elements of the objective function other than w affect fairness. For example, it would be interesting to understand whether optimising the fairness of each individual instance leads to greater fairness over the whole instance set, and what the effect on efficiency would be. Finally, it would be interesting to

understand how operator fairness and passenger fairness interact, and how fairness for the TTRP ought to relate to the fairness of the original timetable construction process.

Chapter 8

Conclusions

The aim of this thesis was given in Chapter 1:

> To develop new optimisation modelling, solution and evaluation techniques for the TTRP that can help overcome the barriers to deployment that exist for optimisation-based rescheduling.

The research presented in Chapters 5, 6 and 7 fulfils this aim. Each of these chapters identifies and addresses a distinct barrier to deployment for optimisation-based timetable rescheduling that arises from inadequacies in the existing TTRP literature.

8.1 Contributions

In Chapter 5, we identified that only a few existing models both perform rerouting and represent route-release signalling systems adequately. Of those that do, none are suitable for solving realistic instances from complex station areas to optimality

in sufficiently short times. The ability to perform rerouting is crucial in complex station areas because rerouting trains can help to avoid conflicts. Modelling route-release signalling is essential for deriving solutions that can be realised in practice. Moreover, the ability to produce provably optimal or near-optimal solutions in the time available for timetable rescheduling is a key attraction of using optimisation technology.

This problem was addressed by proposing a new multicommodity flow model for the TTRP that both performs rerouting and models route-release signalling. The latter is achieved using a novel method for representing track capacity constraints that doesn't require variables for every track circuit. A tailored branch-and-price algorithm was also developed that allows realistic instances to be solved to near-optimality in sufficiently short times. This was demonstrated in a computational study involving a new set of instances for Doncaster station.

The research in Chapter 5 makes three additional contributions. First, a new objective function was developed alongside Network Rail to model their utility with respect to timetable rescheduling. Second, a relationship was demonstrated between the number of conflicts in an instance and the difficulty of solving it. Finally, it has been demonstrated that time-indexed formulations can be viable for real-time applications.

In Chapter 6, we identified that existing methods of modelling of speed profiles within TTRP models are unsatisfactory. Whilst fixed-speed models fail to capture the effects of train speed profiles, the variable-speed models that have been proposed all have shortcomings. Adding a constant amount of time when a train stops is too simplistic and iterative methods are too time consuming. Modelling speed profiles exactly using detailed physics-based modelling can have prohibitive data requirements, and can result in overly complex models that would be challenging to deploy in practice.

This problem was addressed by developing a new way of approximately modelling speed profiles using discrete speed profile types. The model from Chapter 5 was extended to a variable-speed model based on a time-space-type graph. The extension makes the model capable of taking into account the effects of speed profiles on traversal times. Our data-driven approach uses methods from statistics to ensure that additional complexity is only introduced to the model where it can be best justified, resulting in a parsimonious model that would be simple to deploy. It was tested on a new set of instances from Derby station. The results show that the model is able to capture the benefits of variable-speed modelling without any increase in solution times compared with the fixed-speed model.

In Chapter 7, we identified an absence of evidence in the TTRP literature concerning fairness between operators in competitive railway systems. This is likely to be an important consideration for deployment of TTRP technology.

To address this problem, we carried out experiments using the model from Chapter 5 to evaluate the fairness of TTRP solutions for our case study in Doncaster. Definitions of efficiency and fairness for the TTRP were proposed for the first time. We then presented evidence that efficiency-maximising solutions lead to unfair outcomes and explored the reasons for this by looking at conflicts between operators. Finally, we showed that the fairness of efficiency maximising solutions can be increased up to a point by changing the relative weighting of class 2 trains. Our study should inspire confidence that the fairness of TTRP solutions can be both rigorously evaluated and managed.

8.2 Further Research

The research in this thesis has uncovered more opportunities to improve upon and add to the existing TTRP research literature.

8.2.1 Modelling Larger Areas

Perhaps the most important future modelling challenge for TTRP researchers is the question of how to model large geographical areas of railway in microscopic detail such that real instances are tractable. Reactionary delay can propagate across entire countries in a relatively short space of time, and therefore handling ever larger areas will allow rescheduling responses to be more effective. Models based on macroscopic network topologies can already handle large areas, but these generally ignore signalling detail that can be very important to consider, especially in complex station areas. Microscopic models can certainly be used to model larger areas, but these will not become tractable without a very significant improvement in solution methods.

A solution to this challenge is to maintain a number of microscopic models for smaller areas, and to link these models together so that rescheduling decisions in each area are coordinated. However, there is no consensus about how the models should be linked. Corman et al. [2012b] have suggested an approach in which a bilevel centralised coordination problem is solved in order to impose constraints on trains at the borders of local areas. These constraints must be satisfied by solutions produced by local TTRP models. Lamorgese et al. [2016] have shown that in the case of a line linking multiple stations, Benders' decomposition can be an effective strategy. Each subproblem is a station rescheduling problem and the master problem links stations along a line. Toletti et al. [2020] have proposed a Lagrangian decomposition

framework in which each subproblem is a local rescheduling problem. Constraints requiring consistency at the borders of the local areas are associated with Lagranian multipliers that are optimised in an iterative procedure. Although promising, each of these approaches is currently relatively unexplored.

A different approach to the challenge of modelling large areas microscopically is to limit models to consider only the spatial and temporal region most affected by a disruption. Van Thielen et al. [2018], who call this spatio-temporal region a *dynamic impact zone*, have shown that such a model can handle a large area when solved heuristically. Rezanova and Ryan [2010] use a similar idea for train driver recovery, limiting the problem to consider only a subset of related driver duties. An important outstanding challenge in this area is to find a fast method for determining a dynamic impact zone that allows good quality rescheduling solutions to be obtained. If this could be done whilst maintaining guarantees on the optimality of any solutions produced, then that would be a major step forward.

8.2.2 Aggregation based Solution Methods

There are a variety of potential opportunities for developing new exact solution methods for the model in Chapter 5 that are based around the idea of aggregation. This is the idea that if multiple nodes of the time-space graph can be aggregated into a single node without affecting the optimal solution to the problem, then it is preferable to solve the smaller time-space graph. Since good aggregations are not known in advance, they are altered dynamically during the solution process.

One such technique is Dynamic Constraint Aggregation [Elhallaoui et al., 2005]. This is a column generation acceleration strategy that dynamically aggregates and disaggregates constraints (which correspond to nodes in the time-space graph). Another possible algorithmic framework is Dynamic Discretization Discovery [Boland

and Savelsbergh, 2019]. This focuses on the granularity of the time discretization, starting with large time intervals and dynamically subdividing them where necessary. In a similar vein, Fischer and Helmberg [2014] propose a Dynamic Graph Generation algorithm that constructs only parts of the time-space graph that are necessary for solving subproblems. Whilst Fischer and Helmberg [2014] tackle a train timetabling application, to the best of our knowledge none of these methods have yet been applied to the TTRP.

8.2.3 Simulation and Benchmarking for Model Evaluation

A dearth of high quality evaluations of TTRP optimisation models is perhaps the largest remaining barrier to their deployment. Progress in this area would help to build the business case for deployment, and identify the risks involved.

Whilst almost every research paper on this topic reports solution times and evaluation of solution algorithms, the quality of the solutions produced is generally poorly investigated. It must be shown that the solutions produced by models can be implemented in practice and lead to the expected outcomes. Real implementations are rare (Lamorgese et al. [2018] describe some of the few that have taken place), and are not usually designed as experiments. Simulation tools are therefore indispensable for TTRP model evaluation. Models should ideally be tested in dynamic, closed loop control environments, in which they repeatedly feed solutions to a simulator. Fortunately, microscopic railway simulations that model detailed speed profiles, all signalling constraints and uncertainty are widely used in the railway industry, especially for timetable evaluation. These are reviewed by Medeossi and de Fabris [2018].

A particular challenge is that most models have been tested on different sets of instances, making it almost impossible to make conclusive comparisons. This is to some

extent necessary, since different models are designed for different types of railways, perform different rescheduling actions, and take into account different additional constraints. However, most of the time it is due to difficulties in sharing data and instances. A TTRP instance consists of infrastructure data (which is often hardest to obtain and share), timetable data and data about traffic perturbations. Aside from commercial confidentiality considerations, there is a lack of standardisation of data formats both in the UK and internationally. A common set of diverse TTRP instances in a standardised format would allow models to be benchmarked against each other, both in terms of algorithmic performance and solution quality. This would give railway companies confidence to take up the best performing models.

Bibliography

Acuna-Agost, R., Michelon, P., Feillet, D., and Gueye, S. (2011a). A MIP-based Local Search Method for the Railway Rescheduling Problem. *Networks*, 57(1):69–86.

Acuna-Agost, R., Michelon, P., Feillet, D., and Gueye, S. (2011b). SAPI: Statistical Analysis of Propagation of Incidents. A new approach for rescheduling trains after disruptions. *European Journal of Operational Research*, 215(1):227–243.

Ahuja, R., Magnant, T., and Orlin, J. (1993). *Network Flows*. Prentice Hall.

Altazin, E., Dauzère-Pérès, S., Ramond, F., and Tréfond, S. (2020). A multi-objective optimization-simulation approach for real time rescheduling in dense railway systems. *European Journal of Operational Research*, 286:662–672.

Aronson, J. E. (1989). A survey of dynamic network flows. *Annals of Operations Research*, 20:1–66.

Atkinson, A. B. (1970). On the Measurement of Inequality. *Journal of Economic Theory*, 2:244–263.

Barnhart, C., Hane, C. A., and Vance, P. H. (2000). Using Branch-and-Price-and-Cut to Solve Origin-Destination Integer Multicommodity Flow Problems. *Operations Research*, 48(2):318–326.

Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M., and Vance, P. H. (2001). Branch-and-Price: Column Generation for Solving Huge Integer Programs. *Operations Research*, 46(3):316–329.

Bergson, A. (1938). A reformulation of certain aspects of welfare economics. *Quarterly Journal of Economics*, 52(2):310–334.

Bertsimas, D., Farias, V. F., and Trichakis, N. (2011). The price of fairness. *Operations Research*, 59(1):17–31.

Bertsimas, D., Farias, V. F., and Trichakis, N. (2012). On the Efficiency-Fairness Trade-off. *Management Science*, 58(2020):2234–2250.

Bertsimas, D. and Gupta, S. (2016). Fairness and collaboration in network air traffic flow management: An optimization approach. *Transportation Science*, 50(1):57–76.

Bešinović, N., Quaglietta, E., and Goverde, R. M. (2013). A simulation-based optimization approach for the calibration of dynamic train speed profiles. *Journal of Rail Transport Planning and Management*, 3(4):126–136.

Bettinelli, A., Santini, A., and Vigo, D. (2017). A real-time conflict solution algorithm for the train rescheduling problem. *Transportation Research Part B: Methodological*, 106:237–265.

Binder, S., Maknoon, Y., and Bierlaire, M. (2017). The multi-objective railway timetable rescheduling problem. *Transportation Research Part C: Emerging Technologies*, 78:78–94.

Boland, N. and Savelsbergh, M. (2019). Perspectives on integer programming for time-dependent models. *Top*, 27:147–173.

Borndörfer, R., Klug, T., Lamorgese, L., Mannino, C., Reuther, M., and Schlechte, T. (2017). Recent success stories on integrated optimization of railway. *Transportation Research Part C*, 74:196–211.

Borndörfer, R., Klug, T., Lamorgese, L., Mannino, C., Reuther, M., and Schlechte, T., editors (2018). *Handbook of Optimization in the Railway Industry*. International Series in Operations Research & Management Science. Springer.

Borndörfer, R. and Schlechte, T. (2007). Models for railway track allocation. In *7th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems*.

Bouveyron, C., Celeux, G., Murphy, T. B., and Raftery, A. E. (2019). Model-Based Clustering and Classification for Data Science: With Applications in R. In *Model-based Clustering and Classification for Data Science*, pages 15–75. Cambridge University Press.

Brännlund, U., Lindberg, P. O., Nou, A., and Nilsson, J.-E. (1998). Railway Timetabling Using Lagrangian Relaxation. *Transportation Science*, 32(4):358–369.

Breugem, T., Schulz, C., Schlechte, T., and Borndörfer, R. (2019). A Three-Phase Heuristic for Cyclic Crew Rostering with Fairness Requirements. Technical report, Zuse Institute Berlin.

Brünger, O. and Dahlhaus, E. (2014). Running Time Estimation. In Hansen, I. A. and Pachl, J., editors, *Railway Timetabling & Operations*, pages 65–90. Eurailpress, second edition.

Cacchiani, V., Caprara, A., and Toth, P. (2008). A column generation approach to train timetabling on a corridor. *4OR*, 6(2):125–142.

Cacchiani, V., Galli, L., and Toth, P. (2015). A tutorial on non-periodic train timetabling and platforming problems. *EURO Journal on Transportation and Logistics*, 4(3):285–320.

Cacchiani, V., Huisman, D., Kidd, M. P., Kroon, L. G., Toth, P., Veelenturf, L. P., and Wagenaar, J. C. (2014). An overview of recovery models and algorithms for real-time railway rescheduling. *Transportation Research Part B: Methodological*, 63:15–37.

Caimi, G., Chudak, F., Fuchsberger, M., Laumanns, M., and Zenklusen, R. (2011). A New Resource-Constrained Multicommodity Flow Model for Conflict-Free Train Routing and Scheduling. *Transportation Science*, 45(2):212–227.

Caimi, G., Fuchsberger, M., Laumanns, M., and Lüthi, M. (2012). A model predictive control approach for discrete-time rescheduling in complex central railway station areas. *Computers and Operations Research*, 39(11):2578–2593.

Caprara, A., Fischetti, M., and Toth, P. (2002). Modeling and Solving the Train Timetabling Problem. *Operations Research*, 50(5):851–861.

Caprara, A., Galli, L., and Toth, P. (2011). Solution of the Train Platforming Problem. *Transportation Science*, 45(2):246–257.

Caprara, A., Kroon, L. G., and Toth, P. (2010). Optimization Problems in Passenger Railway Systems. *Wiley Encyclopedia of Operations Research and Management Science*, pages 3896–3905.

Cavone, G., Dotoli, M., Epicoco, N., and Seatzu, C. (2017). A decision making procedure for robust train rescheduling based on mixed integer linear programming and Data Envelopment Analysis. *Applied Mathematical Modelling*, 52:255–273.

Chen, L., Schmid, F., Dasigi, M., Ning, B., Roberts, C., and Tang, T. (2010). Real-time train rescheduling in junction areas. *Proceedings of the Institution of Mechanical Engineers Part F-Journal of Rail and Rapid Transit*, 224(F6):547–557.

Cordeau, J.-F., Toth, P., and Vigo, D. (1998). A Survey of Optimization Models for Train Routing and Scheduling. *Transportation Science*, 32(4):380–404.

Corman, F., D'Ariano, A., Hansen, I. A., and Pacciarelli, D. (2011a). Optimal multi-class rescheduling of railway traffic. *Journal of Rail Transport Planning and Management*, 1(1):14–24.

Corman, F., D'Ariano, A., Marra, A. D., Pacciarelli, D., and Samà, M. (2016). Integrating train scheduling and delay management in real-time railway traffic control. *Transportation Research Part E: Logistics and Transportation Review*.

Corman, F., D'Ariano, A., Pacciarelli, D., and Pranzo, M. (2009a). Evaluation of green wave policy in real-time railway traffic management. *Transportation Research Part C*, 17(6):607–616.

Corman, F., D'Ariano, A., Pacciarelli, D., and Pranzo, M. (2010a). A tabu search algorithm for rerouting trains during rail operations. *Transportation Research Part B*, 44(1):175–192.

Corman, F., D'Ariano, A., Pacciarelli, D., and Pranzo, M. (2010b). Centralized versus distributed systems to reschedule trains in two dispatching areas. *Public Transport*, 2(3):219–247.

Corman, F., D'Ariano, A., Pacciarelli, D., and Pranzo, M. (2012a). Bi-objective conflict detection and resolution in railway traffic management. *Transportation Research Part C*, 20(1):79–94.

Corman, F., D'Ariano, A., Pacciarelli, D., and Pranzo, M. (2012b). Optimal inter-area coordination of train rescheduling decisions. *Transportation Research Part E*, 48(1):71–88.

Corman, F., D'Ariano, A., Pranzo, M., and Hansen, I. A. (2011b). Effectiveness of dynamic reordering and rerouting of trains in a complicated and densely occupied station area. *Transportation Planning and Technology*, 34(4):341–362.

Corman, F., Goverde, R. M. P., and D'Ariano, A. (2009b). Rescheduling Dense Train Traffic over Complex Station Interlocking Areas. In Ahuja, R., Möring, R., and Zaroliagis, C., editors, *Robust and Online Large-Scale Optimization*, pages 369–386. Springer.

Corman, F. and Meng, L. (2015). A review of online dynamic models and algorithms for railway traffic control. *IEEE Transactions on Intelligent Transportation Systems*, 16(3):1274–1284.

Dantzig, G. B. and Wolfe, P. (1960). Decomposition Principle for Linear Programs. *Operations Research*, 8(1):101–111.

D'Ariano, A., Corman, F., Pacciarelli, D., and Pranzo, M. (2008). Reordering and Local Rerouting Strategies to Manage Train Traffic in Real Time. *Transportation Science*, 42(4):405–419.

D'Ariano, A., Pacciarelli, D., and Pranzo, M. (2007a). A branch and bound algorithm for scheduling trains in a railway network. *European Journal of Operational Research*, 183(2):643–657.

D'Ariano, A. and Pranzo, M. (2009). An Advanced Real-Time Train Dispatching System for Minimizing the Propagation of Delays in a Dispatching Area Under Severe Disturbances. *Networks and Spatial Economics*, 9(1):63–84.

D'Ariano, A., Pranzo, M., and Hansen, I. A. (2007b). Conflict Resolution and Train Speed Coordination for Solving Real-Time Timetable Perturbations. *IEEE Transactions on Intelligent Transportation Systems*, 8(2):208–222.

D'Ariano, A., Samà, M., D'Ariano, P., and Pacciarelli, D. (2014). Evaluating the applicability of advanced techniques for practical real-time train scheduling. *Transportation Research Procedia*, 3:279–288.

Department for Transport (2019). Government Rail Factsheet https://www.gov.uk/government/statistics/rail-factsheet-2019.

Desaulniers, G., Desrosiers, J., and Solomon, M. M. (2002). Accelerating strategies in column generation methods for vehicle routing and crew scheduling problems. In *Essays and Surveys in Metaheuristics*, pages 309–324. Springer US.

Desrosiers, J. and Lübbecke, M. E. (2005). A primer in column generation. In Desaulniers, G., Desrosiers, J., and Solomon, M. M., editors, *Column Generation*, pages 1–32. Springer.

Dollevoet, T., Corman, F., D'Ariano, A., and Huisman, D. (2014). An iterative optimization framework for delay management and train scheduling. *Flexible Services and Manufacturing Journal*, 26(4):490–515.

Dollevoet, T., Huisman, D., Schmidt, M., and Schöbel, A. (2012). Delay Management with Rerouting of Passengers. *Transportation Science*, 46(1):74–89.

Dollevoet, T., Schmidt, M., and Schöbel, A. (2015). Delay Management including Capacities of Stations. *Transportation Science*, 49(May 2015):185–203.

Elhallaoui, I., Villeneuve, D., Soumis, F., and Desaulniers, G. (2005). Dynamic Aggregation of Set-Partitioning Constraints in Column Generation. *Operations Research*, 53(4):632–645.

Fairbrother, J., Zografos, K. G., and Glazebrook, K. D. (2020). A Slot-scheduling mechanism at congested airports that incorporates efficiency, fairness, and airline preferences. *Transportation Science*, 54(1):115–138.

Fang, W., Yang, S., and Yao, X. (2015). A Survey on Problem Models and Solution Approaches to Rescheduling in Railway Networks. *IEEE Transactions on Intelligent Transportation Systems*, 16(6):2997–3016.

Fischer, F. and Helmberg, C. (2014). Dynamic graph generation for the shortest path problem in time expanded networks. *Mathematical Programming*, 143:257–297.

Foster, J. and Sen, A. (1997). *On Economic Inequality*. Oxford University Press.

Geoffrion, A. (1974). Lagrangean relaxation for integer programming. *Mathematical Programming Study*, 2:82–114.

Gestrelius, S., Peterson, A., and Aronsson, M. (2020). Timetable quality from the perspective of a railway infrastructure manager in a deregulated market: An interview study with Swedish practitioners. *Journal of Rail Transport Planning and Management*, 15:100202.

Ginkel, A. and Schöbel, A. (2007). To wait or not to wait? The bicriteria delay management problem in public transportation. *Transportation Science*, 41(4):527–538.

Gleixner, A., Maher, S. J., Fischer, T., Gally, T., Gamrath, G., Gottwald, R. L., Hendel, G., Koch, T., Lübbecke, M. E., Miltenberger, M., Müller, B., Pfetsch, M. E., Puchert, C., Rehfeldt, D., Schenker, S., Schwarz, R., Serrano, F., Shinano, Y., Weninger, D., Witt, J. T., and Witzig, J. (2018). The SCIP Optimization Suite 6.0. Technical report, ZIB.

Gurobi Optimization LLC (2020). Gurobi Optimizer Reference Manual, http://www.gurobi.com.

Haahr, J. T., Wagenaar, J. C., Veelenturf, L. P., and Kroon, L. G. (2016). A comparison of two exact methods for passenger railway rolling stock (re)scheduling. *Transportation Research Part E: Logistics and Transportation Review*, 91:15–32.

Hansen, I. and Pachl, J., editors (2014). *Railway Timetabling & Operations*. Eurailpress, 2nd edition.

Harrod, S. (2011). Modeling Network Transition Constraints with Hypergraphs. *Transportation Science*, 45(1):81–97.

Harrod, S. (2012). A tutorial on fundamental model structures for railway timetable optimization. *Surveys in Operations Research and Management Science*, 17(2):85–96.

Hartzheim, E. (2005). *Ordered Sets*. Springer.

Hosteins, P., Pellegrini, P., and Rodriguez, J. (2019). Studies on the validity of the fixed-speed approximation for the real time Railway Traffic Management Problem. In *8th International Conference on Railway Operations Modelling and Analysis - RailNorrköping 2019*, pages 409–424.

Huisman, D. (2007). A column generation approach for the rail crew re-scheduling problem. *European Journal of Operational Research*, 180(1):163–173.

IBM (2011). Rail Liberalisation Index. http://www.assorail.fr/wp-content/uploads/2015/08/Rail_Liberalisation_Index_2011.pdf.

Irnich, S., Desaulniers, G., Desrosiers, J., and Hadjar, A. (2010). Path-reduced costs for eliminating arcs in routing and scheduling. *INFORMS Journal on Computing*, 22(2):297–313.

Josyula, S. P. (2019). *Parallel Algorithms for real-time railway rescheduling*. PhD thesis, Blekinge Institute of Technology, Department of Computer Science.

Jütte, S., Müller, D., and Thonemann, U. W. (2017). Optimizing railway crew schedules with fairness preferences. *Journal of Scheduling*, 20(1):43–55.

Kahn, A. B. (1962). Topological sorting of large networks. *Communications of the ACM*, 5(11):558–562.

Kalai, E. and Smorodinsky, M. (1975). Other Solutions to Nash's Bargaining Problem. *Econometrica*, 43(3):513–518.

Kecman, P., Corman, F., D'Ariano, A., and Goverde, R. M. P. (2013). Rescheduling models for railway traffic management in large-scale networks. *Public Transport*, 5(1-2):95–123.

Lamorgese, L. and Mannino, C. (2015). An Exact Decomposition Approach for the Real-Time Train Dispatching Problem. *Operations Research*, 63(1):48–64.

Lamorgese, L. and Mannino, C. (2019). A non-compact formulation for job-shop scheduling problems in traffic management. *Operations Research*, 67(6):1586–1609.

Lamorgese, L., Mannino, C., Pacciarelli, D., and Krasemann, J. T. (2018). Train Dispatching. In Borndörfer, R., Klug, T., Lamorgese, L., Mannino, C., Reuther, M., and Schlechte, T., editors, *Handbook of Optimization in the Railway Industry*, pages 265–283. Springer.

Lamorgese, L., Mannino, C., and Piacentini, M. (2016). Optimal Train Dispatching by Benders' - Like Reformulation. *Transportation Science*, 50(3):910–925.

Land, A. and Doig, A. (1960). An Automatic Method of Solving Discrete Programming Problems. *Econometrica,*, 28(3):497–520.

Li, D., Zhang, T., Dong, X., Yin, Y., and Cao, J. (2019). Trade-off between efficiency and fairness in timetabling on a single urban rail transit line under time-dependent demand condition. *Transportmetrica B*, 7(1):1203–1231.

Luan, X., Wang, Y., De Schutter, B., Meng, L., Lodewijks, G., and Corman, F. (2018a). Integration of real-time traffic management and train control for rail networks - Part 1: Optimization problems and solution approaches. *Transportation Research Part B: Methodological*, 115:41–71.

Luan, X., Wang, Y., De Schutter, B., Meng, L., Lodewijks, G., and Corman, F. (2018b). Integration of real-time traffic management and train control for rail networks - Part 2: Extensions towards energy-efficient train operations. *Transportation Research Part B: Methodological*, 115:72–94.

Lusby, R. M., Haahr, J. T., Larsen, J., and Pisinger, D. (2017). A Branch-and-Price algorithm for railway rolling stock rescheduling. *Transportation Research Part B: Methodological*, 99:228–250.

Lusby, R. M., Larsen, J., and Bull, S. (2018). A survey on robustness in railway planning. *European Journal of Operational Research*, 266(1):1–15.

Lusby, R. M., Larsen, J., Ehrgott, M., and Ryan, D. M. (2011). Railway track allocation: Models and methods. *OR Spectrum*, 33(4):843–883.

Lusby, R. M., Larsen, J., Ehrgott, M., and Ryan, D. M. (2013). A set packing inspired method for real-time junction train routing. *Computers and Operations Research*, 40(3):713–724.

Mannino, C. and Mascis, A. (2009). Optimal Real-Time Traffic Control in Metro Stations. *Operations Research*, 57(4):1026–1039.

Mascis, A. and Pacciarelli, D. (2002). Job-shop scheduling with blocking and no-wait constraints. *European Journal of Operational Research*, 143(3):498–517.

Mazzarello, M. and Ottaviani, E. (2007). A traffic management system for real-time traffic optimisation in railways. *Transportation Research Part B: Methodological*, 41(2):246–274.

Medeossi, G. and de Fabris, S. (2018). Simulation of Rail Operations. In Borndörfer, R., Klug, T., Lamorgese, L., Mannino, C., Reuther, M., and Schlechte, T., editors, *Handbook of Optimization in the Railway Industry*, pages 1–24. International Series in Operations Research & Management Science. Springer.

Meng, L. and Zhou, X. (2014). Simultaneous train rerouting and rescheduling on an N -track network : A model reformulation with network-based cumulative flow variables. *Transportation Research Part B*, 67:208–234.

Min, Y. H., Park, M. J., Hong, S. P. S. H., and Hong, S. P. S. H. (2011). An appraisal of a column-generation-based algorithm for centralized train-conflict resolution on a metropolitan railway network. *Transportation Research Part B: Methodological*, 45(2):409–429.

Mood, A. M. (1950). *Introduction to the Theory of Statistics*. McGraw-Hill.

Nash, J. F. (1950). The Bargaining Problem. *Econometrica*, 18(2):155–162.

National Audit Office (2008). Reducing passenger rail delays by better management of incidents. Retrieved from https://www.nao.org.uk/wp-content/uploads/2008/03/0708308.pdf. Accessed on 17/04/20.

Network Rail (2019a). Network Rail Limited Annual Report and Accounts 2019: Strategic report.

Network Rail (2019b). Technical overview : Payments relating to disruption https://www.networkrail.co.uk/wp-content/uploads/2019/03/Technical-overview-payments-relating-to-disruption.pdf.

Network Rail (2020). Railway Performance. Retrieved from https://www.networkrail.co.uk/who-we-are/how-we-work/performance/railway-performance/. Accessed 17/04/20.

Nielsen, L. K., Kroon, L. G., and Maróti, G. (2012). A rolling horizon approach for disruption management of railway rolling stock. *European Journal of Operational Research*, 220(2):496–509.

Pachl, J. (2014). Timetable Design Principles. In Hansen, I. A. and Pachl, J., editors, *Railway Timetabling & Operations*, pages 9–42. Eurailpress, 2nd edition.

Padberg, M. (1973). On the facial structure of set packing polyhedra. *Mathematical Programming*, 5:199–215.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Pellegrini, P., Marlière, G., Pesenti, R., and Rodriguez, J. (2015). RECIFE-MILP: An Effective MILP-Based Heuristic for the Real-Time Railway Traffic Management Problem. *IEEE Transactions on Intelligent Transportation Systems*, 16(5):2609–2619.

Pellegrini, P., Marlière, G., and Rodriguez, J. (2014). Optimal train routing and scheduling for managing traffic perturbations in complex junctions. *Transportation Research Part B: Methodological*, 59:58–80.

Potthoff, D., Huisman, D., and Desaulniers, G. (2010). Column generation with dynamic duty selection for Railway Crew Rescheduling. *Transportation Science*, 44(4):493–505.

Radtke, A. (2014). Infrastructure Modelling. In Hansen, I. A. and Pachl, J., editors, *Railway Timetabling & Operations*, chapter 3. Eurailpress, 2nd edition.

Reynolds, E. (2020). TTRP Full Results. www.github.com/edwin6/TTRP_Results.

Reynolds, E., Ehrgott, M., Maher, S. J., Patman, A., and Wang, J. Y. T. (2020). A multicommodity flow model for rerouting and retiming trains in real-time to reduce reactionary delay in complex station areas. *(submitted)*.

Rezanova, N. J. and Ryan, D. M. (2010). The train driver recovery problem-A set partitioning based model and solution method. *Computers and Operations Research*, 37(5):845–856.

Rodriguez, J. (2007). A constraint programming model for real-time train scheduling at junctions. *Transportation Research Part B: Methodological*, 41(2):231–245.

Ryan, D. M. and Foster, B. A. (1981). An integer programming approach to scheduling. *Computer scheduling of public transport urban passenger vehicle and crew scheduling*, pages 269–280.

Samà, M., D'Ariano, A., Corman, F., and Pacciarelli, D. (2017). A variable neighbourhood search for fast train scheduling and routing during disturbed railway traffic situations. *Computers and Operations Research*, 78:480–499.

Samà, M., Meloni, C., D'Ariano, A., and Corman, F. (2015). A multi-criteria decision support methodology for real-time train scheduling. *Journal of Rail Transport Planning and Management*, 5(3):146–162.

Samà, M., Pellegrini, P., D'Ariano, A., Rodriguez, J., and Pacciarelli, D. (2016). Ant colony optimization for the real-time train routing selection problem. *Transportation Research Part B: Methodological*, 85:89–108.

Samuelson, P. A. (1947). *Foundations of Economic Analysis*. Harvard University Press, Cambridge, MA.

Scheepmaker, G. M., Goverde, R. M., and Kroon, L. G. (2017). Review of energy-efficient train control and timetabling. *European Journal of Operational Research*, 257(2):355–376.

Schöbel, A. (2001). A model for the delay management problem based on mixed-integer-programming. *Electronic Notes in Theoretical Computer Science*, 50(1):1–10.

Schöbel, A. (2007). Integer programming approaches for solving the delay management problem. In Geraets, F., Kroon, L., Schoebel, A., Wagner, D., and Zaroliagiis, C., editors, *Algorithmic methods for railway optimization*, pages 145–170. Springer.

Serafini, P. and Ukovich, W. (1989). A Mathematical Model for Periodic Scheduling Problems. *SIAM Journal on Discrete Mathematics*, 2(4):550–581.

Shakibayifar, M., Sheikholeslami, A., and Jamili, A. (2018). A Multi-Objective Decision Support System for Real-Time Train Rescheduling. *IEEE Intelligent Transportation Systems Magazine*, 10(3):94–109.

Toletti, A. (2018). *Automated railway traffic rescheduling and customer information*. PhD thesis, ETH Zurich.

Toletti, A., Laumanns, M., and Weidmann, U. (2020). Coordinated railway traffic rescheduling with the Resource Conflict Graph model. *Journal of Rail Transport Planning and Management*, 15.

Törnquist, J. and Persson, J. A. (2007). N-tracked railway traffic re-scheduling during disturbances. *Transportation Research Part B: Methodological*, 41(3):342–362.

Törnquist Krasemann, J. (2012). Design of an effective algorithm for fast response to the re-scheduling of railway traffic during disturbances. *Transportation Research Part C: Emerging Technologies*, 20(1):62–78.

Van Thielen, S., Corman, F., and Vansteenwegen, P. (2018). Considering a dynamic impact zone for real-time railway traffic management. *Transportation Research Part B: Methodological*, 111:39–59.

Vanderbeck, F. and Wolsey, L. (2010). Reformulation and Decomposition of Intger Programs. In *50 Years of Integer Programming 1958-2008*, pages 431–502. Springer.

Vanderbei, R. (2014). *Linear Programming: Foundations and Extensions*. Springer, fourth edition.

Veelenturf, L. P., Wagelmans, A. P., Potthoff, D., Huisman, D., Kroon, L. G., and Maróti, G. (2016). A Quasi-Robust Optimization Approach for Crew Rescheduling. *Transportation Science*, 50(1):204–215.

Virtanen, P. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Williams, K. (2019a). Current railway models: Great Britain and overseas. https://www.gov.uk/government/collections/the-williams-rail-review.

Williams, K. (2019b). The rail sector in numbers. https://www.gov.uk/government/collections/the-williams-rail-review.

Xu, P., Corman, F., Peng, Q., and Luan, X. (2017). A train rescheduling model integrating speed management during disruptions of high-speed traffic under a quasi-moving block system. *Transportation Research Part B: Methodological*, 104:638–666.

Yang, X., Li, X., Ning, B., and Tang, T. (2016). A survey on energy-efficient train operation for urban rail transit. *IEEE Transactions on Intelligent Transportation Systems*, 17(1):2–13.

Yin, J., Tang, T., Yang, L., Xun, J., Huang, Y., and Gao, Z. (2017). Research and development of automatic train operation for railway transportation systems: A survey. *Transportation Research Part C: Emerging Technologies*, 85:548–572.

Yuan, J. and Medeossi, G. (2014). Statistical Analysis of Train Delays and Movements. In Hansen, I. and Pachl, J., editors, *Railway Timetabling & Operations*, pages 217–236. Eurailpress, 2nd edition.

Zhan, S., Kroon, L. G., Veelenturf, L. P., and Wagenaar, J. C. (2015). Real-time high-speed train rescheduling in case of a complete blockage. *Transportation Research Part B: Methodological*, 78:182–201.

Zhou, L., Tong, L. C., Chen, J., Tang, J., and Zhou, X. (2017). Joint optimization of high-speed train timetables and speed profiles: A unified modeling approach using space-time-speed grid networks. *Transportation Research Part B: Methodological*, 97:157–181.

Zucchini, W., MacDonald, I. L., and Langrock, R. (2016). *Hidden Markov Models for Time Series*. CRC Press, 2nd edition.

Zwaneveld, P. J., Kroon, L. G., and Van Hoesel, S. P. M. (2001). Routing trains through a railway station based on a node packing model. *European Journal of Operational Research*, 128:14–33.