A Study of Problems Modelled as Network Equilibrium Flows

Sam Thomas O'Neill

Submitted in partial fulfilment of the requirements of the University of Derby for the award of Doctor of Philosophy (PhD)

October 2021

College of Science and Engineering

University of Derby

Abstract

This thesis presents an investigation into selfish routing games from three main perspectives. These three areas are tied together by a common thread that runs through the main text of this thesis, namely selfish routing games and network equilibrium flows.

First, it investigates methods and models for nonatomic selfish routing and then develops algorithms for solving atomic selfish routing games. A number of algorithms are introduced for the atomic selfish routing problem, including dynamic programming for a parallel network and a metaheuristic tabu search. A piece-wise mixed-integer linear programming problem is also presented which allows standard solvers to solve the atomic selfish routing problem. The connection between the atomic selfish routing problem, mixed-integer linear programming and the multi-commodity flow problem is explored when constrained by unsplittable flows or flows that are restricted to a number of paths. Additionally, some novel probabilistic on-line learning algorithms are presented and compared with the equilibrium solution given by the potential function of the nonatomic selfish routing game.

Second, it considers multi-criteria extensions of selfish routing and the inefficiency of the equilibrium solutions when compared with social cost. Models are presented that allow exploration of the Pareto set of solutions for a weighted sum model (akin to the social cost) and the equilibrium solution. A means by which these solutions can be measured based on the Price of Anarchy for selfish routing games is also presented.

Third, it considers the importance and criticality of components of the network (edges, vertices or a collection of both) within a selfish routing game and the impact of their removal. Existing network science measures and demand-based measures are analysed to assess the change in total travel time and issues highlighted. A new

measure which solves these issues is presented and the need for such a measure is evaluated.

Most of the new findings have been disseminated through conference talks and journal articles, while others represent the subject of papers currently in preparation.

List of Publications and Conference Papers

Publications

1. O. Bagdasar, S. Berry, S. O'Neill, N. Popovici, and R. Raja. Traffic assignment: Methods and simulations for an alternative formulation of the fixed demand problem. *Mathematics and Computers in Simulation*, 155:360–373, 1 2019 - (Appendix A.1)

2. S. O'Neill, O. Bagdasar, and A. Liotta. An online learning approach to a multi-player n-armed functional bandit. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11974 LNCS, 2020 - (Appendix A.2)

3. S. O'Neill, O. Bagdasar, S. Berry, N. Popovici, and R. Raja. Modelling equilibrium for a multi-criteria selfish routing network equilibrium flow problem. *Mathematics and Computers in Simulation*, 2021 - (Appendix A.3)

4. S. O'Neill, P. Wrigley, and O. Bagdasar. A mixed-integer linear programming formulation for the modular layout of three-dimensional connected systems. *Mathematics and Computers in Simulation*, 2021 - (Appendix A.4)

5. S. O'Neill and T. O'Neill. The traffic assignment problem. *Mathematics Today*, pages 52–56, 4 2021 - (Appendix A.5)

Conference Papers

1. S. O'Neill. Traffic modelling: A network analysis of road networks of major cities and towns for England and Wales, *29th European Conference on Operational Research*, Valencia, Spain, 2018 - (Appendix B.1)

2. S. O'Neill. Traffic assignment problem: An overview of link, bush and route based solution methods, *16th EUROOPT Workshop on Advances in Continuous Optimization*, Almeria, Spain, 2018 - (Appendix B.2)

3. S. O'Neill, O. Bagdasar and S. Berry. An online learning approach to a multiplayer n-armed functional bandit, Numerical Computations: Theory and Algorithms, *The 3rd International Conference and Summer School* Calabria, Italy, 2019 - (Appendix B.3)

4. S. O'Neill, P. Wrigley and O. Bagdasar. A mixed-integer linear programming formulation for the modularisation of 3-dimensional connected systems, *2nd International Conference on Mathematical Modeling and Computational Methods in Science and Engineering*, India, 2020 - (Appendix B.4)

5. S. O'Neill, S. Berry and O. Bagdasar. Modelling of equilibrium for a multiobjective network equilibrium flow problem, *2nd International Conference on Mathematical Modeling and Computational Methods in Science and Engineering*, India, 2020 - (Appendix B.5)

Dedication

Emily

To my beautiful Emily. Thank you for being my rock and for all your support and love throughout this time. Love you!

Seth and Imogen

To my gorgeous kids, Seth and Imogen. Having you during this time is the best thing and it has made all of this so much easier.

Mum and Dad

To my wonderful parents, Jan and Tony. Thank you for all your support and love over the years. Dad thanks for making it through my Viva! Mum thanks for being so strong.

Joe and Matt

Thank you for being there during Dad's time in hospital. Joe thanks for all the hard work you did and Matt thanks for being a source of hope in a time of despair!

I love you all!

Declaration

I declare that this thesis has been composed solely by myself and that it has not been submitted, in whole or in part, in any previous application for a degree. Except where states otherwise by reference or acknowledgment, the work presented is entirely my own.

Sam Thomas O'Neill

October 2021

Acknowledgements

Thank you to my supervisors, Dr Ovidiu Bagdasar and Dr Stuart Berry, for providing help, guidance and feedback throughout this project. I would also like to thank Stuart for the seemingly endless supply of coffee and chats.

Contents

1 Introduction, Aims and Objectives **1**

1.1 Background 1

1.2 Preliminaries 3

1.3 Nonatomic Selfish Routing 23

1.4 Atomic Selfish Routing 30

1.5 Multi-objective Optimisation 33

1.6 Importance Measures in Selfish Routing 35

1.7 Aims and Objectives 37

2 Network Equilibrium: Models and Methods **38**

2.1 Introduction 39

2.2 Algorithms for Nonatomic Selfish Routing 40

2.3 Algorithms for Unweighted Atomic Selfish Routing 54

2.4 An Online Learning Approach to Unweighted Atomic Selfish Routing 61

2.5 Chapter Summary 67

3 Multi-criteria Network Equilibrium: Efficiency and Suboptimality **70**

3.1 Introduction 71

3.2 A Study into a Multi-criteria Selfish Routing Problem Considering Fuel Consumption 72

3.3 Measuring the Inefficiency of Equilibrium 87

3.4 Chapter Summary 89

4 Nonatomic Selfish Routing: Assessing Demand-Based Importance Measures **91**

4.1 Introduction 92

4.2 Network Science Measures 94
4.3 Demand-based Measures 108
4.4 An Improved Measure 127
4.5 A Paradoxical Effect 138
4.6 Results: Impact of Changes on Importance 141
4.7 Chapter Summary 146

5 Atomic Selfish Routing: An Application of Mixed-Integer Linear Programming 149

5.1 Introduction 150
5.2 Problem Statement 151
5.3 Methodology 153
5.4 Preliminary Implementation and Results 162
5.5 Extension to Atomic Selfish Routing Games 163
5.6 Chapter Summary 163

6 Summary and Conclusions 165

6.1 Introduction 165
6.2 Chapter One 166
6.3 Chapter Two 166
6.4 Chapter Three 166
6.5 Chapter Four 167
6.6 Chapter Five 168
6.7 Critical Reflection 168
6.8 Potential Applications of Work 171
6.9 Conclusion 172

Bibliography 173

A Appendix: Publications 190

A.1 Traffic assignment: Methods and simulations for an alternative formulation of the fixed demand problem 191
A.2 An online learning approach to a multi-player n-armed functional bandit 192

A.3 Modelling equilibrium for a multi-criteria selfish routing network equilibrium flow problem . . . 193
A.4 A mixed-integer linear programming formulation for the modular layout of three-dimensional connected systems . . . 194
A.5 The traffic assignment problem - Mathematics Today . . . 195

B Appendix: Conference Papers **196**

B.1 Traffic modelling: A network analysis of road networks of major cities and towns for England and Wales . . . 197
B.2 Traffic assignment problem: An overview of link, bush and route based solution methods . . . 198
B.3 An online learning approach to a multi-player n-armed functional bandit 199
B.4 A mixed-integer linear programming formulation for the modularisation of 3-dimensional connected systems . . . 200
B.5 Modelling of equilibrium for a multi-objective network equilibrium flow problem . . . 201

C Appendix: Networks **202**

C.1 Braess Network . . . 203
C.2 Qiang and Nagurney Network . . . 206
C.3 Sioux Falls Network . . . 209
C.4 Dial's Network . . . 214
C.5 Grid 5x5 Network . . . 219

List of Figures

1.1 Pigou's example. 8
1.2 Edge travel time functions. 10
1.3 Path decomposition for example given in Figure 1.2. 10
1.4 Braess network. 15
1.5 A hierarchy of potential games. 20

2.1 Directed network example. 49
2.2 Example bush for vertex 1. 50
2.3 Comparison of implemented algorithms for solving Sioux Falls. 53
2.4 Analysis of gradient projection performance for solving Dial's network. 54
2.5 Computational efficiency for 3 and 10 parallel edges. 59
2.6 Piecewise linear approximation of example edge function $t_e(x_e) = 2+x_e^4$. 59
2.7 Log-lin plots of total cost and cumulative regret for the 4 algorithms averaged over all test data. 68
2.8 Log-lin plots illustrating the convergence of players costs for the 4 origin/destination pairs. 68

3.1 Single commodity with parallel edges. 73
3.2 Fuel consumption curve $F_e(s_e)$ for $A = 0.0019$, $B = -0.2784$ and $C = 17.337$. The dotted lines indicate $F_e(s_e(x_e))$ (from right to left). The squares show $(m_e, F_e(m_e))$ for $m_1 = 110$, $m_2 = 95$ and $m_3 = 65$ kph (see Table 3.1). 76
3.3 Interplay between edge flow functions. 78
3.4 Comparison of normalised and original scale plots. 84
3.5 Weighted sum model for 3 different configurations of $\mathbf{m}$. 84

3.6 Total fuel consumption vs total travel time for varying free-flow speeds m_e. 86
3.7 Decomposition of user equilibrium solutions plotted against mean edge speed $\overline{m}$. 86
3.8 Illustration of the ratios described by equation (3.18) using Euclidean distance between points. 88

4.1 Chesterfield BUA and network extracted via OSNMX. 99
4.2 Vertex measures for Chesterfield. 100
4.3 Pearson correlation matrix for selected measures for UK BUA's. . . . 102
4.4 Degree distributions of selected cities. 103
4.5 Explanation of vertex degree values of 2 and 6. 103
4.6 Pearson correlation matrix for network science measures (removal of an edge e). 106
4.7 Pearson correlation matrix for network science measures (removal of a vertex v). 107
4.8 Braess network. 114
4.9 Network from Qiang and Nagurney [139]. 115
4.10 Comparison of I_{NQ}, I_{LM} and I_z for the removal of an **edge** e. 119
4.11 Pearson correlation matrix for removal of an **edge** e. 120
4.12 Comparison of I_{NQ}, I_{LM} and I_z for the removal of an **edge** e (Ranking). 121
4.13 Pearson correlation matrix for removal of an **edge** e (Ranking). . . . 122
4.14 Comparison of I_{NQ}, I_{LM} and I_z measures for the removal of a **vertex** v. 123
4.15 Pearson correlation matrix for removal of a **vertex** v. 124
4.16 Comparison of I_{NQ}, I_{LM} and I_z for the removal of a **vertex** v (Ranking). 125
4.17 Pearson correlation matrix for removal of a **vertex** v (Ranking). . . . 126
4.18 Comparison of $\hat{I}_R$ measure for the removal of an edge e. 135
4.19 Comparison of $\hat{I}_R$ measure for the removal of a vertex v. 136
4.20 Edge and vertex importance measures for Sioux Falls. 137
4.21 Sioux Falls - Network edges coloured by importance $\hat{I}_R$. 138
4.22 Amended Pigou network. 139
4.23 Amended Braess network. 140

4.24 Edge and vertex importance measures for side-to-side external demand. 143
4.25 Edge and vertex importance measures for corner-to-corner external demand. 144
4.26 Network edges coloured by importance $\hat{I}_R$. 145
4.27 Demand changes for side-to-side network. 146

5.1 Multigraph with a single commodity $(1, 4)$. 152
5.2 Example multi-commodity flow problem. 154
5.3 Optimal path flow solution for example given in Figure 5.2. 155
5.4 Edge (3,4) capacity increased from 10 to 30. 157
5.5 Optimal path flow solution for (5.2) with $N_k = 1,\ \forall k \in \{1, \ldots, K\}$. . 158
5.6 Transformation of multigraph into a graph via introduction of dummy vertices for commodities. 159

C.1 Braess network diagram. 203
C.2 Qiang and Nagurney network diagram. 206
C.3 Sioux Falls network diagram. 209
C.4 Dial's network diagram. 214
C.5 Grid 5x5 network diagram. 219

List of Tables

1.1 Edge cost functions for example given in 1.4. 15

2.1 Comparison of nonatomic selfish routing method types 54

3.1 Edge travel time parameters. 75

3.2 Comparison of weighted sum model solutions for $\mathbf{m}^1 = (110, 80, 65), \mathbf{m}^2 = (95, 95, 95)$ and $\mathbf{m}^3 = (110, 80, 40)$. 83

4.1 Global averages of measures for Basildon vs Chesterfield. 101

4.2 Central tendency measures for UK. 101

4.3 Edge importance measures for Braess network. 113

4.4 Vertex importance measures for Braess network. 114

4.5 Edge importance measures for network in Figure 4.9. 116

4.6 Edge importance measure rankings for network in Figure 4.9. 116

4.7 Vertex importance measures for network in Figure 4.9. 116

4.8 Vertex importance measure rankings for network in Figure 4.9. 117

4.9 Edge importance measures for Braess network given in Figure 4.8. . . 132

4.10 Vertex importance measures for Braess network given in Figure 4.8. . 132

4.11 Edge importance measures for network given in Figure 4.9. 133

4.12 Edge importance measure rankings for network given in Figure 4.9. . 133

4.13 Vertex importance measures for network given in Figure 4.9. 133

4.14 Vertex importance measures for network given in Figure 4.9. 134

5.1 Comparison of solutions for (5.2) with varying edge capacities. 162

C.1 Link functions for Braess network. 204

C.2 Origin-destination trip table for Braess network. 205

C.3 Link functions for Qiang and Nagurney network. 207

C.4 Origin-destination trip table for Qiang and Nagurney network. 208
C.5 Link functions for Sioux Falls network. 211
C.6 Origin-destination trip table for Sioux Falls network. 213
C.7 Link functions for Dial's network. 216
C.8 Origin-destination trip table for Dial's network. 218
C.9 Link functions for grid 5x5 network. 221
C.10 Origin-destination trip table for grid 5x5 network. 222
C.11 Origin-destination reduced trip table for grid 5x5 network. 224
C.12 Origin-destination increased trip table for grid 5x5 network. 226

Acronyms

AEC Average Excess Cost.

BUA Built-up Area.

EPG Exact Potential Game.

GAP Relative Gap.

KKT Karush–Kuhn–Tucker.

MAB Multi-armed Bandit.

MAX-SAT Maximum Satisfiability Problem.

MCFP Multi-commodity Flow Problem.

MILP Mixed-integer Linear Programming.

MINLP Mixed-integer Nonlinear Programming.

NP Non-deterministic Polynomial Time.

OD Origin-destination.

OPG Ordinal Potential Game.

PAS Paired Alternative Segments.

PLS Polynomial Local Search.

PoA Price of Anarchy.

PoS Price of Stability.

PPAD Polynomial Parity Arguments on Directed graphs.

TAP Traffic Assignment Problem.

TAPAS Traffic Assignment by Paired Alternative Segments.

VoT Value of Time.

WPG Weighted Potential Game.

List of Symbols

Set Theory

$\mathbb{R}$	The set of real numbers
$\mathbb{Z}$	The set of integers
$\mathbb{N}$	The set of natural numbers

Game Theory

N	The finite set of n players
S	The set of strategy profiles (strategy space) $S = S_1 \times \cdots \times S_n$, where S_i is the set of strategies available to player i
s	A given strategy profile, $s = (s_1, \ldots, s_n) \in S$
u	The vector of utility functions $u = (u_1, \ldots, u_n)$, where $u_i : S \mapsto \mathbb{R}$ is the utility function for player i
C	Social cost function - $C : S \mapsto \mathbb{R}$
W	Social welfare function - $W : S \mapsto \mathbb{R}$

Selfish Routing

G	$G = (V, E)$ - directed graph of vertices and edges
V	The set of vertices for a graph or network G
E	The set of edges for a graph or network G
K	The set of all commodities for a network
Π	The set of all paths
Π_j	The set of all paths for commodity $j \in K$
π	A given path $\pi \in \Pi_j$
d_j	The demand to be routed for commodity $j \in K$

f_π The flow on path π

c_π The cost of the path π

x_e The flow on an edge e

$t_e(x_e)$ The congestible cost function of an edge e

U Objective function for user equilibrium

T Objective function for system optimal

Congestion Games

N The finite set of n players

$\mathcal{E}$ The set of congestible elements

$\mathcal{A}$ The set of pure strategy profiles

l The vector of cost functions $l = (l_e)_{e \in E}$

Potential Games

Φ The potential function - $\Phi : S \mapsto \mathbb{R}$

Chapter 1

Introduction, Aims and Objectives

Chapter Preface

This chapter places the work presented in this thesis into the context of congestion games and, in particular, selfish routing. It reviews previous work done into selfish routing and identifies areas to which this thesis makes particular contributions. This thesis aims to investigate these areas and present work that contributes to the identified gaps.

Chapter Keywords

Selfish Routing, Network Flow, Congestion Games, Traffic Assignment Problem

1.1 Background

The study of multiagent systems with strategic agents that behave in a self-interested manner has been of practical and theoretical interest for almost a century. Game theory has provided a powerful framework for the study and analysis of such systems, with the concepts of equilibria, social welfare, social cost, drawing inspiration from the fields of philosophy, politics and economics. With the continued rise of decentralised systems such as the internet, economy and wireless networks, it has become increasingly apparent that the selfish behaviour of multiple agents (as modelled by game theory) is no longer confined to traffic or the classic examples given in

textbooks (e.g. the prisoner's dilemma), but can be applied to numerous real-world problems.

Noncooperative games, in which no external mechanism incentivises cooperation between agents, provides a very natural way of modelling concepts which arise in the real-world such as processor scheduling, routing and network design [92]. For example, in transport modelling, the traffic assignment stage is underpinned by a widely accepted phenomenon that, given the selfish nature of drivers' attempts to minimise their own travel time, a system will settle to a state of equilibrium in which no driver can benefit from switching their current path (there is no unilateral profitable deviation) [167]. Although this system can be considered stable, it does not (in almost all cases) result in the minimum average journey (minimum total time for all drivers) which can only happen with social cooperation or central control. Under these circumstances, certain drivers will be worse off when compared to the stable state of equilibrium; however, the average driver benefits. In the case of traffic assignment, the attractiveness of one state compared to the other may be a matter of opinion but it is without debate that for some real-world applications, such as processor scheduling, minimising the total time taken is preferred [22].

In recent years the field of algorithmic game theory [127] has emerged in an attempt to answer the core questions around selfish behaviour of multi-agent systems using the principles of game theory as a foundation for the design and implementation of algorithms. As well as the modelling of existing systems, research has been undertaken into the design of multiagent games whereby the equilibrium state generated by self-interest coincides with a socially optimal one - a high profile use of this is second price auctions employed by the likes of eBay [143]. Common measures for the suboptimality of an equilibrium state, when compared with the socially optimal, are the *price of anarchy* (PoA) and the *price of stability* (PoS).

The natural stability that arises out of the selfish behaviour of agents has been studied extensively. However, it has been found that computing a stable solution (mixed-strategy Nash equilibrium) is PPAD-complete (Polynomial Parity Arguments on Directed graphs) [134, 47, 60] and thus the design of exact convergent algorithms is not only hard but a thankless task. Thus, the study of classes of problems that are computable in polynomial time is warranted; however, for intractable

problems, advances in combinatorial optimisation and metaheuristics are needed.

Of particular interest is the class of potential games, which have a set of useful properties such as the existence of at least one pure-strategy Nash equilibrium, known to be PLS-complete (Polynomial Local Search) [60, 141] whereby the computation of a stable state takes an exponential number of steps, but a solution can be validated in polynomial time via searching the local neighbourhood. This class of games has been shown to model network flow problems in distributed networks, such as resource allocation in wireless networks and distributed power control and scheduling. The presence of at least one Nash equilibrium guarantees that both the price of anarchy and price of stability are defined for this class of games and provide measures that can be used for analysis and identification of issues, suggesting potential improvements, such as infrastructure investment or patron subsidiaries.

Selfish routing games are a class of potential games with specific properties (see Section 1.2.5) and are born out of the area of transportation planning and modelling. The concept of selfish routing is at the heart of these games and shares the same principles as game theory, namely the search for mixed-strategy Nash equilibrium; however, it is restricted to the flows on a network.

This thesis looks at selfish routing games from three main perspectives. First, it investigates methods and models for nonatomic selfish routing games and then develops algorithms for solving atomic selfish routing games. Second, it considers multi-criteria extensions of selfish routing and the inefficiency of the equilibrium solutions when compared with the social cost. Third, it considers the importance and criticality of components of the network (edges, vertices or a collection of both) within a selfish routing game and the impact of their removal.

The notation used within this thesis is defined, when introduced, within each chapter.

1.2 Preliminaries

1.2.1 Game Theory

Game Theory is the applied branch of mathematics concerned with the study of strategic interactions between self-interested rational agents (often referred to as

players). First studied by John Von Neumann in 1928, the main theoretic concepts were formalised in the 1944 treatise *Theory of Games and Economic Behaviour* [164]. Loosely, a game consists of a set of agents, a set of available agent strategies and the set of reachable game states (combinations of agents' played strategies). To measure an agent's happiness with a the state of the game, a utility function is employed that, for each player, maps the state of the game to a real-valued function. In general two types of games are considered, noncooperative and cooperative. In noncooperative games, individual agents play strategies to benefit their personal standpoint, preferences, needs, etc, whereas in cooperative games, agents play strategies which benefit the group (self-interest here is still applicable but it is aligned with group interest). The remainder of this section provides an overview and formally defines key concepts necessary in the analysis and optimisation of network flow problems. For all definitions given in Section 1.2.1 see Chapter 1 and 2 of [101].

1.2.1.1 Normal Form

A game in which agents (players - in the language of game theory) move simultaneously, i.e. without knowledge of their opponents move, can be represented in *Normal Form* as follows:

Definition 1.2.1 (Normal form game)**.** *A game in normal form is a tuple* (N, S, u) *where:*

- N *- The finite set of* n *players;*
- S *- The set of strategy profiles (strategy space)* $S = S_1 \times \cdots \times S_n$*, where* S_i *is the set of strategies available to player* i*. Each vector* $s = (s_1, \ldots, s_n) \in S$ *is a given strategy profile;*
- u *- The vector of utility functions* $u = (u_1, \ldots, u_n)$*, where* $u_i : S \mapsto \mathbb{R}$ *is the utility function for player* i*.*

Strategies are either chosen as deterministic (pure-strategies), whereby the game state determines the pure-strategy an agent will play, or probabilistic (mixed-strategies), agents play a pure-strategy with a given probability. A pure-strategy is just the degenerate case of a mixed-strategy in which the deterministic pure-strategy to be

played at a given move has probability 1, and all other pure-strategies have probability 0.

1.2.1.2 Pareto Optimality

Pareto optimality is a state in which any attempt to benefit an agent's utility results in a loss to some other agent. This observation is made by considering all the agents in the game.

Definition 1.2.2 (Pareto dominant)**.** *Given a strategy profile s and s', s Pareto dominates s' if there exists for some $j \in N$, $u_j(s) > u_j(s')$ and that $u_i(s) \geq u_i(s')$, $\forall i \in N$.*

That is, no agent is worse off given some agents move to a better strategy.

Definition 1.2.3 (Pareto optimality (efficiency))**.** *A strategy profile s is Pareto optimal if there does not exist another strategy profile s' that dominates s.*

Given the partial ordering from definition 1.2.2there must be at least one optimum; however, multiple optima may exist and the set of all Pareto optimal strategy profiles is referred to as the *Pareto frontier*.

1.2.1.3 Nash Equilibrium

The concept of Nash equilibrium was first introduced by John F. Nash Jr. [121] who built on the work done by von Neumann and Morgenstern [91, 164] for 2 player noncooperative games. It differs from Pareto optimality in that the benefit of a different strategy on an agent's utility is only from the standpoint of the agent in question. Let $s_{-i} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)$ be the strategy profile of all other agents and s_i the strategy of agent i. The best response of agent i to the strategy profile s_{-i} is defined as [101]:

Definition 1.2.4 (Best response)**.** *The mixed strategy $s_i^* \in S_i$ is a best response for an agent i to the strategy s_{-i} such that $u_i(s_i^*, s_{-i}) \geq u_i(s_i, s_{-i})$, $\forall s_i \in S_i$.*

If an agent knew the strategies of the other agents, s_i^* is a strategy (not necessarily unique) that maximises their utility. The concept of Nash equilibrium follows [101],

Definition 1.2.5 (Nash equilibrium)**.** *A strategy s is a Nash equilibrium if s_i is the best response to s_{-i}, $\forall i \in N$.*

Therefore, no agent has any motivation to change their strategy given knowledge of the other agents strategies.
Nash proved that, for any mixed-strategy game with a finite number of agents and strategies, there exists at least one equilibrium state [121]. If the best response (1.2.4) is strict ($>$), then the Nash equilibrium is referred to as a strict Nash and, for a weak best response ($\geq$), a weak Nash. A strict Nash is necessarily a pure-strategy equilibrium [101].

1.2.1.4 Social Choice Optimality

A strategy profile of a game is said to be socially optimal (sometimes referred to as system optimal) if it is the optimal solution for some given objective function in the feasible set of strategy profiles [127], in particular a social welfare objective is maximised and a social cost is minimised. In general, the self-interest and motivation of players do not align with the social objective and thus a strategy that is an equilibrium from a strategy that is socially optimal differ. In fact, the socially optimal state can be considered unstable in a system in which players have self-interested preferences. Parallels are easily drawn with philosophy, politics and economics, e.g. utilitarianism and social choice theory, where a greater good is sought.

Definition 1.2.6 (Socially Optimal)**.** *Given a game (N, S, u) and an objective function - $C : S \mapsto \mathbb{R}$, the strategy $s^* \in S$ is socially optimal if and only if:*

$$C(s^*) \leq C(s), \quad \forall s \in S.$$

Whilst this definition minimises the social cost, the social welfare can be maximised by letting $C(s) = -W(s)$, where $W : S \mapsto \mathbb{R}$ is the social welfare objective function.

1.2.1.5 Price of Anarchy

The price of anarchy was first introduced by Koutsoupias and Papadimitriou [89] and is a commonly used measure in the analysis of how suboptimal equilibria are from

the socially optimal state. It provides an upper bound when measuring the ratio of the worse equilibrium compared with the socially optimal solution. It is analogous to the approximation ratio used in measuring how far an the approximate solution obtained by an algorithm is from the optimal solution and is a tool used for analysis in algorithmic game theory [127].

Definition 1.2.7 (Price of anarchy)**.** *For a game $G(N, S, u)$ with social objective function $C : S \mapsto \mathbb{R}$ and the set of Nash equilibrium strategy profiles S_E, the price of anarchy is:*

$$\frac{\max_{s \in S_E} C(s)}{\min_{p \in S} C(p)}.$$

1.2.1.6 Price of Stability

The price of stability is another measure by which the best equilibrium is used in the analysis of how suboptimal equilibria are. Rationale for this measure is to consider games in which the price of stability and price of anarchy differ considerably. Such a case motivates a strategy in which the game converges to a good equilibria. For example, Anshelevich et al [5] showed that for the network design game with fair cost allocation the price of stability is at most $O(log(n))$, where n is the number of users. First studied by Correa, Schulz and Moses [41], the first appearance of the name price of stability is given in [5].

Definition 1.2.8 (Price of stability)**.** *For a game (N, S, u) with objective function $C : S \mapsto \mathbb{R}$ and the set of Nash equilibrium strategy profiles S_E, the price of stability is:*

$$\frac{\min_{s \in S_E} C(s)}{\min_{p \in S} C(p)}.$$

1.2.2 Selfish Routing

The concept of selfish routing arises naturally in many real world scenarios where agents wish to route their flow through a network as cheaply as possible for a given commodity (origin-destination (OD) pair). Examples include traffic networks, computing networks, mechanical and electrical networks. The most recognisable of these to the layperson is traffic networks, whereby each vehicle wishes to route itself between two points and research into this area has been ongoing since the example

given by Pigou in the 1920s [1]. Seminal work was undertaken by Wardrop [167], Beckmann, McGuire and Winsten [19], Braess [33], Dafermos and Sparrow [43] which formed the basis of traffic assignment in transportation theory. The term 'selfish routing' was coined by Roughgarden and Tardos [147] and makes use of the price of anarchy.

1.2.2.1 Pigou's Example

Consider the graph given in Figure 1.1 with flow dependent edge cost functions $c_e(x_e)$.

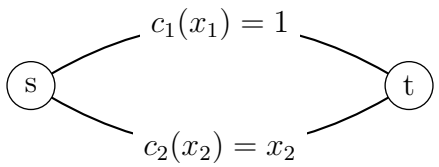


Figure 1.1: Pigou's example.

The edge cost functions determine the travel time on each edge as a function of the fraction of the total flow using that edge. In this example the total demand to be routed on the network between the source s and the sink t is 1 unit. i.e. the demand routed on each of the edges sums to 1, $x_1 + x_2 = 1$.
Under the assumption that an agent (some fraction of the flow) wishes to minimise the travel time that it experiences, no agent has any incentive to ever use the upper edge. Consider the case where ϵ amount of flow has been routed on the upper edge, then the cost of the upper and lower edges are 1 and $1 - \epsilon$ respectively. Given this strategy profile, the best response for any vehicle (agent) residing on the upper edge would be to switch to the lower edge. Thus equilibrium is reached with $(x_1, x_2) = (0, 1)$.

The total travel time experienced by all users for the equilibrium is $x_1c_1(x_1) + x_2c_2(x_2) = 0 \cdot 1 + 1 \cdot 1 = 1$, however the minimum overall travel time possible is to route half the flow on each edge $(x_1, x_2) = (\frac{1}{2}, \frac{1}{2})$ resulting in a cost of $\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot \frac{1}{2} = \frac{3}{4}$. The PoA for this example is therefore $\frac{4}{3}$.
This example motivates a central theme in selfish routing, comparing the minimum overall travel time possible and the overall travel time subject to selfish routing.

1.2.2.2 Wardrop's Principles

In his 1952 paper *Road Paper. Some Theoretical Aspects of Road Traffic Research* [167] Wardrop proposed two approaches to analysing the flow on a traffic network [135].

Wardrop's first principle (user equilibrium):

The journey times on all the routes used between an origin and a destination are equal, and less than those which would be experienced by a single vehicle on any unused route.

Wardrop's second principle (system optimal):

The average journey time is at a minimum.

Under specific restrictions (Section 1.2.4), user equilibrium has been shown to be equivalent to a non-cooperative Nash equilibrium for discrete flows by Rosenthal [142] and extended for continuous flows by Devarajan [51].

1.2.3 Traffic Assignment Problem (TAP)

The Traffic Assignment Problem (TAP) has been at the heart of the transport modelling community for over 60 years since its formulation by Beckmann, McGuire and Winsten [19] and under a set of simple assumptions provides a unique solution, in terms of the edge flows, to Wardrop's first principle - user equilibrium.

A full formulation of the TAP model (nonatomic selfish routing) is given in Section 1.3.1.

1.2.3.1 Illustrative Example

The following introductory example given by Sheffi [152] serves to illustrate the idea of sefish routing of traffic between commodities. Figure 1.2 depicts the flow dependent travel time functions for each edge in the network and Figure 1.3 highlights the two potential paths $\{\pi_1, \pi_2\}$ for the commodities (A, E) and $\{\pi_3, \pi_4\}$ for (B, E).

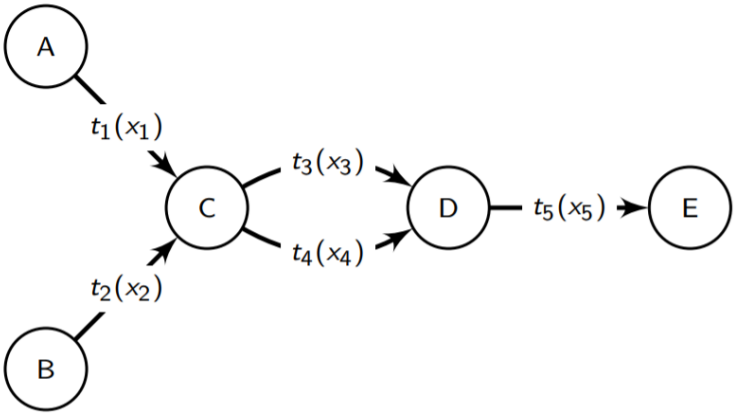
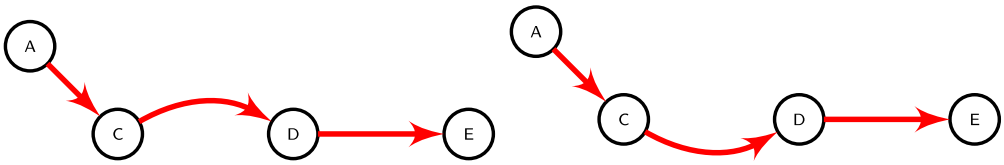
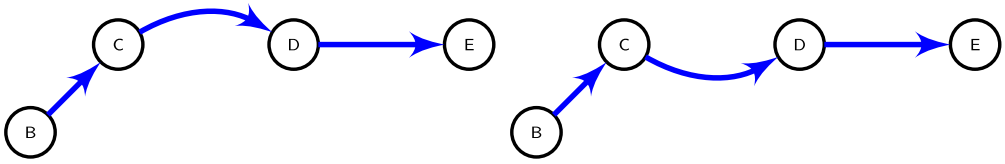


Figure 1.2: Edge travel time functions.



(a) Paths π_1 and π_2 between A and E.



(b) Paths π_3 and π_4 between B and E.

Figure 1.3: Path decomposition for example given in Figure 1.2.

For the given example the travel time functions for the five roads (edges) are

$$t_1(x_1) = 1$$
$$t_2(x_2) = 2$$
$$t_3(x_3) = 2 + x_3$$
$$t_4(x_4) = 1 + 2x_4$$
$$t_5(x_5) = 1,$$

where $\mathbf{x} = (x_1, x_2, x_3, x_4, x_5)^T \in \mathbb{R}^5_{\geq 0}$ denotes the amount of traffic flow on each of the 5 edges. The flow on the 4 paths (Figure 1.3) is represented by $\mathbf{f} = (f_1, f_2, f_3, f_4)^T \in \mathbb{R}^4_{\geq 0}$ and the flow on an edge e is found by summing the flow on the paths using edge e, e.g. $x_1 = f_1 + f_2$. The cost of the paths, $\mathbf{c} = (c_1, c_2, c_3, c_4)^T$, is then the sum of the travel times of its edges and thus also dependent on any other flow using the same edges, e.g.

$$c_1(\mathbf{f}) = t_1(x_1) + t_3(x_3) + t_5(x_5).$$

The objective of user equilibrium is to route a demand d_1 between (A, E) such that the cost of the used paths is equal and to separately route a demand d_2 between (B, E) such that the cost of the used paths is equal. More formally, find a set of path flows $\mathbf{f} = (f_1, f_2, f_3, f_4)^T$ that satisfy a routing demand d_1 for (A, E) given by $f_1 + f_2 = d_1$ and a routing demand d_2 for (B, E) given by $f_3 + f_4 = d_2$, such that the cost of the nonempty paths for each commodity are equal and less than the empty paths.

For the given example, the commodity demands are:

$$d_1 = 2$$
$$d_2 = 3.$$

A solution to user equilibrium is $\mathbf{f}^* = (1, 1, 2, 1)^T$, which results in the set of edge flows $\mathbf{x}^* = (2, 3, 3, 2, 5)^T$. The cost of the paths for each commodity are equal and minimised.

$$c_1(\mathbf{f}) = c_2(\mathbf{f}) = 7$$
$$c_3(\mathbf{f}) = c_4(\mathbf{f}) = 8.$$

It can be shown that given certain conditions for the travel time functions, the edge flow vector $\mathbf{x}^*$ is unique [135, 152], see Section 1.3.3. Given the solution $\mathbf{x}^*$ is unique, what about the path flow vector $\mathbf{f}^*$, is this unique?

To illustrate that this is not the case it is possible to form a linear system of equations. First, as the flow on an edge is the sum of the paths using the edge write $\mathbf{x} = \Delta\mathbf{f}$, where $\mathbf{\Delta}$ is the edge/path incidence matrix - note that this also gives convenient expressions for the flows and their respective costs, $\mathbf{f} = \Delta^{\mathbf{T}}\mathbf{x}$ and $\mathbf{c} = \Delta^{\mathbf{T}}\mathbf{t}$, where $\mathbf{t} = (t_1(x_1), t_2(x_2), t_3(x_3), t_4(x_4), t_5(x_5))^T$. Second, $f_1 + f_2 = d_1$ and $f_3 + f_4 = d_2$ can be expressed by the relationship $\mathbf{Af} = \mathbf{d}$, where $\mathbf{A}$ specifies the paths for a given commodity and $\mathbf{d}$ is the demand for said commodity. Thus the unique solution $\mathbf{x}^*$ can be represented by the system of equations

$$\begin{bmatrix} \mathbf{\Delta} \\ \mathbf{A} \end{bmatrix} \mathbf{f} = \begin{bmatrix} \mathbf{x}^* \\ \mathbf{d} \end{bmatrix}.$$

The given example is a system of $[7 \times 4]$ equations,

$$\begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ d_1 \\ d_2 \end{bmatrix} = \begin{bmatrix} 2 \\ 3 \\ 3 \\ 2 \\ 5 \\ 2 \\ 3 \end{bmatrix},$$

having rank $\left(\begin{bmatrix} \mathbf{\Delta} \\ \mathbf{A} \end{bmatrix} \right) = 3$, resulting in the underdetermined system with one free variable e_1 given by the system of equations

$$\begin{aligned} f_1 &= e_1 \\ f_2 &= 2 - e_1 \\ f_3 &= 3 - e_1 \\ f_4 &= e_1. \end{aligned}$$

Thus, infinitely many path flow solutions can be generated, demonstrating that path flow solutions are not necessarily unique.

1.2.3.2 Brief History of TAP

The formulation of the traffic assignment problem with elastic demand (TAP-E) was first developed by Beckmann, McGuire and Winsten [19] who prove that the optimal solution to the user equilibrium problem is unique with regards to the flow on the edges. Dafermos and Sparrow [43] defined the fixed demand case TAP in its edge-path form. The idea of marginal cost taxation is discussed by Pigou [1], Beckmann, McGuire and Winsten [19] and Dafermos and Sparrow [43]. Roughgarden [147] makes use of the idea in his analysis of the price of anarchy. Murchland [114], and Dantzig [45] showed that TAP and TAP-E are equivalent under network transformations that introduce edges to simulate the behaviour of the inverse demand function. Thus it had been sufficient to develop methods to solve the fixed demand case. The convex combination (Frank-Wolfe) algorithm was first presented as a method for solving quadratic programs [65]; Bruynooghe, Gilbert and Sakarovitch [34] provide the first known application of the Frank-Wolfe algorithm for TAP; for a detailed survey of link-based methods (link-based is the name given in the literature, where link is an alternative name for edge) and models, see Patrikkson [135]. Owing to the convergence issues surrounding link-based methods, path-based (e.g. Jayakrishnan et al [83]) and bush-based (hybrid link-path) methods (e.g. Dial [57], Bar-Gera [13]) have been developed converging to a greater degree of accuracy. Empirical studies comparing edge-based, path-based and bush-based methods are found in ([57], [136], [154]).

It is easy to understate the contribution and importance of the work of Martin J. Beckmann. Boyce, Mahmassani and Hani [32] provide a retrospective of the influential 1956 work *Studies in the Economics of Transportation* [19] and Hauptmann, Krelle and Mosler capture his contributions made to the fields of Operations Research and Economic Theory [75]. During the early 1950s Beckmann worked on a continuous model of transport [17] [18] utilising the Euler-Lagrange equations. Then, with the newly presented conditions of optimality by Khun and Tucker [87], Beckmann and his colleagues arrived at the objective function for TAP (see Boyce for a more detailed discussion [30]).

1.2.3.3 System Optimal

The traffic assignment problem is equivalent to Wardrop's first principle but it is easily modified to be equivalent to Wardrop's second principle. The second principle states that the total travel time spent by all routed flow for all commodities should be minimised. As the cost experienced by flow on the edge e is $t_e(x_e)$, the total cost experienced by all flow for edge e is then $x_e t_e(x_e)$. Thus, the total travel time T experienced for all flow is the sum over all edges of the total edge costs given by the formula $T(\mathbf{x}) = \sum_{e \in E} x_e t_e(x_e)$.

1.2.3.4 Braess Paradox

Pigou's example illustrates a key point for the traffic assignment problem (inherent in Game Theory) that selfish behaviour rarely leads to a socially optimal solution, i.e. although some may suffer, the overall effect is beneficial to the group. More disturbing is the paradox discovered by Braess [33] in which the addition of an edge can actually lead to an equilibrium flow that is not only socially suboptimal, but also detrimental to all users. The difficulty of discovering a Braess paradox has been shown to be NP-hard by Roughgarden [145]. It has also been shown by Nagurney that the wisdom of crowds prevails at a high enough demand [116]. That is to say, for a given demand it is profitable for a user to take the Braess path; however, beyond that the users shift towards the non-Braess paths, until at a high enough demand the Braess path is no longer profitable to any user.

Remarkably the Braess paradox has been found to exist in many real physical systems and field such as transportation networks [144], electrical engineering, mechanical systems [39], biology [113] and sports [153, 72].

To illustrate the paradox the following example from [118] is presented, Figure 1.4 displays the classic Braess network, with cost functions given by Table 1.1 and a single commodity which routes a demand $d = 6$ between vertices 1 and 4.

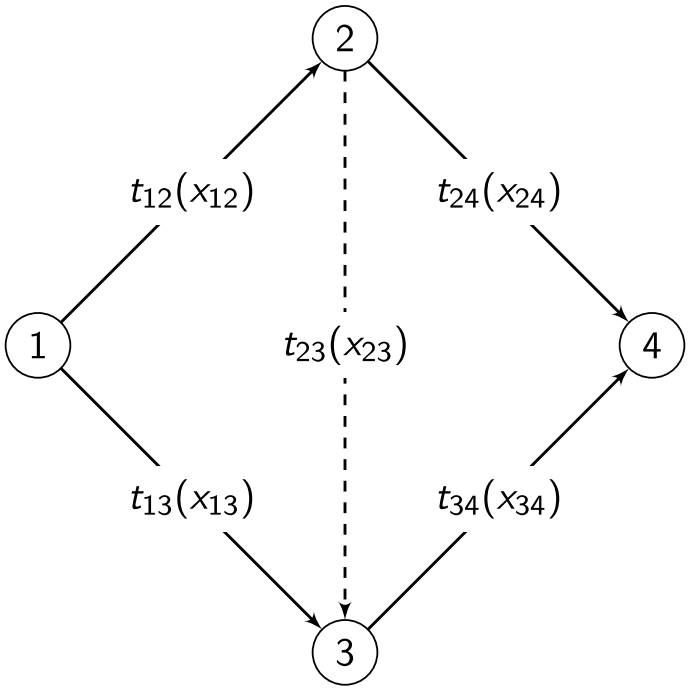


Figure 1.4: Braess network.

x_e	$t_e(x_e)$
x_{12}	$t_{12}(x_{12}) = 10x_{12}$
x_{13}	$t_{13}(x_{13}) = x_{13} + 50$
x_{23}	$t_{23}(x_{23}) = x_{23} + 10$
x_{24}	$t_{24}(x_{24}) = x_{24} + 50$
x_{34}	$t_{34}(x_{34}) = 10x_{34}$

Table 1.1: Edge cost functions for example given in 1.4.

Consider first the case when the dotted edge between vertices 2 and 3 is not present. In this case the equilibrium cost can be calculated by equating the cost of path π_1, $c_{\pi_1} = 10x_{12} + x_{24} + 50$, passing through vertices $1-2-4$ and the cost path π_2, $c_{\pi_2} = x_{13} + 10x_{34} + 50$, passing through vertices $1-3-4$ which results in the equation

$$10x_{12} + x_{24} + 50 = x_{13} + 10x_{34} + 50. \tag{1.1}$$

Clearly the flow on the edges on path π_1 are equal and the flow on the edges on path π_2 are equal, i.e. $x_{12} = x_{24}$ and $x_{13} = x_{34}$, and the flow leaving vertex 1 must be 6, i.e. $x_{12} = x_{13} = 6$. Substituting these into equation (1.1) gives the equilibrium edge

flow solution $x_{12} = x_{24} = x_{13} = x_{34} = 3$ and path flow solution $f_{\pi_1} = f_{\pi_2} = 3$. The cost of the paths $c_{\pi_1} = c_{\pi_2} = 83$.

Considering the case where the edge $(2, 3)$ is introduced to the current equilibrium solution. The cost of the path π_3 (which is currently empty, i.e. $x_{23} = 0$) passing through vertices $1 - 2 - 3 - 4$ is $c_{\pi_3} = 10x_{12} + x_{23} + 10 + 10x_{34} = 70$. Thus there is an incentive for users to shift their paths from the higher cost paths of π_1 and π_2 to the lower cost path π_3. Similarly if all flow is placed on the new path π_3, then paths π_1 and π_2 are of lower cost and flow will shift evenly between the two paths. Finally, if flow is placed on paths π_1 and π_3 and cost equalised $c_{\pi_1} = c_{\pi_3}$, then π_2 would be a cheaper option than π_3, as the cost of the edges not common to both paths for π_3, x_{12} and x_{23} are clearly more expensive than the empty edge x_{13} for path π_2. Therefore the set of paths that satisfy Wardrop's first principle (user equilibrium), i.e. there is not an empty path of lower cost available, is all three paths which must have equal cost, $c_{\pi_1} = c_{\pi_2} = c_{\pi_3}$.

To find the equilibrium for the second case the path-edge equations can be utilised, namely the flow on a given edge is the sum of the flow on the paths that utilise the edge.

$$x_{12} = f_{\pi_1} + f_{\pi_3} \tag{1.2a}$$

$$x_{13} = f_{\pi_2} \tag{1.2b}$$

$$x_{23} = f_{\pi_3} \tag{1.2c}$$

$$x_{24} = f_{\pi_1} \tag{1.2d}$$

$$x_{34} = f_{\pi_2} + f_{\pi_3}. \tag{1.2e}$$

Substituting (1.2) into the paths costs c_{π_1}, c_{π_2} and c_{π_3} and taking into account the demand equation $f_{\pi_1} + f_{\pi_2} + f_{\pi_3} = 6$, c_{π_3} the following system of equations is obtained,

$$c_{\pi_1} = 11f_{\pi_1} + 10f_{\pi_3} + 50 \tag{1.3a}$$

$$c_{\pi_2} = 11f_{\pi_2} + 10f_{\pi_3} + 50 \tag{1.3b}$$

$$c_{\pi_3} = 10f_{\pi_1} + 10f_{\pi_2} + 21f_{\pi_3} + 10 \tag{1.3c}$$

$$6 = f_{\pi_1} + f_{\pi_2} + f_{\pi_3}, \tag{1.3d}$$

which must also satisfy the equilibrium condition,

$$c_{\pi_1} = c_{\pi_2} = c_{\pi_3}.$$

Clearly equating c_{π_1} (1.3a) and c_{π_2} (1.3b) yields $f_{\pi_1} = f_{\pi_2}$. Finally by equating $c_{\pi_1} = c_{\pi_3}$ and some algebraic manipulation results in the equilibrium path solution $\pi_1 = \pi_2 = \pi_3 = 2$ with associated costs $c_{\pi_1} = c_{\pi_2} = c_{\pi_3} = 92$. Hence the cost per user has risen from 83 to 92.

This illustrates the aforementioned idea that the introduction of an edge can result in all users being worse off.

1.2.4 Congestion Games

The class of games known as congestion games was first introduced by Rosenthal [141]; in the same year he also published a paper on network equilibrium expressed in integers [142], which is a generalisation of the user equilibrium problem. A congestion game consists of a set of players, a set of congestible facilities and a set of strategy profiles determining how players choose a set of facilities. The cost of each facility is dependent on the number of players choosing the facility and the payoff to each player is then the sum of the costs of their respective choices of facilities. Rosenthal showed that finite unweighted congestion games emit a potential function (Section 1.2.5), which under best response dynamics (Section 1.2.4) converges to a Nash equilibrium and this result was extended to the continuous case by Devarajan [51] in which the number of players tends to infinity. An unweighted congestion game can be defined as follows, for further details see [127].

Definition 1.2.9 (Unweighted congestion game)**.** *An unweighted congestion game is a tuple* $(N, \mathcal{E}, \mathcal{A}, l)$ *where:*

- N *- The finite set of n players;*
- $\mathcal{E}$ *- The set of congestible elements;*
- $\mathcal{A}$ *- The set of pure strategy profiles (strategy space)* $\mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_n$*, where* $\mathcal{A}_i$ *is the set of strategies available to player* i *and a given strategy for player* i*,* $a_i \in \mathcal{A}_i$ *is a set of congestible elements* $a_i \subseteq \mathcal{E}$*. A pure strategy* a *is then a vector of player strategies* $a = (a_1, \ldots, a_n)$*;*

- *l - The vector of cost functions $l = (l_e)_{e \in \mathcal{E}}$, where $l_e : \mathbb{N} \mapsto \mathbb{R}$ is the cost (latency) for congestible element (facility) e subject to the number of players x_e choosing e.*

  For a given strategy profile a, $x_e = |X|$, where $X = \{i : e \in a_i : a_i \in a\}$. A players cost is then given by $\lambda_j = \sum_{e \in a_i} l_e(x_e)$.

Note that the size of strategies available to player i is bounded by $2^{|E|}$ (either player i chooses facility j or not) and can make the problem intractable. Studies into restricted classes of congestion games such as linearly independent paths [64] and single resource strategies [79] demonstrate tractability but are limited by the scope of problems that can be modelled.

1.2.4.1 Existence of Nash Equilibria

Rosenthal [141] showed that an unweighted congestion game has at least one pure strategy Nash equilibrium which can be found by minimising the function $\Phi = \sum_{e \in \mathcal{E}} \sum_{i=1}^{x_e} l_e(x_e)$. Theorem 1.2.1 is integral to providing bounds on the price of anarchy and price of stability. i.e. it guarantees the existence of an equilibrium solution. Another key consequence is that it exposes the notion of the potential function as a theoretical and computation means to finding an equilibrium solution. Section 1.2.5 details the broader area of potential games that arose from Rosenthal's work.

Theorem 1.2.1. *(Rosenthal [141]) Unweighted congestion games possess at least one pure-strategy Nash equilibrium.*

Proof. Consider the outcome of a player's payoff under best response dynamics from a strategy a_i to a_i' to the cost of the facilities $e \in \mathcal{E}$. Denote the corresponding strategy profiles a and a'. Edges common to strategies are unaffected, elements not in either strategy are unaffected, elements $e \in a_i \setminus a_i'$ decrease their cost to $l_e(x_e - 1)$ and elements $e \in a_i' \setminus a_i$ increase their cost to $l_e(x_e + 1)$. The effect to the cost of the player is therefore $\sum_{e \in a_i' \setminus a_i} l_e(x_e + 1) - \sum_{e \in a_i \setminus a_i'} l_e(x_e)$, however under best response dynamics $\sum_{e \in a_i' \setminus a_i} l_e(x_e + 1) - \sum_{e \in a_i \setminus a_i'} l_e(x_e) < 0$.

Consider the function $\Phi = \sum_{e \in E} \sum_{i=1}^{x_e} l_e(x_e)$, then the change in Φ for strategy

profile a to a' is,

$$\begin{aligned}\Phi(a) - \Phi(a') &= \left(\sum_{e \in a'_i \setminus a_i} \sum_{i=1}^{x_e} l_e(x_e) + \sum_{e \in a_i \setminus a'_i} \sum_{i=1}^{x_e} l_e(x_e) \right) \\ &\quad - \left(\sum_{e \in a'_i \setminus a_i} \sum_{i=1}^{x_e+1} l_e(x_e) + \sum_{e \in a_i \setminus a'_i} \sum_{i=1}^{x_e-1} l_e(x_e) \right) \\ &= \sum_{e \in a'_i \setminus a_i} l_e(x_e + 1) - \sum_{e \in a_i \setminus a'_i} l_e(x_e),\end{aligned}$$

which is precisely the change to the cost of the player. i.e. Φ exactly maps the changes under best response dynamics.

Thus a Nash equilibrium can be found by the strategy a which minimises Φ. □

1.2.5 Potential Games

Potential games are a class of games introduced by Monderer and Shapley [112] that generalises the work introduced by Rosenthal on congestion games which exhibit a number of useful properties for studying the existence and computation of Nash equilibria. In essence they map players' individual strategy changes to maximise their utility u_i to a real-valued function (potential function) based on the current strategy profile (all player strategies) of the game $\Phi : S \mapsto \mathbb{R}$. If a player i moves to increase their utility u_i, the potential function Φ increases. The potential function derives its name from the notion of potential energy and can be seen as a similar mechanism for finding states of equilibrium, e.g. the potential energy of a mechanical system is at a local/global optimum. An extension to the class of potential games was given by Sandholm [148], who defined the notion of continuous player sets. The traffic assignment problem and nonatomic selfish routing games belong to this class of potential game, whereby the potential function is the objective (1.5a) and the players are the infinitesimal flow distributed across the paths. Numerous other problems within communications and networking can be modelled as a potential game, with applications in resource allocation, power control and interference avoidance. For a detailed summary of applications see [92].

Monderer and Shapley originally proposed four categories of potential games: exact; weighted; ordinal; generalised ordinal (not considered below). A hierarchy of potential games is given in Figure 1.5.

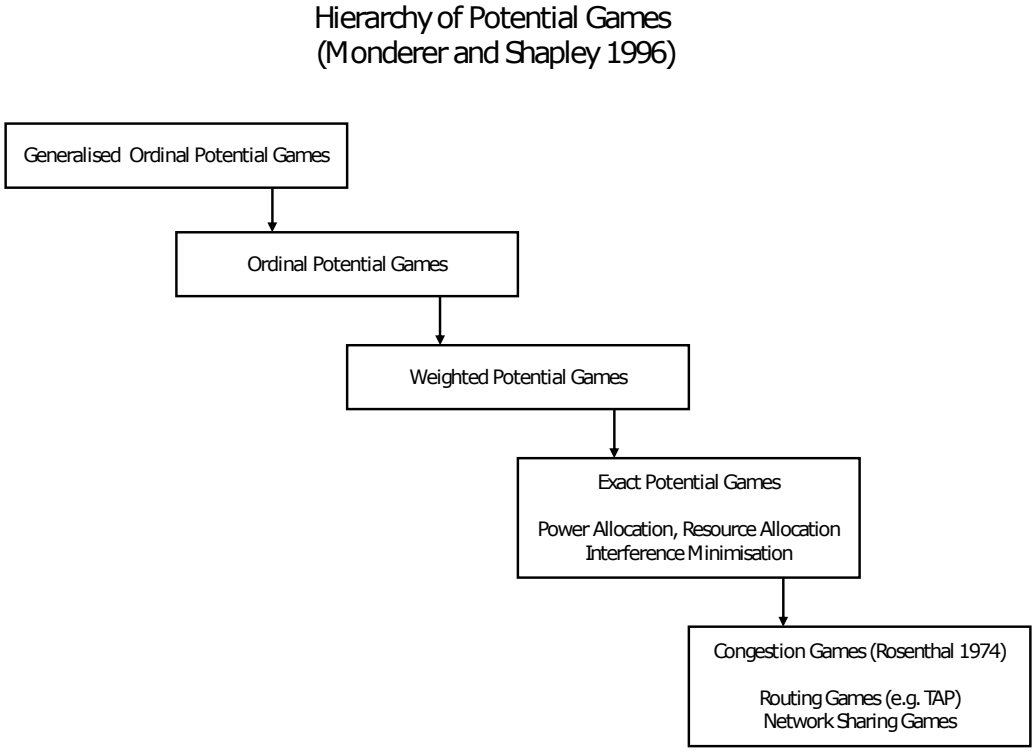


Figure 1.5: A hierarchy of potential games.

1.2.5.1 Potential Game Categories

The following categories are defined for a finite game (N, S, u), for further details see [92]. Definitions for continuous games, games with continuous real strategy sets and continuous and differentiable utility functions are also given by Monderer and Shapley [112].

Definition 1.2.10. *(Exact Potential Game) The game (N, S, u) is an exact potential game (EPG) iff a function $\Phi : S \mapsto \mathbb{R}$ exists such that for strategies $s = (s_i, s_{-i})$ and $s' = (s'_i, s_{-i})$,*

$$u_i(s') - u_i(s) = \Phi(s') - \Phi(s), \quad \forall s_i, s'_i \in S_i, \forall s_{-i} \in S_{-i}, \forall i \in N.$$

i.e. the change in a player's utility by that player switching to an alternative strategy results in an equal change to the potential function.

Definition 1.2.11. *(Weighted Potential Game) The game (N, S, u) is an weighted potential game (WPG) iff a function $\Phi : S \mapsto \mathbb{R}$ exists such that for strategies $s = (s_i, s_{-i})$, $s' = (s'_i, s_{-i})$ and positive weight w_i,*

$$u_i(s') - u_i(s) = w_i(\Phi(s') - \Phi(s)), \quad \forall s_i, s'_i \in S_i, \forall s_{-i} \in S_{-i}, \forall i \in N.$$

i.e. the change in a player's utility by that player switching to an alternative strategy results in an equal weighted change to the potential function.

Note that an exact potential game is a weighted potential game with weights all equal to 1.

Definition 1.2.12. *(Ordinal Potential Game) The game (N, S, u) is an ordinal potential game (OPG) iff a function $\Phi : S \mapsto \mathbb{R}$ exists such that for strategies $s = (s_i, s_{-i})$ and $s' = (s'_i, s_{-i})$,*

$$u_i(s') - u_i(s) > 0 \iff \Phi(s') - \Phi(s) > 0, \quad \forall s_i, s'_i \in S_i, \forall s_{-i} \in S_{-i}, \forall i \in N.$$

i.e. the change in a player's utility by that player switching to an alternative strategy results in a change to the potential function of equal sign and vice versa. Owing to this equivalence, clearly a WPG is also an OPG. Hence the following hierarchy is established: every EPG is a WPG which is a OPG. Thus any properties of OPGs apply to both EPGs and WPGs.

1.2.5.2 Existence of Pure Strategy Nash Equilibrium

One of the most fundamental results of ordinal potential games is that they have at least one pure strategy Nash equilibrium.

Theorem 1.2.2. *(Monderer and Shapley [112]) The set of strategy profiles that result in a value of Φ that cannot be improved by any player switching to another strategy are pure strategy Nash equilibrium of the ordinal potential game.*

Proof. Let $s = (s_i, s_{-i})$ $s' = (s'_i, s_{-i})$ be strategy profiles whereby $\Phi(s') - \Phi(s) > 0, \quad \forall s_i, s'_i \in S_i; \forall s_{-i} \in S_{-i}; \forall i \in N$, i.e. there is no unilateral deviation by any player that improves Φ. By (1.2.12), $\forall i \in N$, any move by a player $i \in N$ would result in a decrease in the utility u_i. Hence s is stable. □

Corollary 1.2.2.1. *(Monderer and Shapley [112]) Every ordinal potential game has at least one pure strategy Nash equilibrium.*

Proof. Let s be a strategy profile that maximises Φ. Thus any move by any player $i \in N$ would result in a decrease in Φ and by (1.2.12) would result in a decrease in the utility u_i. Therefore $\forall i \in N$ any move by a player i results in a decrease in their utility i and hence s is stable. □

1.2.5.3 Best Response Dynamics

Best response dynamics utilises the concept of a player's best response, but moves are made sequentially. That is, players take turns to make a best response to the current state of the game. It was shown by Monderer and Shapley [112] that finite ordinal potential games (those with a finite strategy set) have a finite improvement property, that is, any improvement path, a sequence of strategy profiles generated by each player (in turn) making a best response move to improve their utility and terminating if no player can improve their utility, is finite. Such a sequence of moves $(s_0, s_1, s_2, \ldots, s_k)$ must terminate in a finite ordinal potential game as $\Phi(s_0) < \Phi(s_1) < \Phi(s_2) < \cdots < \Phi(s_k)$ and the strategy profile set S is finite and thus Φ is bounded. Moreover, a finite improvement path must terminate at a Nash equilibrium, as otherwise this would lead to a contradiction as the path would have terminated with a best response move available to a player. The finite improvement property and the existence of at least one pure strategy Nash equilibrium provide a powerful technique for analysing Nash equilibria and bounding the price of stability in these types of games and is the main rationale for the attention they have received.

1.2.5.4 Bounds on the Price of Stability

The potential function of an ordinal potential game can be used to establish an upper bound on the price of stability and is implicit in [5].

Theorem 1.2.3. *For an ordinal potential game with potential function Φ, assume that for a strategy profile s' that maximises Φ and strategy profile s^* that maximises the social cost C and some constants $A, B > 0$,*

$$A \cdot C(s') = \Phi(s')$$

and,

$$\frac{C(s^*)}{B} = \Phi(s^*).$$

Then the price of stability is at most AB.

Proof. As s' is the strategy profile maximising Φ, then s' is a Nash equilibrium. Thus,

$$A \cdot C(s') = \Phi(s') \geq \Phi(s^*) = \frac{C(s^*)}{B}$$

and,

$$\frac{C(s')}{C(s^*)} \leq AB.$$

□

Note that this bound may not be tight as the equilibrium found my maximising the potential function may not be the best equilibrium (i.e. the one with minimum cost). For the Global Connection Game, a cost sharing congestion game where every congestible element has a cost $c_e(x_e) = \frac{k_e}{x_e}$ (users share the fixed cost k_e, as opposed to contributing to it) the upper bound on the price of stability is $\mathcal{H}_k$ (the kth harmonic number) generated by the constants 1 and $\mathcal{H}_k$ [5].

1.2.5.5 PLS-Completeness

Best Response Dynamics provides an algorithmic approach to finding pure strategy Nash equilibria in potential games; however, the computation of these equilibria need not necessarily be done in reasonable time. The class of PLS problems, introduced by Johnson, Papadimitriou and Yannakakis [85], is a complexity class in which the problem of finding a local optimum is hard, but a local optimum can be verified within its neighbourhood in polynomial time. Fabrikant, Papadimitriou, Talwar [60] showed that ordinal potential games (referred to as general potential games) are PLS-complete.

The first problem to be shown to be PLS-complete by Johnson et al [85] was the maximum satisfiability problem (MAX-SAT). Krentel [90] showed that the weighted MAX-SAT problem is also PLS-complete and it is through a reduction to weighted MAX-SAT by which the computation of Nash equilibria in potential games is shown to be PLS-complete [60].

Whilst the nature of the PLS-complete class makes designing algorithms difficult, it does however motivate the study of local search methods, such as metaheuristics, for generating Nash equilibria in potential games.

1.3 Nonatomic Selfish Routing

The notation used to define both the nonatomic selfish routing and unweighted atomic selfish routing models is consistent with the literature, but uses some addi-

tional set notation that is based on the notation used by Roughgarden in [147].

Given a network $G = (V, E)$ and a set of commodities K of k OD pairs $\{(r_1, s_1), \ldots, (r_k, s_k)\}$. For each commodity (r_i, s_i) there is an amount of traffic d_i to be routed between r_i and s_i, the set of used paths by commodity (r_i, s_i) is denoted by Π_i. The set of all paths used for all commodities is then $\Pi = \bigcup_i \Pi_i$. Let f_π to be the amount of flow routed along path π and x_e to be the amount of flow induced onto edge e, $x_e = \sum_{\pi \in \Pi : e \in \pi} f_\pi$. The cost of an edge $t_e(x_e)$ is a function of the flow on the edge e, which is the aggregated flow from all paths that use edge e. A feasible flow $\mathbf{f}$ induces a flow $\mathbf{x} = (x_e)_{e \in E}$ and therefore the cost of a path π is $c_\pi = c_\pi(\mathbf{f}) = \sum_{e \in \pi} t_e(x_e)$.

Defining the shortest path and its cost between commodity (r_i, s_i) as π_i^* and $c_{\pi_i^*}$, respectively, then the principle of user equilibrium is succinctly stated as the nonlinear complementarity problem [62],

$$0 \leq c_\pi - c_{\pi_j^*} \perp f_\pi \geq 0 \quad \forall \pi \in \Pi_j, \forall j \in \{1, \ldots, k\}, \tag{1.4}$$

that is, either the cost of a path $\pi \in \Pi_j$ between a given commodity (r_j, s_j) is equal to the shortest path π_j^* for commodity (r_j, s_j) or there is no flow on the path.

An instance of a nonatomic selfish routing game is given by the triple (G, d, t) where $d = (d_j)_{j \in \{1, \ldots, k\}}$ is a vector of demands and $t = (t_e)_{e \in E}$ is a vector of edge travel time functions (an example was given in Section 1.2.3). The above definition therefore makes the assumption that commodities are homogeneous, i.e. there is a single demand d_i to be routed for a given OD pair (r_i, s_i), and that the flows on edges x_e and paths f_π can be fractional.

1.3.1 Path-based Model

The path-based model is motivated by the potential combinatorial explosion of the supply/demand constraints of the link-based formulation given in Section 1.3.5. The decision variables are the paths used by the commodities and the number of constraints and variables can be managed through column-generation methods (see Section 5.3.2). The path-based model can be stated as the following optimisation problem [135]:

$$\text{minimise} \qquad U(\mathbf{x}) = \sum_{e \in E} \int_0^{x_e} t_e(\omega) d\omega \tag{1.5a}$$

subject to

$$\sum_{\pi \in \Pi_j} f_\pi = d_j, \qquad \forall j \in \{1, \ldots, k\} \tag{1.5b}$$

$$\sum_{\pi \in \Pi : e \in \pi} f_\pi = x_e, \qquad \forall e \in E \tag{1.5c}$$

$$f_\pi \geq 0, \qquad \forall \pi \in \Pi, \tag{1.5d}$$

subject to the three assumptions:

A1. The network is strongly connected.

A2. The traffic demand d_j is non-negative for all $j \in \{1, \ldots, k\}$.

A3. The travel time function $t_e : [0, \infty) \to [0, \infty)$ are non-negative, continuous and non-decreasing. N.B. Functions of this type will be referred to as a congestible function for the remainder of this thesis.

To find a solution to Wardrop's second principle one can replace the objective function $U(\mathbf{x})$ with the objective function that minimises the total travel time, resulting in the following optimisation problem adhering to assumptions A1, A2 and A3.

$$\text{minimise} \qquad T(\mathbf{x}) = \sum_{e \in E} x_e t_e(x_e) \tag{1.6a}$$

subject to

$$\sum_{\pi \in \Pi_j} f_\pi = d_j, \qquad \forall j \in \{1, \ldots, k\} \tag{1.6b}$$

$$\sum_{\pi \in \Pi : e \in \pi} f_\pi = x_e, \qquad \forall e \in E \tag{1.6c}$$

$$f_\pi \geq 0, \qquad \forall \pi \in \Pi. \tag{1.6d}$$

For the optimisation problems (1.5) and (1.6), assumptions A1 and A2 ensure that every pair of vertices can be a commodity and that a positive amount of traffic is routed. Assumption A3 ensures the uniqueness of the solution in terms of edge flows (see Section 1.3.3). It is common in the literature for a weaker variant of A3 to

be used which guarantees uniqueness for the equilibrium problem, but not system optimal problem.

Whilst unique in terms of edge flows, the path decomposition for edges $x_e = \sum_{\pi \in \Pi : e \in \pi} f_\pi$ is generally not unique. In fact for most instances of the path-based model the set of all possible paths for all commodities is much greater than the set of edges E and would result in an underdetermined system of linear equations with a non-empty null space for the set of paths.

1.3.2 Equivalence of the Optimisation Problem (1.5) to User Equilibrium

It can be shown that the optimal solution to the problem given by (1.5) is equivalent to user equilibrium, i.e. it solves Wardrop's first principle (Section 1.2.2.2) and the nonlinear complementarity problem (1.4). A proof via the Karush–Kuhn–Tucker (KKT) conditions is given and further details can be found in [152].

Theorem 1.3.1. *Let $\mathbf{x}^*$ be an optimal solution to the optimisation problem (1.5), then any network flow $\mathbf{f}$ which induces the edge flow $\mathbf{x}^*$ is a solution to the nonlinear complementarity problem (1.4), i.e. it is equivalent to user equilibrium.*

Proof. Let $\mathbf{x}^*(\mathbf{f})$ be an optimal solution to (1.5) induced by a flow $\mathbf{f}$. The Lagrangian of the optimisation problem is given by,

$$\mathcal{L}(\mathbf{f}, \lambda) = U(\mathbf{x}^*(\mathbf{f})) + \sum_{j=1}^{k} \lambda_j \left(d_j - \sum_{\pi \in \Pi_j} f_\pi \right), \tag{1.7}$$

and given that,

$$\frac{\partial x_e}{\partial f_\pi} = \frac{\partial}{\partial f_\pi} \sum_{\pi \in \Pi : e \in \pi} f_\pi = \begin{cases} 1 & \text{if } e \in \pi \\ 0 & \text{if } e \notin \pi. \end{cases}$$

Let δ_e^π be an indicator variable that represents whether edge e is on path π, i.e. $\delta_e^\pi = 1$ represents that edge e is on path the π, then by the chain rule,

$$\frac{\partial U(\mathbf{x}^*(\mathbf{f}))}{\partial f_\pi} = \sum_{e \in E} \frac{\partial U}{\partial x_e} \frac{\partial x_e}{\partial f_\pi} = \sum_{e \in E} t_e(x_e) \delta_e^\pi = \sum_{e \in \pi} t_e(x_e) = c_\pi(\mathbf{f}),$$

the resulting KKT conditions are:

$$f_\pi \frac{\partial \mathcal{L}(\mathbf{f}, \lambda)}{\partial f_\pi} = 0 \implies f_\pi(c_\pi(\mathbf{f}) - \lambda_j) = 0 \qquad \forall \pi \in \Pi_j,\ \forall j \in \{1, \ldots, k\} \tag{1.8}$$

$$\frac{\partial \mathcal{L}(\mathbf{f}, \lambda)}{\partial f_\pi} \geq 0 \implies c_\pi(\mathbf{f}) - \lambda_j \geq 0 \qquad \forall \pi \in \Pi_j,\ \forall j \in \{1, \ldots, k\} \tag{1.9}$$

$$\frac{\partial \mathcal{L}(\mathbf{f}, \lambda)}{\partial \lambda_j} = 0 \implies \sum_{\pi \in \Pi_j} f_\pi = d_j \qquad \forall j \in \{1, \ldots, k\} \tag{1.10}$$

$$f_\pi \geq 0 \qquad \forall \pi \in \Pi_j,\ \forall j \in \{1, \ldots, k\}. \tag{1.11}$$

The KKT conditions (1.10) and (1.11) are just the feasibility constraints of the program (1.5) . Replacing the constant λ_j with the shortest path for commodity j, $c_{\pi_j^*}$, then equations (1.8) and (1.9) are equivalent to the nonlinear complementary problem, that is either the cost of a path $\pi \in \Pi_j$ is greater than the shortest path for commodity j or the flow on the path is zero (the path is empty). □

1.3.3 Uniqueness of the User Equilbrium Solution of (1.5)

Much like the equivalence of the previous section, it can also be shown that the optimal solution to the problem given by (1.5) is unique in terms of flows on the edges. Further details can be found in [152].

Theorem 1.3.2. *An optimal solution* $\mathbf{x}^*$ *to* (1.5) *is unique in terms of the edges flows* $\mathbf{x}$*.*

Proof. The feasible region, defined by the linear equality constraints (1.5b) and non-negativity constraints (1.5d), is clearly closed and convex.

The first and second derivatives of the objective function (1.5a) with respect to the edge flows x_e are given by,

$$\frac{\partial U(\mathbf{x})}{\partial x_e} = t_e(x_e) \tag{1.12}$$

$$\frac{\partial^2 U(\mathbf{x})}{\partial x_e \partial x_a} = \frac{\partial t_e(x_e)}{\partial x_a} = \begin{cases} \frac{dt_e(x_e)}{dx_e} & \text{if } e = a \\ 0 & \text{if otherwise.} \end{cases} \tag{1.13}$$

The Hessian $\nabla^2 U(\mathbf{x})$ is positive definite and the objective function is strictly convex. As the feasible region is closed and convex, the optimal solution $\mathbf{x}^*$ is unique. □

1.3.4 Marginal Cost Taxation

An important result relating to nonatomic selfish routing games is that via a small change to the edge cost functions, the user equilibrium solution to the changed network is also the socially optimal solution of the original network. This allows methods that solve the user equilibrium problem to be used to solve the socially optimal problem. The following proof is given, for further details see [135].

Theorem 1.3.3. *Let (G, d, t) be a nonatomic selfish routing instance and $(G, d, \tilde{t})$ be another instance with marginal cost functions $\tilde{t_e}(x_e) = t_e(x_e) + x_e t'_e(x_e)$. A solution $\mathbf{x}^*$ is a solution to system optimal for the instance (G, d, t) if and only if it is a solution to user equilibrium for $(G, d, \tilde{t})$.*

Proof. The optimisation problem for system optimal is defined by (1.6).

For an instance (G, d, t) for which the system optimal solution is sought, the Lagrangian is given by,

$$\mathcal{L}(\mathbf{f}, \lambda) = T(\mathbf{x}(\mathbf{f})) + \sum_{j=1}^{k} \lambda_j \left(d_j - \sum_{\pi \in \Pi_j} f_\pi \right), \tag{1.14}$$

and noting that,

$$\frac{\partial T(\mathbf{x}(\mathbf{f}))}{\partial f_\pi} = \sum_{e \in E} \frac{\partial T(\mathbf{x})}{\partial x_e} \frac{\partial x_e}{\partial f_\pi} = \sum_{e \in E} (t_e(x_e) + x_e t'_e(x_e)) \delta_e^\pi = \sum_{e \in \pi} (t_e(x_e) + x_e t'_e(x_e)).$$

Letting $\tilde{t}_e(x_e) = t_e(x_e) + x_e t'_e(x_e)$ and $\tilde{c}_\pi(\mathbf{f}) = \sum_{e \in \pi} \tilde{t}_e(x_e)$, then the resulting KKT conditions are:

$$f_\pi \frac{\partial \mathcal{L}(\mathbf{f}, \lambda)}{\partial f_\pi} = 0 \implies f_\pi(\tilde{c}_\pi(\mathbf{f}) - \lambda_j) = 0 \qquad \forall \pi \in \Pi_j,\ \forall j \in \{1, \ldots, k\} \tag{1.15}$$

$$\frac{\partial \mathcal{L}(\mathbf{f}, \lambda)}{\partial f_\pi} \geq 0 \implies \tilde{c}_\pi(\mathbf{f}) - \lambda_j \geq 0 \qquad \forall \pi \in \Pi_j,\ \forall j \in \{1, \ldots, k\} \tag{1.16}$$

$$\frac{\partial \mathcal{L}(\mathbf{f}, \lambda)}{\partial \lambda_j} = 0 \implies \sum_{\pi \in \Pi_j} f_\pi = d_j \qquad \forall j \in \{1, \ldots, k\} \tag{1.17}$$

$$f_\pi \geq 0 \qquad \forall \pi \in \Pi_j,\ \forall j \in \{1, \ldots, k\}. \tag{1.18}$$

Thus $\tilde{t}_e(x_e)$ can be seen as the marginal cost incurred by all flow using the edge due to a marginal increase in flow [152].

Let $\mathbf{f}^*$ be an optimal flow that induces a set of optimal edge flows $\mathbf{x}^*$ for system optimal for the instance (G, d, t), then clearly $\mathbf{f}^*$ satisfies (1.15) and (1.16). Replacing

λ_j with the shortest path for commodity j in $(G, d, \tilde{t})$, $\tilde{c}_{\pi_j^*}$, the user equilibrium conditions are recovered for $(G, d, \tilde{t})$ for the flow $\mathbf{f}^*$ and edge flows $\mathbf{x}^*$. $\mathbf{f}^*$ is also a feasible flow for $(G, d, \tilde{t})$ as the feasibility is not affected by the change to the edge cost functions t to $\tilde{t}$.

Let $\hat{\mathbf{f}}$ be an optimal flow that induces an optimal set of edge flows $\hat{\mathbf{x}}$ for user equilibrium for $(G, d, \tilde{t})$. Clearly, as above, $\hat{\mathbf{f}}$ is also a feasible flow for (G, d, t). For every commodity j, every path $\pi \in \Pi_j$ if used should be equal to the shortest path cost $\tilde{c}_{\pi_j^*} \implies c_\pi(\hat{\mathbf{f}}) - \tilde{c}_{\pi_j^*}(\hat{\mathbf{f}}) = 0$ and $f_\pi > 0$. Any unused path should be equal or cost more than the shortest path and have zero flow $\implies c_\pi(\hat{\mathbf{f}}) - \tilde{c}_{\pi_j^*}(\hat{\mathbf{f}}) \geq 0$ and $f_\pi = 0$. Let $\tilde{c}_{\pi_j^*}$ replace $\lambda_j, \forall j \in \{1, \ldots, k\}$, then $\hat{\mathbf{f}}$ simultaneously satisfies (1.15) and (1.16) □

It is interesting to note that by considering the instance $(G, r, \tilde{t})$, the objective function (1.5a) for (1.5) follows trivially,

$$\tilde{t}_e(x_e) = \frac{d}{dx_e}(x_e t_e(x_e)) \tag{1.19}$$

$$\int_0^{x_e} \tilde{t}_e(\omega) d\omega = x_e t_e(x_e) \tag{1.20}$$

$$\sum_{e \in E} \int_0^{x_e} \tilde{t}_e(\omega) d\omega = \sum_{e \in E} x_e t_e(x_e). \tag{1.21}$$

The connection between the two instances (G, d, t) and $(G, d, \tilde{t})$ provide a simple means of marginal cost taxation applied to the edges. Tax each edge $x_e t'_e(x_e)$, edges belonging to (G, d, t) now cost $\tilde{t}_e(x_e)$ and thus a state of user equilibrium under the prescribed taxation rule results in system optimal for the instance (G, d, t).

The consequence of this should not be understated as it allows any method developed to solve (1.5) to also solve (1.6)

1.3.5 Multi-commodity Flow Model

Network equilibrium can be modelled as a multi-commodity flow problem (MCFP) whereby the cost function on an edge t_e adheres to the restrictions placed upon it in Section 1.3.1.

The decision variables of the optimisation problem are the flow on each of the edges. It is assumed that the flow that is routed between a commodity is homogenous and fractional. A consequence of this is that it can be split across multiple paths

between commodities source and sink. A major issue with the formulation is that it suffers with an combinatorial explosion of supply/demand constraints as the size of the network increases.

Let K be the set of k commodities $K = \{(r_1, s_1), (r_2, s_2), \dots (r_k, s_k)\}$ which each route a demand of d_k. Let E_i^+ represent the set of outgoing edges of vertex i and E_i^- the set of incoming edges of vertex i. The multi-commodity flow model is then given as:

$$\text{minimise} \qquad U(\mathbf{x}) = \sum_{e \in E} \int_0^{x_e} t_e(\omega) d\omega \tag{1.22a}$$

subject to

$$\sum_{e \in E_i^-} x_e^j - \sum_{e \in E_i^+} x_e^j = \begin{cases} d_j & i = r_j \\ -d_j & i = s_j \\ 0 & \text{otherwise} \end{cases}, \qquad \forall i \in V \tag{1.22b}$$

$$x_e = \sum_{j=1}^{k} x_e^j, \qquad \forall e \in E \tag{1.22c}$$

$$x_e^j \geq 0, \qquad \forall e \in E, \forall j \in \{1, \dots, k\}. \tag{1.22d}$$

The supply and demand balance constraints given by (1.22b) ensure that for a given vertex the incoming flow and outgoing flow is appropriately balanced with the demand leaving or entering vertex. The bundle constraints (1.22c) ensure that the total flow on an edge is the sum of all induced flows by all commodities on edge e.

1.4 Atomic Selfish Routing

An atomic selfish routing game is defined as follows. Let G be a directed graph $G = (V, E)$ and for each player $i \in \mathcal{N}$ associate a commodity (s_i, r_i) and positive traffic demand d_i. Let $t_e(x_e)$ be the congestible cost function for edge e that are non-negative, continuous and non-decreasing. A player's strategy set Π_i is the set of possible paths from source to sink, i.e. a strategy $\pi \in \Pi_i$ is a path consisting

of edges $e \in E$. Therefore the cost to a given player choosing a particular path is dependent on the number of players choosing paths which share edges in the graph.

An atomic routing game is given by the triple (G, d, t), where $d = (d_i)_{i \in \mathcal{N}} > 0$ and $t = (t_e)_{e \in E}$.

The difference between atomic and nonatomic selfish routing is that whereas a commodity in nonatomic routing games represents many agents (users) controlling a negligible (fractional) amount of traffic, a commodity in an atomic instance represents an agent who controls a non-negligible amount of traffic [127]. Loosely they can be seen to represent the continuous and discrete routing cases.

1.4.1 Weighted Atomic Selfish Routing Game

A weighted atomic selfish routing game is an atomic selfish routing game where players do not control an equal amount of traffic demand d_i. Awerbuch et al. (2005) provided an example that showed that equilibrium is not guaranteed in such instances [7].

1.4.2 Unweighted Atomic Selfish Routing Game

An unweighted atomic selfish routing game is an atomic routing game given by (G, d, t) where each player routes an equal demand $d_i = D$ [146]; however, only the case when a player routes him or herself is considered, i.e. $d_i = 1,\ \forall i \in \mathcal{N}$.

1.4.2.1 Existence of Nash Equilibrium

Theorem 1.4.1. *(Rosenthal) Unweighted atomic selfish routing games, $d_i = 1,\ \forall i \in \mathcal{N}$, possess at least one pure-strategy Nash equilibrium*

Proof. An instance of an unweighted atomic selfish routing game (G, d, t), $d_i = 1,\ \forall i \in \mathcal{N}$, is clearly an unweighted congestion game by the mapping:

- $N = \mathcal{N}$
- $\mathcal{E} = E$, the edges of the graph G.
- $\mathcal{A}$ - All feasible flows f over all paths $\Pi = \bigcup_i \Pi_i$, all paths for all commodities.

- $l = (t_e)_{e \in E}$, the cost of an edge in the graph subject to the flow on the edge.

Thus by Theorem 1.2.1 a Nash equilibrium can be found by the strategy a which minimises the potential function $\Phi = \sum_{e \in E} \sum_{i=1}^{x_e} t_e(x_e)$. □

1.4.3 Non-uniqueness of Nash Equilibrium

Whilst Theorem 1.4.1 guarantees at least one pure-strategy Nash equilibrium, it does not provide any guarantees that this is unique. In fact Rosenthal showed via a simple counter-example that this is indeed the case [142].

1.4.4 Path-based Model

The following optimisation problem captures the unweighted atomic selfish routing game where $d_i = 1,\ \forall i \in \mathcal{N}$.

$$\text{minimise} \qquad U(\mathbf{x}) = \sum_{e \in E} \sum_{i=0}^{x_e} t_e(i) \tag{1.23a}$$

subject to

$$\sum_{\pi \in \Pi_i} f_\pi = 1, \qquad \forall i \in \mathcal{N} \tag{1.23b}$$

$$\sum_{i=1}^{n} \sum_{\pi \in \Pi : e \in \pi} f_\pi = x_e, \qquad \forall e \in E \tag{1.23c}$$

$$f_\pi \geq 0, \qquad \forall \pi \in \Pi \tag{1.23d}$$

$$f_\pi \in \{0, 1\}, \qquad \forall \pi \in \Pi. \tag{1.23e}$$

Given the assumption that players are homogeneous and route 1 unit of flow, let i represent the group of players routing flow between (r_i, s_i). This can then be considered as commodity i and the subsequent demand d_i is the number of players attached to i. Let k be the number of commodities (groups), then the optimisation problem can be recast to mirror the optimisation problem given for the nonatomic selfish routing model.

$$\text{minimise} \qquad U(\mathbf{x}) = \sum_{e \in E} \sum_{i=0}^{x_e} t_e(i) \tag{1.24a}$$

subject to

$$\sum_{\pi \in \Pi_j} f_\pi = d_j, \qquad \forall j \in \{1, \ldots, k\} \tag{1.24b}$$

$$\sum_{\pi \in \Pi : e \in \pi} f_\pi = x_e, \qquad \forall e \in E \tag{1.24c}$$

$$f_\pi \geq 0, \qquad \forall \pi \in \Pi \tag{1.24d}$$

$$f_\pi \in \mathbb{Z}, \qquad \forall \pi \in \Pi. \tag{1.24e}$$

Note that the solution to this problem is the equilibrium found by the potential function (1.24a), but others may exist.

System optimal can be found by replacing the objective functions (1.23a) and (1.24a) by

$$T(\mathbf{x}) = \sum_{e \in E} x_e t_e(x_e). \tag{1.25}$$

1.4.5 Connection to Nonatomic Selfish Routing

By comparing the optimisation problems given by (1.5) and (1.24) it is clear to see that (1.24) is a discretisation of (1.5), that is, it is the discrete traffic assignment problem in which flow is discretised into integers, i.e. the corresponding Riemann sum over the domain $[0, x_e]$,

$$\sum_{e \in E} \int_0^{x_e} t_e(\omega) d\omega = \sum_{e \in E} \lim_{\Delta\omega \to 0} \sum_{i=1}^{n} t_e(\omega_i^*) \Delta\omega_i.$$

1.5 Multi-objective Optimisation

Many authors have discussed multi-objective selfish routing (traffic assignment) whereby users consider two or more objectives simultaneously. Two objectives were studied as early as Schneider (1968) [151] and Dial (1979) [54], albeit that these objectives are flow-independent. Two or more flow-dependent objectives were studied by Dafermos [42], Leurent [98, 99], Dial [55, 56]. Multiple objectives and classes of vehicles are considered by Nagurney [115], Nagurney and Dong [117], Yang and Huang [171], Han and Yang [74] and Sun et al [157]. For a further discussion on

the history see Raith [140]. A key cornerstone of this work is that the objectives are desirable when viewed from the perspective of the user and thus are optimised based on their preferences. In general, these have sought to minimise a weighted linear combination of the original criteria, converted into units known as value of time (VoT).

Although the weighted sum method is widely recognised as one of the key scalarisation techniques in multi-objective optimization (see, e.g. Ehrgott [59], Göpfert et al. [70] and Jahn [82], and the references therein), this method has its own limitations when modelling more complex network flow optimisation problems, where one considers competing objectives which road users are not necessarily motivated to optimise, such as fuel consumption. An alternative approach is to formulate a bi-level optimisation problem whereby an upper-level objective is to be optimised, but is subject to the optimisation of a set of lower-level objectives. For example, the upper-level objective could be the reduction of emissions and the lower-level objective would be to solve the traffic assignment problem under the current fixed set of variables for the upper-level problem. An example is the bi-level transportation network design problem with environmental considerations (BTPE) (see Szeto et al [158]).

Multi-objective optimisation problems naturally have numerous candidates for an optimal solution and in general the best that can be done is to classify the set of solutions that are Pareto optimal [59]. There are also the concepts of the ideal and nadir solutions, which are the theoretical bounds on the Pareto set and generally unattainable [59, 172, 173].

Methods for solving multi-objective optimisation problems are normally tailored to the problem at hand. As noted above, one approach is to combine the multi-objectives into a weighted single objective and attempt to solve via common single objective optimisation techniques. Another approach is to use a heuristic method, such as a meta-heuristic, e.g. an evolutionary algorithm, to find solutions to the problem, although the solutions found may be suboptimal and not represent the true Pareto set [48]. The multi-criteria optimisation in this thesis was done from a principle view of modelling options and proof of concepts, not to develop state of the art methods for solving multi-objective optimisation methods. Chapter 2 covers

single criteria methods which can then be used to solve multi-criteria problems using the weighted sum approach.

The price of anarchy and the price of stability are both defined for single objective optimisation problems, i.e. a utility function $C : S \mapsto \mathbb{R}$. Chen, Kedong, et al [37] provided a bound for the price of anarchy for a multi-class and multi-criteria traffic problem; however, this relied on the VoT and using a weighted sum to provide a single objective for both the equilibrium and system optimal cases. Assessing the suboptimality and efficiency of an equilibrium solution against the system optimal in a multi-criteria Pareto context has not been explored and requires careful consideration of the following: What is a socially optimal solution?; What is an equilibrium solution?; How are these compared to assess the degradation of the network?; How are they measured?

1.6 Importance Measures in Selfish Routing

Given the nature of selfish routing games and their representation via a network of edges and vertices, it is natural to consider how the changes to network infrastructure affect flow in the network.

The field of studying networks dates back to Euler's original work on the Seven Bridges of Königsberg which spawned the field of graph theory. More recently, owing to the prevalence of networks within the modern developed world, there has been an increasing desire and need to better understand the properties of networks applicable to such diverse fields as computer science, socio-economics, neuroscience and biology to name a few [100]. The field of network science attempts to statistically classify particular phenomena within networks, such as their topology and routing properties . Measures exist such as centrality, degree and clustering, which can give a means to ranking edges and vertices are a good insight into particular questions that arise that involve networks. Saxana and Iyengar provide a recent survey of network science measures [149].

In recent years, much research has been done in considering the importance of network components within the context of selfish routing, that is, what effect changes to the network infrastructure have on the flow? Within the transportation sector,

much emphasis has been given to ranking the network components (edges/vertices) by their importance to provide insight into how the removal of these components impacts the network's ability to perform the function of selfish routing.

The term network criticality, used to describe the ordering of the network components by their importance, was given by Jenelius, Petersen and Mattsson in 2006 [84], who proposed and applied three edge importance measures to road transportation networks in Sweden. There have been uses of other terms such as reliability, vulnerability and importance analysis [102]. Numerous metrics have been proposed and discussed and these mainly fall into three categories of functionality, assessing travel cost, connectivity and accessibility [81].

With respect to assessing the travel cost, Wang, Chan and Li [165] consider the unweighted case, which does not take into account the demand within the network itself. Demand weighted measures have been studied by Balijepalli and Oppong [11]; Dehghani, Flintsch and McNeil [49]; Du, Kishi, Aiura and Nakatsuji [58]; Gauthier, Furno and El Faouzi [66] and effectively weight the the travel cost by the associated demand on the network. Aydin, Duzgun, Wenzel and Heinimann [8]; Demirel, Kompil and Nemry [50]; Kermanshah and Derrible [86]; Wang and Cullinane [166] all studied a weighted betweenness centrality metric which measures the fraction of all-pairs shortest paths (based on travel cost) passing through a given vertex of the network.

The study of connectivity within selfish routing has been undertaken by Mishra, Welch and Jha [107] who consider the OD k-connectivity of a network which measures the decrease in the number of distinct OD paths. Baroud, Barker, Ramirez-Marquez and Rocco [16]; Qiang and Nagurney [138] looked at the unsatisfied demand which quantifies the amount of demand that cannot be routed due to the change in the network infrastructure.

Jafino, Kwakkel and Verbraeck (2019) [81] provide a survey and empirical comparison of the metrics and their performance on multimodal freight transport networks in Bangladesh as a case study, finding a high degree of correlation between many of the metrics and outlining a guideline for their selection.

1.7 Aims and Objectives

To condense the goal of this thesis into a single aim is a difficult task as, in the development of this work, a number of different but interrelating work on selfish routing games were undertaken. However, this thesis sets out:

1. To investigate and implement methods for the use in further analysis of nonatomic selfish routing concepts and atomic selfish routing concepts

2. To consider multi-criteria selfish routing and understand/extend the use of the Price of Anarchy/Stability in multi-criteria scenarios

3. To analyse the existing means of assessing the criticality of network components under equilibrium against the primary function, total travel cost.

Chapter 2

Network Equilibrium: Models and Methods

Chapter Preface

Algorithms that solve the nonatomic selfish routing problem have been of great interest for over 60 years because of their use in traffic planning. Several new methods have emerged which have allowed the problem to be solved in a much shorter timeframe for vastly large networks. Atomic selfish routing is also an active area of interest and can be viewed as a integer/discrete version of the problem.

This chapter introduces the models and methods that solve nonatomic and atomic selfish routing problems and those that are used in the remainder of this thesis for the purpose of analysis.

Section 2.3 is based on work from the paper by Bagdasar, Berry, O'Neill, Popovici and Raja [9] (Appendix A.1) and Section 2.4 is based on work from the paper by O'Neill, Bagdasar and Liotta [132] (Appendix A.2).

Chapter Keywords

Nonatomic Selfish Routing, Atomic Selfish Routing, Traffic Assignment, Link-based, Path-based, Bush-based, Dynamic Programming, Metaheuristic, Online Learning

2.1 Introduction

After Beckmann, McQuire and Winsten had presented their seminal ideas in *Studies in Economics of Transportation* [19], it took over a decade before algorithms were proposed by Almond [3], Bruynooghe et al [34] and Dafermos and Sparrow [43] to solve the Traffic Assignment Problem (Nonatomic Selfish Routing). The first algorithms which solved the problem on small test networks were given by Leblanc [95] and Nguygen [124, 123]. Leblanc also introduced the classic Sioux Falls network (see Appendix C.3) to the literature, which has been a starting test case for any proposed novel algorithms. Boyce [31] categorises algorithms into 3 broad categories, link-based, path-based and bush-based.

Link-based algorithms were first proposed by Almond in 1967 [3], who provided a method of averaging successive all-or-nothing assignments (see Definition 2.2.1). Leblanc [95] amended the Frank-Wolfe algorithm (convex combination algorithm) by replacing the step that considers the linear approximation of the objective function, with a shortest path problem. Mitradjieva and Lindberg [108] provided an updated Frank-Wolfe algorithm by insisting that the current search direction should be either conjugate or bi-conjugate to the previous search directions.

Path-based algorithms can be traced back to a working paper by Bothner and Lutter [27]; however, the first major appearances in the literature are Larsson and Patriksson [93] who proposed disaggregated simplicial decomposition and Jayakrishnan et al [83] who proposed a gradient projection method in 1994. Fifteen years later Florian et al [63] proposed an algorithm based on Rosen's projected gradient algorithm.

In the early 2000s two algorithms, Origin-Based Assignment (OBA) (Bar-Gera [13]) and Algorithm B (Dial [57]) were developed. Both algorithms reported excellent results when solving the Chicago Regional network [161], consisting of 1,790 commodities, 12,982 vertices, 39,018 edges and a total demand of 1,360,427, compared with link-based and path-based algorithms, Dial reported that Algorithm B reached a relative gap (defined in Section 2.2.1) of 10^{-4} in approximately 30 minutes as compared to OBA's 175. As of 2017, Dial reports solving the Chicago Regional network in just 18.5 minutes on a single core to a relative gap of 10^{-10} [52].

These two algorithms were also the genesis for Local User Cost Equilibrium

(Gentile [68]), QBA - a revised OBA which circumvents line search (Nie [126]) and Traffic Assignment by Paired Alternative Segments (TAPAS) (Bar-Gera [15]). For a more detailed history see Boyce [31].

Algorithms and methods for solving the Atomic Selfish Routing game are scarce, owing to their combinatorial nature and historical lack of practical application [88]. They do however share a large amount of structure with the MCFP (see Section 5.5), the major differences being that that the flows in the MCFP are splittable, that is, they do not require that the demand for a given commodity is routed on a single path and edge costs are fixed and do not depend on the flow on the edge [127, 2].

This chapter is organised as follows. In Section 2.2 algorithms for nonatomic selfish routing are reviewed and outlined. Section 2.3 presents dynamic programming, tabu search metaheuristic and piecewise linear approximation approaches developed to solve unweighted atomic selfish routing games. Finally Section 2.4 introduces a set of novel probabilistic online learning algorithms for the atomic selfish routing games based on the idea of bandit machines and semi-bandit and bandit feedback.

2.2 Algorithms for Nonatomic Selfish Routing

2.2.1 Convergence Measures

In the case of nonatomic selfish routing, it is important to establish the convergence measures required to determine that an algorithm has been successfully.

This section presents the common ones used in the literature [154] and settles on the use of the Average Excess Cost (AEC) for the remainder of the thesis.

Relative Gap (GAP)

The Relative Gap is defined as:

$$\text{GAP} = \frac{\sum_{e \in E} x_e t_e(x_e)}{\sum_{j \in \{1,\ldots,k\}} \lambda_j d_j} - 1,$$

where k is the number of commodities, λ_j is the shortest path for commodity j and d_j is the demand for commodity j.

The GAP is the ratio of the total travel time of the network and the cost of routing the demand d_j for each commodity j on the shortest path λ_j. As the network tends to equilibrium, all used paths for a commodity will tend to the shortest path cost λ_j. Thus, the GAP tends to 0.

Average Excess Cost

The Average Excess Cost is defined as:

$$\text{AEC} = \frac{\sum_{e \in E} x_e t_e(x_e) - \sum_{j \in \{1,\ldots,k\}} \lambda_j d_j}{\sum_{j \in \{1,\ldots,k\}} d_j},$$

where k is the number of commodities, λ_j is the shortest path for commodity j and d_j is the demand for commodity j.

The AEC is the excess cost experienced when the network is not at equilibrium. It is the difference between the total travel time and the cost of routing the demand d_j for each commodity j on the shortest path λ_j. Again, as the network tends to equilibrium, all used paths for a commodity will tend to the shortest path cost λ_j. Thus, the AEC tends to 0.

2.2.2 Link-based Algorithms

Link-based algorithms can be summarised as algorithms that maintain edge (link) flows $\mathbf{x}$ and iteratively attempt to move the edge flows closer to an equilibrium solution. This is generally done by using a convex combination of the current feasible edge flows $\mathbf{x}$ and a new feasible target set of edge flows $\mathbf{y}$.

Variants of link-based algorithms tend to differ on two courses of action. First on how the choice of target solution $\mathbf{y}$ is generated and second, how the current solution $\mathbf{x}$ and the target solution $\mathbf{y}$ are combined, i.e. the choice of weighting given for the convex combination. A full treatment is given by Patriksson [135].

A key step in the algorithms is the all-or-nothing assignment of network flow which is often used to obtain the target set of edge flows $\mathbf{y}$ and is defined as follows.

Definition 2.2.1 (All-or-nothing Assignment)**.** *Given a routing game (G, d, t), an all-or-nothing assignment assigns, for all commodities i, the entire demand d_i to the time-dependent shortest path π_i^*.*

2.2.2.1 General Iterative Scheme for a Link-based Algorithm

Link-based algorithms can be summarised by the following 5 steps.

1. **Initialisation**: Generate initial edge flows.

   - Generate $\mathbf{x^1} = (x_1^1, \ldots, x_{|E|}^1)$ via an all-or-nothing assignment for OD demands based on initial edge costs $\mathbf{t^0} = (t_1(0), \ldots, t_{|E|}(0))$.
   - Set $k = 1$.

   Note: An all-or-nothing assignment assigns all demand to the time-dependent shortest path.

2. **Update**: Update travel times for edges.

   - $\mathbf{t^k} = (t_1(x_1^k), \ldots, t_{|E|}(x_{|E|}^k))$.

3. **Direction Finding**: Generate the target flows.

   - Generate $\mathbf{y^k} = (y_1^k, \ldots, y_{|E|}^k)$ based on $\mathbf{t^k}$.

4. **Move**:

   - $\mathbf{x^{k+1}} = (1 - \alpha_k)\mathbf{x^k} + \alpha_k \mathbf{y^k}$.

   Note: This is a convex combination of $\mathbf{x^k}$ *and* $\mathbf{y^k}$*. As* $\mathbf{x^k} \in \mathcal{X}$ *and* $\mathbf{y^k} \in \mathcal{X}$ *are in the set of feasible flows* $\mathcal{X}$*, then* $\mathbf{x^{k+1}} \in \mathcal{X}$ *is also a feasible flow.*

5. **Convergence Test**: If convergence met, stop. Else set $k = k + 1$ and return to Step 2.

2.2.2.2 Method of Successive Averages

During each iteration the method of successive averages generates the target solution $\mathbf{y^k}$ via an all-or-nothing assignment and simply takes a weighted average by setting $\alpha_k = \frac{1}{k+1}$. Hence α_k takes successive values from the sequence $(\frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \ldots)$.

2.2.2.3 Frank-Wolfe

Named after the convex combination algorithm proposed by Frank and Wolfe in 1956 [65], the Frank-Wolfe method also generates the target solution $\mathbf{y^k}$ via an all-or-nothing assignment. However, for the the move step of the algorithm α_k is found by a line search that solves the following optimisation problem,

$$\min_{\alpha_k \in [0,1]} \left(U((1-\alpha_k)\mathbf{x^k} + \alpha_k \mathbf{y^k}) \right).$$

As $U(\mathbf{x})$ is generally a nonlinear function, this can be done by a root-finding method such as a bisection search.

2.2.2.4 Conjugate Frank-Wolfe

The target solution $\mathbf{y^k}$ should be conjugate to the previous target $\mathbf{y^{k-1}}$, i.e.

$$(\mathbf{y^{k-1}} - \mathbf{x^k})^T \mathbf{H^k} (\mathbf{y^k} - \mathbf{x^k}) = 0, \tag{2.1}$$

where,

$$\mathbf{H^k} = \text{diag}\left(t_1(x_1^k), \ldots, t_{|E|}(x_{|E|}^k) \right),$$

is the positive definite Hessian of the Beckmann function $U(\mathbf{x})$ (1.5a).

Let $\mathbf{y^k}$ be a convex combination of the the previous target solution $\mathbf{y^{k-1}}$ and an all-or-nothing assignment $\hat{\mathbf{y}}^\mathbf{k}$,

$$\mathbf{y^k} = (1-\theta)\hat{\mathbf{y}}^\mathbf{k} + \theta \mathbf{y^{k-1}}. \tag{2.2}$$

Substituting equation (2.2) into equation (2.1) and solving for θ,

$$\theta = \frac{(\mathbf{y^{k-1}} - \mathbf{x^k})^T \mathbf{H^k} (\hat{\mathbf{y}}^\mathbf{k} - \mathbf{x^k})}{(\mathbf{y^{k-1}} - \mathbf{x^k})^T \mathbf{H^k} (\hat{\mathbf{y}}^\mathbf{k} - \mathbf{y^{k-1}})}. \tag{2.3}$$

Thus, the target solution $\mathbf{y^k}$ is calculated by equation (2.2) using the value of θ given by equation (2.3).

The move step proceeds according to the standard Frank-Wolfe method, i.e. α_k is found by a line search.

2.2.2.5 Bi-conjugate Frank-Wolfe

Bi-conjugate Frank-Wolfe extends Conjugate Franke-Wolfe by considering the previous two target solutions $\mathbf{y^{k-1}}$ and $\mathbf{y^{k-2}}$. $\mathbf{y^k}$ is chosen to be conjugate to both $\mathbf{y^{k-1}}$

and $\mathbf{y^{k-2}}$, that is it solves the pair of equations,

$$(\mathbf{y^{k-1}} - \mathbf{x^k})^T \mathbf{H^k} (\mathbf{y^k} - \mathbf{x^k}) = 0 \tag{2.4}$$

$$(\mathbf{y^{k-2}} - \mathbf{x^k})^T \mathbf{H^k} (\mathbf{y^k} - \mathbf{x^k}) = 0, \tag{2.5}$$

and is a convex combination of the previous target solutions,

$$\mathbf{y^k} = \beta_0 \hat{\mathbf{y}}^\mathbf{k} + \beta_1 \mathbf{y^{k-1}} \beta_2 \mathbf{y^{k-2}}. \tag{2.6}$$

Full details of both conjugate and bi-conjugate Frank-Wolfe are given in [108].

2.2.3 Path-based Algorithms

Path-based algorithms are more closely related to traditional gradient based optimisation methods and follow a similar iterative scheme. This section presents the general iterative scheme and two variants from Jayakrishnan et al (1994) [83] and Florian et al (2009) [63].

2.2.3.1 General Iterative Scheme for a Path-based Algorithm

1. **Initialisation**:
   - For each commodity i (OD pair (r_i, s_i)):
     - Find the shortest path π_i^* based on initial edge costs $\mathbf{t^0} = (t_1(0), \ldots, t_{|E|}(0))$
     - Initialise the set of used paths $\hat{\Pi}_i \leftarrow \{\pi_i^*\}$
     - Assign all demand d_i to path π_i^*
   - $\hat{\Pi} \leftarrow \bigcup_i \hat{\Pi}_i$
   - $k \leftarrow 1$
2. **Update**:
   - Calculate $\mathbf{x^k}$ based on all used paths $\hat{\Pi}$
   - $\mathbf{t^k} = (t_1(x_1^k), \ldots, t_{|E|}(x_{|E|}^k))$
   - For each commodity i (OD pair (r_i, s_i)):
     - Find the shortest path π_i^* based on the edges costs $\mathbf{t^k}$

- Add the path to the set of used paths (if not already used)
  $\hat{\Pi}_i \leftarrow \hat{\Pi}_i \cup \{\pi_i^*\}$

3. **Move**:

   - For each commodity i (OD pair (r_i, s_i)):
     - Shift flow between used paths, $\pi \in \hat{\Pi}_i$, to bring closer to equilibrium
       $\mathbf{f_i^{k+1}} \leftarrow \mathbf{f_i^k} + \alpha_k \mathbf{\Delta f_i^k}$

   Note: travel times may also be updated after shifting flow for each OD pair.

4. **Convergence Test**: If convergence met, stop. Else set $k = k + 1$ and return to Step 2.

2.2.3.2 Gradient Projection [Jayakrishnan 1994]

Jayakrishnan et al proposed a quasi-newton method of the form $\mathbf{f}^{k+1} = \mathbf{f}^k - \alpha_k \mathbf{B}^{-1} \nabla_{\mathbf{f}} U(\mathbf{f}^k)$ [83]. The **Move** step of the algorithm proceeds by shifting flow to the shortest path for a commodity i from the other used paths.

Gradient projection effectively computes the gradient, takes a step in that direction and then projects back onto the feasible set of solutions.

Noting that the flow $f_{\pi_i^*}$ on the shortest path π_i^* can be expressed in terms of the demand d_i and the other used paths for commodity i,

$$f_{\pi_i^*} = d_i - \sum_{\pi \in \hat{\Pi}_i : \pi \neq \pi_i^*} f_\pi, \tag{2.7}$$

the shortest path flow variable can be eliminated and the gradient and **Move** step computed as follows.

Let δ_e^π be an indicator variable that represents whether edge e is on path π, i.e. $\delta_e^\pi = 1$ represents that edge e is on path the π,

$$\begin{aligned} x_e &= \sum_{i \in K} \sum_{\pi \in \hat{\Pi}_i : \pi \neq \pi_i^*} \delta_e^\pi f_\pi + \sum_{i \in K} \delta_e^{\pi_i^*} f_{\pi_i^*} \\ &= \sum_{i \in K} \sum_{\pi \in \hat{\Pi}_i : \pi \neq \pi_i^*} \delta_e^\pi f_\pi + \sum_{i \in K} \delta_e^{\pi_i^*} \left(d_i - \sum_{\pi \in \hat{\Pi}_i : \pi \neq \pi_i^*} f_\pi \right) \end{aligned} \tag{2.8}$$

$$\frac{\partial U(\mathbf{x}(\mathbf{f}))}{\partial f_\pi} = \sum_{e \in E} \frac{\partial U}{\partial x_e} \frac{\partial x_e}{\partial f_\pi} = \sum_{e \in E} t_e(x_e)[\delta_e^\pi - \delta_e^{\pi_i^*}] = c_\pi(\mathbf{f}) - c_{\pi_i^*}(\mathbf{f}) \tag{2.9}$$

The diagonal entries of the Hessian are given by,

$$\frac{\partial^2 \bar{U}}{\partial f_\pi^2} = \frac{\partial}{\partial f_\pi}\left(\frac{\partial \bar{U}}{\partial f_\pi}\right) = \frac{\partial}{\partial x_e}\frac{\partial x_e}{\partial f_\pi}\left(\sum_{e\in E} t_e(x_e)[\delta_e^\pi - \delta_e^{\pi_i^*}]\right)$$
$$= \sum_{e\in E} \frac{dt_e(x_e)}{dx_e}[\delta_e^\pi - \delta_e^{\pi_i^*}]^2 \tag{2.10}$$

Let E_π represent the edges on path π and not on π_i^*, i.e. $\delta_e^\pi = 1, \delta_e^{\pi_i^*} = 0$ and $E_{\pi_i^*}$ represent the edges on path π_i^* and not on π, i.e. $\delta_e^\pi = 0, \delta_e^{\pi_i^*} = 1$. Then the approximation to the Hessian $\mathbf{B}$ consists of diagonal entries,

$$\frac{\partial^2 \bar{U}}{\partial f_\pi^2} = \sum_{e\in E_\pi \Delta E_{\pi_i^*}} \frac{dt_e(x_e)}{dx_e}, \qquad \begin{cases} E_\pi \text{ set of edges on path } \pi \\ E_{\pi_i^*} \text{ set of edges on path } \pi_i^* \end{cases},$$

and is a positive definite diagonal matrix and therefore computing the inverse is not computationally intensive.

Utilising the iterative scheme $\mathbf{f}^{k+1} = \mathbf{f}^k - \alpha_k \mathbf{B}^{-1}\nabla_{\mathbf{f}} U(\mathbf{f}^k)$, the **Move** step is given by:

Move: Calculate the new path flows.

- For each commodity i (OD pair (r_i, s_i))

$$f_\pi^{k+1} = \max\{0, f_\pi^k - \frac{\alpha_k}{b_\pi^k}(c_\pi^k - c_{\pi_i^*}^k)\}, \qquad \forall \pi \in \hat{\Pi}_i : \pi \neq \pi_i^*$$

where,

$$c_\pi^k - c_{\pi_i^*}^k = \frac{\partial \bar{U}}{\partial f_\pi^k}, \qquad \forall \pi \in \hat{\Pi}_i : \pi \neq \pi_i^*$$
$$b_\pi^k = \sum_{e\in E_\pi \Delta E_{\pi_i^*}} \frac{dt_e(x_e^k)}{dx_e^k}, \qquad \forall \pi \in \hat{\Pi}_i : \pi \neq \pi_i^*$$
$$f_{\pi_i^*}^{k+1} = d_i - \sum_{\pi\in\hat{\Pi}_i : \pi\neq\pi_i^*} f_\pi^{k+1}$$

and α_k is a scalar step-size.

The final stage of the **Move** is to calculate the new edge flows $x_e^{k+1}\ \forall e \in E$. Note that this can either be done after flow is shifted for each commodity i or after all flow has been shifted for all commodities. The later is preferred as it requires computing

the shortest paths once per iteration, this can be done using a single-source all destinations algorithm such as Dijkstra's algorithm and updating the edges flows once. The former, for n commodities, requires updating the edge flows n times and computing single shortest paths n times.

2.2.3.3 Projected Gradient [Florian et al 2009]

Projected gradient differs from gradient projection in that the gradient is computed and then projected onto a feasible direction, thus when a step is taken the new point is within the feasible set of solutions, whereas gradient projection potentially leaves the feasible set of solutions to then be projected back.

The algorithm essentially shifts flow given by the average cost of the working paths to maintain feasibility. The partial derivatives of $U(\mathbf{f})$ with respect to the flow on a path f_π, $\frac{\partial U(\mathbf{f})}{\partial f_\pi}$ are given by (Section 1.3.2),

$$\frac{\partial U(\mathbf{f})}{\partial f_\pi} = c_\pi(\mathbf{f}),$$

and therefore the gradient vector $\nabla_{\mathbf{f}} U$ is,

$$\nabla_{\mathbf{f}} U = (c_\pi)^T_{\pi \in \Pi}.$$

Considering only the used paths of commodity i, the direction of steepest descent $\mathbf{q}$ is given by,

$$\mathbf{q_i} = -(c_\pi)^T_{\pi \in \hat{\Pi}_i}.$$

The idea of the algorithm is to take a step $\mathbf{\Delta f}$ in the direction of $\mathbf{q_i}$ whilst remaining in the feasible set. This requires that the following equation holds,

$$\sum_{\pi \in \Pi_i} (f_\pi + \Delta f_\pi) = d_i,$$

and given,

$$\sum_{\pi \in \Pi_i} f_\pi = d_i,$$

from equation (1.5b) then,

$$\sum_{\pi \in \Pi_i} \Delta f_\pi = 0.$$

Therefore, project $\mathbf{q_i}$ onto the set $\Delta F = \{\mathbf{\Delta f} : \sum_{\pi \in \Pi_i} \Delta f_\pi = 0\}$, this has a simple closed form and full details on the derivation and proof can be found in [63]. The

projection of steepest descent for commodity i is given by,

$$\mathbf{\Delta f_i} = (c_\pi - \bar{c}_\pi)^T_{\pi \in \hat{\Pi}_i},$$

and at iteration k by,

$$\mathbf{\Delta f_i^k} = (c_\pi(\mathbf{f^k}) - \bar{c}_\pi(\mathbf{f^k}))^T_{\pi \in \hat{\Pi}_i},$$

note that the cost of the path π at iteration k is a function of the flow on the network at iteration k, i.e. $c_\pi(\mathbf{f^k})$.

The flows are then updated accordingly,

$$\mathbf{f_i^{k+1}} \leftarrow \mathbf{f_i^k} + \alpha_k \mathbf{\Delta f_i^k}.$$

To maintain non-negativity the following condition must also hold,

$$f_\pi^k + \alpha_k \Delta f_\pi^k \geq 0, \quad \forall \pi \in \hat{\Pi}_i.$$

Therefore, for any path $\Delta f_\pi^k < 0$,

$$\alpha_k \leq \frac{f_\pi^k}{\Delta f_\pi^k}.$$

Letting

$$\bar{\alpha_k} = \min_{\pi \in \hat{\Pi}_i : \Delta f_\pi^k < 0} \frac{f_\pi^k}{\Delta f_\pi^k},$$

then α_k is found via a line search for the interval $[0, \bar{\alpha_k}]$.

2.2.4 Bush-based Algorithms

As aforementioned in Section 2.1, two new algorithms were developed in the early 2000s, Origin-Based Assignment (OBA) (Bar-Gera [13]) and Algorithm B (Dial [57]) and would subsequently be categorised as bush-based algorithms (Nie [126]). There is some debate over the exact timeline of these two algorithms, Dial reports that his original submission to the Transportation Research Board (TRB) annual conference was in 1999 in which Bar-Gera also submitted his PhD thesis on OBA. Official publication of Algorithm B would not happen until 2006 [52].

Bush-based methods are label correcting algorithms that rely on the concept of a bush and shifting flow between paired alternative segments (PAS) (see Section 2.2.4.2). Algorithm B and OBA differ in the way that they both equilibriate a

bush and the iterative scheme in which they operate. The efficiency of bush-based methods is due to the fact that bushes (see Section 2.2.4.1) are an extension of directed acyclic graphs (DAGs) and thus computation of shortest and longest paths can be done in linear time due to their topological ordering.

Bush-based algorithms have not been implemented for use in this thesis, but an overview of Dial's method is given and some notes on the others. Due to the complexity of these algorithms and the required space for correct presentation, a description of their workings is opted for. Full details of the algorithms can be found in the original publications as mentioned in each corresponding section.

2.2.4.1 Bush

Dial first introduced bushes in 1971 [53] on work related to probabilistic multipath traffic assignment. His initial definition of a bush was based on the concept of efficient paths, paths that contained only edges whereby the tail vertex is closer to the origin node, i.e. paths do not backtrack, between an origin vertex and all other vertices. Later definitions do not enforce backtracking and instead simply use the shortest path definition. Boyce [31] refers to a bush as including shortest paths between an origin and all destinations; for some bush-based algorithms this is sufficient, but for others paths from the origin to all other vertices in the network may required for efficient updating of the labels. The following definition of a bush assumes the more general case as all algorithms are unaffected by this definition.

Definition 2.2.2 (Bush)**.** *A bush is a connected, acyclic network which includes paths from an origin vertex to all other vertices.*

For the directed graph given in Figure 2.1, an example of a bush is given for vertex 1 in Figure 2.2.

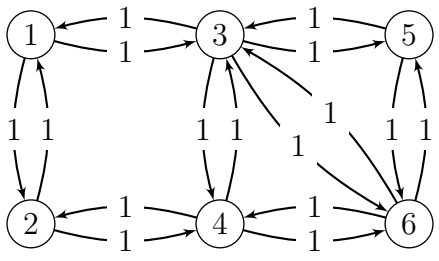


Figure 2.1: Directed network example.

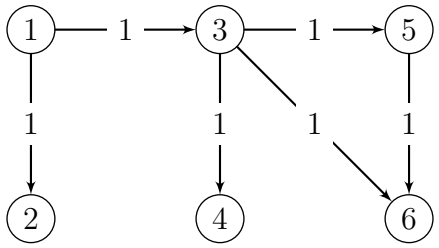


Figure 2.2: Example bush for vertex 1.

2.2.4.2 Paired Alternative Segment

The term 'paired alternative segment' was coined by Bar-Gera in 2010 [15], although the concept is utilised in both Dial [57] and Bar-Gera's [13] earlier work, who proposed an algorithm that satisfied a proposed 'proportionality condition' which stated that 'the proportion of travelers on each of the two alternative segments should be the same regardless of their origin or their destination' [15].

Definition 2.2.3 (Paired Alternative Segments)**.** *For a given origin vertex i and destination vertex j in a bush, and shortest and longest paths, let a be the last vertex common to both paths. Then the shortest and longest path segments from a to j form a PAS.*

On inspection of Figure 2.2 the shortest and longest paths for vertex 1 to vertex 6 are 1-3-6 and 1-3-5-6, respectively. The last common vertex is vertex 3 and the segments 3-6 (shortest) and 3-5-6 (longest) form a pair of alternative segments.

2.2.4.3 General Iterative Scheme for a Bush-based Algorithm

1. **Initialisation**: For each origin, find initial bush (acyclic network) and feasible flow (all or nothing assignment)

2. **Update**: For each bush, update bush labels

3. **Equilibrate Bush**: For each bush, shift flow such that the path costs between all used paths for an origin and destination within the bush is minimal.

4. **Improve Bush**: For each bush, introduce cheaper edges such that bush remains acyclic and flows are feasible.

5. **Convergence Test**: If convergence met, stop. Else return to step 2.

2.2.4.4 Algorithm B

The idea behind Algorithm B can be summarised as iteratively shifting flow between maximum and minimum path PAS segments based on Newton's method. Algorithm B can be seen as a label correcting algorithm and a general description is presented here, for full details see [57].

For each origin, create a bush and label the vertices in topological order, label each vertex j with the shortest possible path L_j and longest path U_j.

Let i be the vertex with highest topological order within the bush, find the last vertex a where the minimum and maximum travel times do not differ, the segments between a and i form a PAS. Equilibriate the PAS (i.e. the time for the minimum and maximum path for vertex i are the same) using a Newton step similar to the step used in gradient projection until either costs are the same or the flow (within the bush) on the maximum path is empty. Repeat for all vertices in the bush.

To improve a bush, drop any empty edge that does not affect connectivity of the bush and add edges (i, j) whereby $L_i + t_{ij} < L_j$ (t_{ij} is the time experienced on edge (i, j)), that is add a edge which provides an improvement on the shortest path to j.

Nie [126] points out that whilst it can be shown that if the bushes are at equilibrium this guarantees that a cycle is not introduced to the bush, as bushes will only be epsilon equilibriated, introducing a cycle is theoretically possible. However, there does not appear to be any practical evidence of this happening in Dial's implementation and results. Nie suggests a slightly different approach whereby edges are added based on $U_i + t_{ij} > U_j$ which guarantees that a cycle will not be introduced. It does however require additional calculation of all the longest path labels to vertices, which are not required in Dial's and may have a detrimental impact on performance.

2.2.4.5 Origin Based Assignment (OBA)

Introduced by Bar-Gera in 2002, OBA differs from Algorithm B by attempting to shift flow among many paths simultaneously as opposed to Dial's Algorithm B which shifts flow between the PAS on the shortest and longest paths to a given node. For full details of the algorithm see [13].

To achieve this, the algorithm uses routing variables to represent the proportion

of the flow entering a vertex i from each of the incoming edges. Then, using a combination of labels, which keep track of the average travel time to a vertex and estimates of the derivative with respect to the the routing variables, i.e. the rate of change of the average travel time based on changes to an incoming edge, flow is shifted according to a Newton step.

2.2.4.6 Traffic Assignment by Paired Alternative Segments

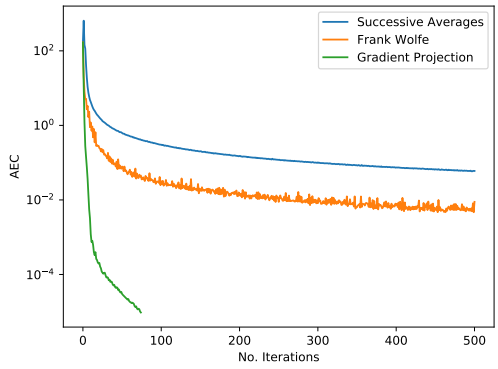
As aforementioned in Section 2.2.4.2, Bar-Gera proposed an algorithm that uniquely determines path flows based on a 'proportionality condition'. The concept of the algorithm is to maximise the path entropy, that is to find the most likely configuration of paths for a given optimal edge solution. Maximisation of path entropy was studied by Bar-Gera in [14]. TAPAS is much more efficient on large scale networks than OBA, but still lags behind Algorithm B [52].

2.2.5 Computational Efficiency and Discussion

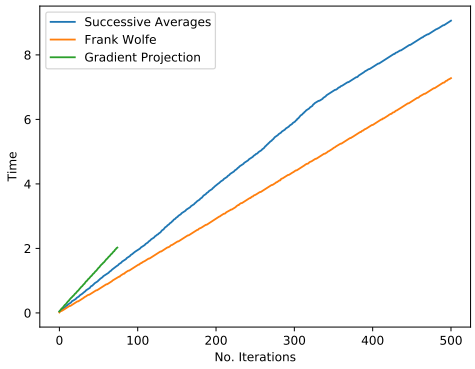
Link-based and path-based algorithms rely on shortest path algorithms, such as Dijkstra's algorithm during each iteration, which are computationally expensive. Recent work into shortest path algorithms has seen some major advances in the computational efficiency of finding shortest paths. Critically the development of contraction hierarchies [67] has resulted in a dramatic speed up of these algorithms. Schnek and Nokel found that on the largest test networks, contraction hierarchies resulted in a speedup factor of 42 [150].

The current fastest algorithms for solving the nonatomic selfish routing problem are all bush-based, taking advantage of the acyclic nature of the bush and the speed in which shortest and longest paths can be found through the manipulation of labels. Table 2.1 summarises link-based, path-based and bush-based methods in terms of their convergence speed, space requirement and accuracy. For a detailed analysis of the computational efficiency of algorithms for nonatomic selfish routing see [154].

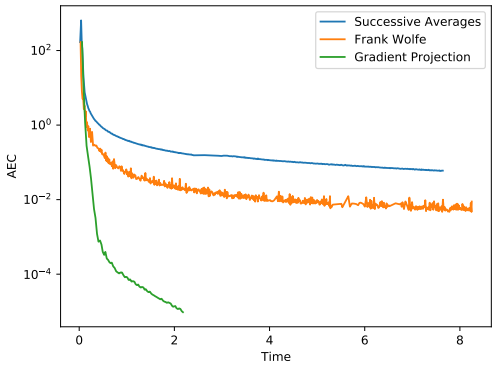
The algorithms presented in Section 2.2.2.2, 2.2.2.3 and 2.2.3.2 were implemented (implementations can be found at [129]) and Figure 2.3 gives a computational comparison for Sioux Falls (see Appendix C.3) for the implemented methods. Whilst the speed per iteration is much the same, clearly the convergence behaviour of the path-



(a) No iterations vs AEC.



(b) No iterations vs Time.



(c) Time vs AEC.

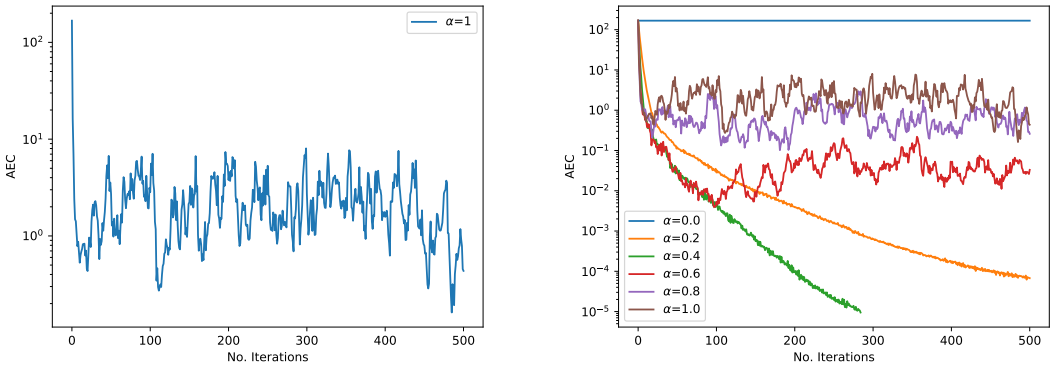
Figure 2.3: Comparison of implemented algorithms for solving Sioux Falls.

based method which uses gradient descent is far superior. The other observation is the tailing of successive averages and Frank-Wolfe which, due to using extreme points in the direction finding step (all-or-nothing assignment), bounces around the search space.

Whilst not implemented for the purposes of this thesis, as aforementioned, bush-based algorithms remain the fastest known algorithm for solving nonatomic selfish routing games and the advances in shortest path technology mean that very large networks can now be solved in a fraction of the previous time.

2.2.5.1 A Note on the Convergence of Networks with Symmetry

Figure 2.4a illustrates a case in which the path-based approach fails to converge in a reasonable number of iterations with a step size of $\alpha = 1$. Figure 2.4b displays different values of α.



(a) $\alpha = 1$ (b) Comparison of step sizes (α).

Figure 2.4: Analysis of gradient projection performance for solving Dial's network.

Method Type	Convergence Speed	Space Requirement	Accuracy
Link-based	Slow	Low	Low
Path-based	Medium	High	Medium
Bush-based	Fast	Medium	High

Table 2.1: Comparison of nonatomic selfish routing method types

Given the symmetry of the network, the sensitivity of the algorithm to the step size is likely due to the following. Whilst the minimum is globally optimal in terms of edge flows, in terms of paths it is not and therefore it is likely that due to the number of similar paths available to each commodity pair, the path-based algorithm finds it difficult to focus in a particular subset of the paths that lead to the minimum and is also overshooting the minimum in the cases of the larger step sizes.

2.3 Algorithms for Unweighted Atomic Selfish Routing

In general, all the algorithms presented in the previous section can be amended to restrict the flow to be integral by enforcing the amount of flow that is shifted between paths and edges to be integral. As minimising the potential function of (1.24) tracks the moves of best response dynamics, shifting integral flow that results in better value of the objective function can be seen as moving the current solution

towards a Nash equilibrium.

However, in addition this section presents a number of novel algorithms which solve (1.24) or a restricted version of the problem. Implementation of these algorithms can be found at [129].

2.3.1 Dynamic Programming

Bagdasar, Berry, O'Neill, Popovici and Raja [9] introduced a polynomial time dynamic programming method to solve the unweighted atomic selfish routing game (1.24) with objective (1.25) on a single commodity, parallel edge network. A brief explanation of this algorithm is presented along with the extension of how to find exact solutions to (1.24).

For (1.24) with objective (1.25), with m resources and n players, cost functions $t_1(x_1), \ldots, t_m(x_m)$ and total cost $T(x) = \sum_{i=1}^{m} x_i t_i(x_i)$, define $g_i(x_i) = x_i t_i(x_i)$ recursively the Bellman functions $G_1, \ldots, G_m : [0, n] \cap \mathbb{N} \to \mathbb{R}$ for all $c \in [0, n] \cap \mathbb{N}$

$$\begin{cases} G_1(c) & = g_1(c); \\ G_k(c) & = \min_{x \in [0,c] \cap \mathbb{N}} \left[g_k(x) + G_{k-1}(c-x) \right], \quad k = 2, 3, \ldots, m. \end{cases} \tag{2.11}$$

Then, the optimal value of the problem is given by

$$\min\{T(x) \mid x = (x_1, \ldots, x_m) \in \mathbb{N}^m,\ x_1 + \cdots + x_m = n\} = G_m(n). \tag{2.12}$$

An optimal solution $x^0 = (x_1^0, \ldots, x_m^0)$ of the problem can be deduced by the backward recursive procedure:

Let $c := n$ and choose $x_m^0 \in \operatorname{argmin}_{x \in [0,c] \cap \mathbb{N}} \left[g_m(x) + G_{m-1}(c-x) \right]$,

Let $c := n - x_m^0$ and choose $x_{m-1}^0 \in \operatorname{argmin}_{x \in [0,c] \cap \mathbb{N}} \left[g_{m-1}(x) + G_{m-2}(c-x) \right]$,

...

Let $c := n - x_m^0 - \ldots - x_3^0$ and choose $x_2^0 \in \operatorname{argmin}_{x \in [0,c] \cap \mathbb{N}} \left[g_2(x) + G_1(c-x) \right]$,

Let $x_1^0 := n - x_m^0 - \ldots - x_3^0 - x_2^0$.

A full explanation of this method and examples are given in [21].

To find a solution to (1.24), i.e. a Nash equilibrium, replace $g_i(x_i) = \sum_{i=1}^{x_i} t_i(x_i)$, the potential function of atomic selfish routing game.

2.3.1.1 Time Complexity

The time complexity of dynamic programming depends on the main recursive operations given in Section 2.3.1. For fixed $k \in \{1, m\}$ and $c \in [0, n]$, the number of operations required to compute $G_k(c)$ is $c + 1$ by (2.11). Therefore, the number of operations required to compute $G_k(c)$ for $c \in [0, n] \cap \mathbb{N}$ is,

$$\sum_{c=0}^{n}(c+1) = \frac{(n+1)(n+2)}{2}.$$

Evaluating for m roads, the total number of operations required will be,

$$m\left[\frac{(n+1)(n+2)}{2}\right],$$

thus resulting in a complexity of,

$$O\left(mn^2\right).$$

2.3.2 Tabu Search

For the standard atomic selfish routing game the metaheuristic method of a tabu search can be employed. A detailed explanation of the general tabu search method is given by Bandaru and Deb [12] in their survey of metaheuristics. Bagdasar, Berry and O'Neill [9] presented a tabu search with variable step size for solving the unweighted atomic selfish routing game (1.24) with objective (1.25) on a single commodity, parallel edge network. When compared with the dynamic programming method whose solution is optimal, the tabu search method performed favorably [9]. For user equilibrium, a variance-based formulation was used in which the costs of all available strategies is minimised. This resulted in good results for demands high enough that all resources (e.g. roads) are chosen by at least one player (e.g. vehicle).

Algorithm 1 presents an outline for a tabu search in the format presented in [12] that can be used with atomic selfish routing games with a finitely tractable strategy space for finding the minimum social cost and user equilibrium. For an exponentially large strategy space, feasible strategy profiles that may improve the solution can be added after each iteration similar to column generation used in the likes of Danzig-Wolfe decomposition [46].

For this method to be effective, it requires an understanding of parameter tuning and convergence criteria (true convergence requires a local polynomial search to

check that the solution is indeed a pure Nash equilibrium, see section 1.2.5.5). The parameters N, τ, k are the allowed neighbourhood size, the number of iterations a solution is banned and the maximum size of the tabu list, respectively. A solution is the strategy profile $a = (a_i, \ldots, a_n)$ of the strategies currently played by each of the n players. The fitness function $f : \mathcal{A} \mapsto \mathbb{R}$ for finding the minimum social cost is defined as,

$$\sum_{e \in E} x_e l_e(x_e),$$

and, utilising the potential function of the atomic selfish routing game, the fitness function $f : \mathcal{A} \mapsto \mathbb{R}$ for finding a Nash equilibrium,

$$\sum_{e \in E} \sum_{k=1}^{x_e} l_e(k)$$

where the number of players x_e using resource e is,

$$\sum_{i=1}^{n} \sum_{a_i \in \mathcal{A}_i : e \in a_i} f_{a_i} = x_e.$$

2.3.2.1 Time Complexity

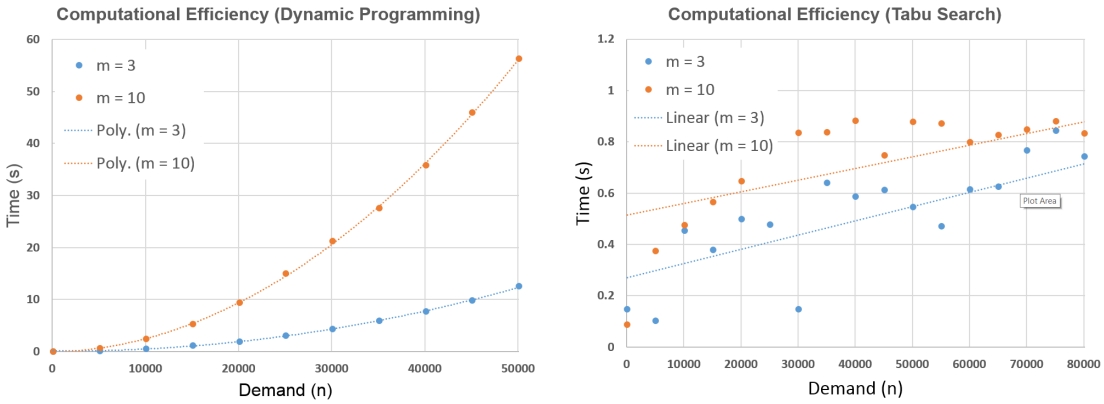
As the tabu search method is a metaheuristic it is not possible to give a time complexity and one must be careful as to how to determine whether the algorithm has converged, the simplest option being to stop after a specified number of iterations whereby the best solution has not improved. Figure 2.5 presents the performance comparison of the dynamic programming method and tabu search method in solving a 3 edge and 10 edge parallel edge network. Inspection of Figure 2.5 and the associated fitted curves showcase the quadratic nature of the dynamic programming method in terms of n and that the tabu search can be approximated as linear.

2.3.3 Linear Programming - Piecewise Linear Convex Function

As the objective function given in the optimisation problem (1.24) is evaluated at discrete intervals, the objective function can be modelled as a piecewise linear approximation (Figure 2.6) matching the discrete potential function of the atomic selfish routing game. As the feasible region is also a set of linear equalities and

Algorithm 1 Tabu search for a atomic selfish routing game

Require N, τ, k
t = 0
Initialise random solution s_0
Initialise tabu list T
Add s_0 to the tabu list T
Set $s_{best} = s_0$
repeat
    Evaluate fitness of s_t
    Set $\mathcal{N}(s_t) = [\,]$
    repeat
        Find a neighbour s'_t
        if s'_t is in T **then**
            Discard neighbour
        else
            Add neighbour to $\mathcal{N}(s_t)$
        end if
    until $|\mathcal{N}(s_t)| = N$ **or** no more neighbours exist
    Find best solution s'_t in $\mathcal{N}(s_t)$
    if $f(s'_t) < f(s_t)$ **then**
        $s_{best} = s'_t$
    end if
    Add s'_t to T
    Remove solutions older than τ from T
    if $|T| > k$ **then**
        Remove oldest solution from T
    end if
until Stopping criteria is satisfied



(a) Dynamic Programming (b) Tabu Search

Figure 2.5: Computational efficiency for 3 and 10 parallel edges.

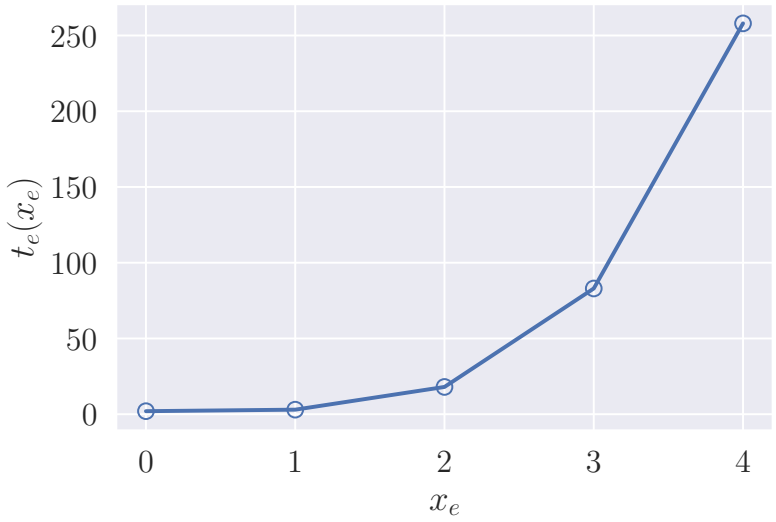


Figure 2.6: Piecewise linear approximation of example edge function $t_e(x_e) = 2 + x_e^4$.

inequalities, the optimisation problem 1.24 can be solved via the following mixed-integer linear programming problem (MILP).

Let $D = \sum_{j \in \{1,\dots k\}} d_j$, that is the upper bound on the amount of flow that can be induced on an edge.

$$\text{minimise} \quad \sum_{e \in E} y_e \tag{2.13a}$$

subject to

$$\sum_{i=0}^{D} \lambda_e^i i = x_e, \qquad \forall e \in E \tag{2.13b}$$

$$\sum_{i=0}^{D} \lambda_e^i \sum_{j=0}^{i} t_e(j) = y_e, \qquad \forall e \in E \tag{2.13c}$$

$$\sum_{i=0}^{D} \lambda_e^i = 1, \qquad \forall e \in E \tag{2.13d}$$

$$\sum_{\pi \in \Pi_j} f_\pi = d_j, \qquad \forall j \in \{1, \ldots, k\} \tag{2.13e}$$

$$\sum_{\pi \in \Pi : e \in \pi} f_\pi = x_e, \qquad \forall e \in E \tag{2.13f}$$

$$f_\pi \geq 0, \qquad \forall \pi \in \Pi \tag{2.13g}$$

$$f_\pi \in \mathbb{Z}, \qquad \forall \pi \in \Pi \tag{2.13h}$$

$$\lambda_e^i \in \{0, 1\}, \qquad \forall i \in \{1, \ldots, D\}, \forall e \in E \tag{2.13i}$$

$$x_e \geq 0 \tag{2.13j}$$

$$y_e \geq 0. \tag{2.13k}$$

Constraints (2.13b), (2.13d) and (2.13i) guarantee that only $\lambda_e^{x_e} = 1$ for a given edge e. Thus, $y_e = \sum_{i=0}^{x_e} t_e(i)$ and therefore the objective function (2.13a) is $\sum_{e \in E} \sum_{i=0}^{x_e} t_e(i)$ as per the objective function given in (1.24), e.g. if for edge a, $x_a = 5$, $\lambda_a^5 = 1$ and $y_a = \sum_{i=0}^{5} t_a(i)$.

This is a variant of the standard linear programming trick for converting a non linear function into a piecewise linear approximation. The difference here is that the additional constraint (2.13i) stipulates that the function is evaluated at integer points, not a combination of contiguous end points of piecewise linear segments, as per those specified when using the special order sets of type 2 (SOS2) [169]. The optimal solution of the MILP is therefore an optimal solution to (1.24).

In addition the constraints (2.13h) and (2.13i) can be removed and the constraint that $\lambda_e^i \in \text{SOS2}, \forall i \in \{1, \ldots, k\}, \forall e \in E$ added. Amending (1.24c) to be discretised at N points, $\Delta D = D/N$, $\sum_{i=0}^{N} \lambda_e^i \sum_{j=0}^{i} t_e(j \cdot \Delta D) \cdot \Delta D = y_e, \forall e \in E$. As $N \to \infty$ and $\Delta D \to 0$, y_e will approach the Beckmann integral (1.5a) and thus the optimal solution of (1.5a) is approached.

Whilst sacrificing some accuracy for solving (1.5), the MILP model allows for interesting extensions to be undertaken such as edge capacity constraints, a limit on the number of edges, etc. via the use of binary variables. Solutions to non-convex functions via the use of piecewise linear approximations can also be guaranteed.

An implementation of this approach using Gurobi [73] can be found at [129].

2.4 An Online Learning Approach to Unweighted Atomic Selfish Routing

The final algorithms are based on an online probabilistic approach which utilises the concept of bandit machines.

In the advent of big data and deep learning it is interesting to consider approaches to the problem in which algorithms can learn to route based on selfish user input (in this case represented as bandits). An important question, within the context of selfish routing, is whether these algorithms tend to the equilibrium solution found by solving the problem using traditional methods and the user equilibrium objective function (i.e. the potential function of the game). Work on online learning algorithms using multi-armed-bandits is already prevalent in many areas such as online experiment design and A/B testing, for surveys into practical applications see [35, 28, 29].

2.4.1 Background

The multi-armed bandit (MAB) problem has received much attention in recent years within the online and machine learning community due to its appropriateness for demonstrating the fundamental trade-off between exploration and exploitation in online learning. The basic MAB problem is for an agent to maximise the cumulative reward received after playing a number of rounds (finite or infinite). In each round the agent is required to choose one of K bandits and subsequently receives an associated reward. For an agent to be successful it must employ a strategy which balances the trade-off between exploration and exploitation. Explore too little and the agent's preferred choice may remain sub-optimal, explore too often and the agent fails to exploit the most optimal choices. Numerous algorithms have been

studied for variants of the MAB problem and a popular measure of an algorithm's performance is the notion of expected regret, whereby the agent's received reward is compared with the expected reward that would have been received for the optimal choices [20].

In strategic repeated games, a natural approach towards equilibrium is to employ an online learning algorithm in which the expected regret of the player(s) is minimised over the time horizon [36]. Whilst expected regret analysis and convergence of equilibrium are important and rich areas of research, they make some key assumptions that could, in certain modelling scenarios, be deemed too restrictive. First, when bounding the regret of an algorithm it is necessary that the utility received by a player is itself bounded, therefore restricting the types of utility function. Second, convergence to a state of equilibrium does not take into account the capricious nature of certain individuals and that a player's rationality is often bounded by both the intractability of the decision making process and the player's preference for exhaustive search [69]. Therefore, the best one may be able to do is express a player's belief in the most preferable choices over a set of tractable strategies.

The above concepts are particularly inherent in routing games, a form of strategic repeated game in which multiple players (e.g. drivers of vehicles) simultaneously route flow across a network in an attempt to minimise their own cost. Routing games belong to the larger class of congestion games which possess the property of emitting at least one pure strategy Nash equilibrium [141] and have received much attention within the field of algorithmic game theory [127]. However, due to the underlying graph structure, the strategy set for these games suffers from the "curse of dimensionality" whereby the strategy set for a source sink pair (available paths) grows exponentially with the size of the underlying graph. Traditionally, methods have employed a centralised approach in which full information of the costs associated with all strategies is known, and flow is shifted globally between paths so as to satisfy a set of constraints representing a state of equilibrium for the given problem [135]. Such approaches fail to consider both the decentralised nature of the decision making processes within the system and that individual players have a particularly myopic view of the system and, therefore, tend to make decisions on very little information.

Motivated by the concepts of bounded rationality and random/deliberate sub-optimal choices, the focus of this section is to model unweighted atomic routing games under a restricted subset of strategies via noisy feedback, i.e. the utility may vary due to external factors. An investigation into a exponential weights algorithm is undertaken in which at each time step (round) uses feedback as a mechanism for a player to update their personal beliefs (probability distribution) of the best course of action and an ϵ-greedy algorithm in which the best course of action is selected greedily with probability $p = 1 - \epsilon$. Variants of both algorithms are implemented for the semi-bandit and bandit feedback scenarios.

2.4.2 Learning Under Bandit Feedback

An N-player congestion game consists of a finite number of players $\mathcal{N} = \{1, \cdots, N\}$, a set of congestible elements $\mathcal{E}$ with associated cost (latency) functions $l_e : \mathbb{N} \mapsto \mathbb{R}$ for each element $e \in \mathcal{E}$ and a set of playable strategies $\mathcal{A}_i$ for each player i, where a given strategy $a_i \in \mathcal{A}_i$ is a set of congestible elements $a_i \subseteq \mathcal{E}$. The number of players choosing element e is $x_e = \sum_{i \in \mathcal{N}} \sum_{e_i \in a_i} \mathbb{1}(e_i = e)$, where $\mathbb{1}$ is the indicator function. The associated cost to player i playing strategy a_i is $u_i(a_i; a_{-i}) = \sum_{e \in \mathcal{E}} \sum_{e_i \in a_i} \mathbb{1}(e_i = e) \cdot l_e(x_e)$. That is, each player picks a set of congestible elements and their associated costs are dependent not only on their own strategy, but on those played by the other players. The total cost U under strategy profile $\mathbf{a} = (a_i)_{i \in \mathcal{N}}$ is then,

$$U(\mathbf{a}) = \sum_{i \in \mathcal{N}} u_i(a_i; a_{-i}) = \sum_{i \in \mathcal{N}} \sum_{e \in \mathcal{E}} \sum_{e_i \in a_i} \mathbb{1}(e_i = e) \cdot l_e(x_e) = \sum_{e \in \mathcal{E}} x_e l_e(x_e).$$

Let $\mathcal{A} = \prod_i \mathcal{A}_i$ to be the set of all strategy profiles and $l = (l_e)_{e \in \mathcal{E}}$ the vector of cost functions associated with each e, then the congestion game is described by the tuple $(\mathcal{N}, \mathcal{E}, \mathcal{A}, l)$.

For an unweighted atomic routing game, let the set of congestible elements $\mathcal{E}$ be the edges in the graph $G = (V, E)$ and for each player $i \in \mathcal{N}$ associate a source/sink pair (o_i, d_i) and traffic demand $k_i = 1$, i.e. players route themselves. In general an unweighted traffic rate routes the same quantity $k_i = k, \ \forall i \in \mathcal{N}$. A player's strategy set $\mathcal{A}_i$ is the set of possible paths from source to sink, i.e. a strategy $a_i \in \mathcal{A}_i$ is a path consisting of edges $e \in E$ [144]. Therefore, the cost to a given player choosing a particular path is dependent on the number of players choosing paths which share

edges in the graph.

As a bandit problem, an unweighted atomic routing game consists of $\mathcal{N}$ players, a set E of functional bandit machines (edges), with corresponding congestion functions l. Each player $i \in \mathcal{N}$ then pulls a combination of bandit machines $a_i \subseteq E$ (path) from the strategy set $\mathcal{A}_i$ (set of available paths for (o_i, d_i) pair) and receives feedback given the strategy profile of played actions $\mathbf{a} = (a_i)_{i \in \mathcal{N}}$.

The following section introduces the exponential weights and ϵ-greedy algorithms for both semi-bandit and bandit feedback.

For each player i let $W_i^t = (W_{ia_i}^t)_{a_i \in \mathcal{A}_i}$ be a set of weights associated with the player's available strategies at a given round t. Denote the probability of a player selecting strategy a_i as,

$$\mathbf{P}_{ia_i}^t = \frac{W_{ia_i}^t}{\sum_{j=1}^{|\mathcal{A}_i|} W_{ij}^t},$$

and the probability distribution over all strategies $\mathcal{A}_i$ as $\mathbf{P}_i^t = (\mathbf{P}_{ia_i}^t)_{a_i \in \mathcal{A}_i}$.

2.4.2.1 Semi-bandit Feedback

Under semi-bandit feedback, the player has access to the entire payoff vector of playable strategies. The noisy feedback for a given strategy a_i^t played by player i in round t is,

$$\hat{r}_{ia_i}(a_{-i}^t) = u_i(a_i^t; a_{-i}^t) + \xi_{ia_i}^t,$$

and the entire payoff vector for all strategies available to player i is then,

$$\hat{\mathbf{r}}_\mathbf{i}^\mathbf{t} = (\hat{r}_{ia_i}(a_{-i}^t))_{a_i \in \mathcal{A}_i}.$$

For each player i, the exponential weights algorithm (see Algorithm 2) maintains the probability distribution $\mathbf{P}_i^t = (\mathbf{P}_{ia_i}^t)_{a_i \in \mathcal{A}_i}$ reflecting the beliefs about player i's best strategy from the strategy set $\mathcal{A}_i$. At time t, player i samples an action $a_i^t \sim \mathbf{P}_i^t$ and updates the distribution $\mathbf{P}_i^{t+1}$ based on the semi-bandit feedback it receives [38]. Note that due to the interdependence of the congestion functions, all players' actions must be selected and played before players receive their corresponding feedback.

The ϵ-greedy algorithm (see Algorithm 3) updates the average reward for all player strategies via the feedback vector and greedily selects the best known strategy with probability $p = 1 - \epsilon$ and randomly selects an action with probability $p = \frac{\epsilon}{\mathcal{A}_i}$.

Algorithm 2 Exponential weights with semi-bandit feedback [EW-SB]

Require: $\gamma_t = t^{-\frac{1}{\alpha}}\ \forall t \in [1, \dots, T]$, $W_i^0 \in \mathbf{1}^{|\mathcal{A}_i|}\ \forall i \in \mathcal{N}$

for $t = 1, \dots, T$ **do**
  for each player i in $\mathcal{N}$ **do**
    $\mathbf{P}_i^t = \frac{W_i^t}{\sum_{j=1}^{|\mathcal{A}_i|} W_{ij}^t}$ ▷ Calculate probability distribution for strategies
    $a_i^t \sim \mathbf{P}_i^t$ ▷ Sample action from probability distribution
  end for
  for each player i in $\mathcal{N}$ **do**
    $\hat{\mathbf{r}}_\mathbf{i}^\mathbf{t} = (\hat{r}_{ia_i}(a_{-i}^t))_{a_i \in \mathcal{A}_i}$ ▷ Observe estimated reward for strategies
    $W_i^{t+1} = W_i^t \cdot \exp\left(\frac{\gamma_t \hat{\mathbf{r}}_\mathbf{i}^\mathbf{t}}{|\mathcal{A}_i|}\right)$ ▷ Update weights
  end for
end for

2.4.2.2 Bandit feedback

Under bandit feedback the player only has access to feedback for the strategy played in round t and therefore a player must attempt to estimate the cost of strategies over time. The exponential Weights algorithm can be amended (see Algorithm 4) by utilising the importance sampling estimator.

The feedback for strategy a_i^t received in round t is the individual cost incurred by the player,

$$\hat{u}_i^t = u_i(a_i^t; a_{-i}^t) + \xi_i^t,$$

and the full feedback vector $\mathbf{r}_\mathbf{i}^\mathbf{t}(a_{-i}^t)$ can be estimated by $\hat{\mathbf{r}}_\mathbf{i}^\mathbf{t} = (\hat{r}_{ia_i}^t)_{a_i \in \mathcal{A}_i}$, where,

$$\hat{r}_{ia_i}^t = \begin{cases} \frac{\hat{u}_i^t}{\mathbf{P}_{ia_i}^t}, & \text{if } a_i = a_i^t. \\ 0, & \text{otherwise.} \end{cases} \qquad \forall a_i \in \mathcal{A}_i.$$

It can be shown [38] that under certain probabilistic assumptions , $\hat{r}_{ia_i}^t$ results in an unbiased estimator of the feedback received by player i playing action a_i calculated over the joint probability of all other strategy profiles $a_{-i} \in \prod_{j \neq i} \mathcal{A}_j$,

$$\mathbf{P}_{-i}^t = (\mathbf{P}_{-ia_{-i}}^t)_{a_{-i} \in \mathcal{A}_{-i}},$$

namely,

$$\mathbb{E}_t[\hat{r}_{ia_i}^t] = u_i(a_i; \mathbf{P}_{-i}^t) = \sum_{a_{-i} \in \mathcal{A}_{-i}} \mathbf{P}_{-ia_{-i}}^t u_i(a_i^t; a_{-i}^t).$$

Algorithm 3 ϵ-greedy with semi-bandit feedback [ϵG-SB]

Require: $W_i^0 \in \mathbf{0}^{|\mathcal{A}_i|}\ \forall i \in \mathcal{N}$

for $t = 1, \ldots, T$ **do**
  for each player i in $\mathcal{N}$ **do**
    if $\epsilon_t \sim unif(0,1) < \epsilon$ **then**
      $a_i^t \sim unif\{1, |\mathcal{A}_i|\}$ ▷ Choose at random with probability $p = \frac{1}{|\mathcal{A}_i|}$
    else
      $a_i^t = \arg\max_{a_i \in \mathcal{A}_i}(W_{ia_i}^t)$
    end if
  end for
  for each player i in $\mathcal{N}$ **do**
    $\hat{\mathbf{r}}_\mathbf{i}^\mathbf{t} = (\hat{r}_{ia_i}(a_{-i}^t))_{a_i \in \mathcal{A}_i}$ ▷ Observe estimated feedback for strategies
    $W_i^{t+1} = W_i^t + \frac{1}{t+1}\left[\hat{\mathbf{r}}_\mathbf{i}^\mathbf{t} - W_i^t\right]$ ▷ Update average feedback
  end for
end for

For the ϵ-greedy algorithm (see Algorithm 5) amend the update of the average rewards W_i^{t+1} to only update the strategy that has been played at time t.

2.4.3 Results

Algorithms 2-5 were tested on a bidirectional lattice network with 16 vertices and 48 edges. Given the stochastic nature of the algorithms, 10 randomly generated instances of the lattice network were generated and 250 players were routed between 4 origin destination pairs. The results were averaged over 10 episodes per network - each episode consisting of a 100 rounds ($T = 100$). Implementation and results can be found at [129]. Figure 2.7 (a) plots the total cost U averaged over the data set and for comparison, the total cost U_Φ experienced at the equilibrium given by the potential function Φ. Figure 2.7 (b) plots the regret of each algorithm defined to be the cumulative sum of the difference between the total cost of the played strategy profile $\mathbf{a}^t$ at time t and the equilibrium total cost U_Φ,

$$R_t = \sum_{i=t}^{t} \left[U(\mathbf{a}^\mathbf{t}) - U_\Phi\right].$$

Algorithm 4 Exponential weights with bandit feedback [EW-B]

Replace lines $7-8$ in algorithm 2 with:

$\hat{u}_i^t = u_i(a_i^t; a_{-i}^t) + \xi_i^t$ ▷ Observe estimated feedback for played strategy

$$\hat{r}_{ia_i}^t = \begin{cases} \frac{\hat{u}_i^t}{\mathbf{P}_{ia_i}^t}, & \text{if } a_i = a_i^t. \\ 0, & \text{otherwise.} \end{cases} \quad \forall a_i \in \mathcal{A}_i$$

▷ Estimate feedback vector $\hat{\mathbf{r}}_\mathbf{i}^\mathbf{t}$

$W_i^{t+1} = W_i^t \cdot \exp\left(\frac{\gamma_t \hat{\mathbf{r}}_\mathbf{i}^\mathbf{t}}{|\mathcal{A}_i|}\right)$ ▷ Update weights

Algorithm 5 ϵ-greedy with bandit feedback [ϵG-B]

Replace lines $10-11$ in algorithm 3 with:

$\hat{u}_i^t = u_i(a_i^t; a_{-i}^t) + \xi_i^t$ ▷ Observe estimated feedback for played strategy

$$\hat{r}_{ia_i}^t = \begin{cases} \frac{1}{t+1}\left[\hat{u}_i^t - W_{ia_i}^t\right], & \text{if } a_i = a_i^t. \\ 0, & \text{otherwise.} \end{cases} \quad \forall a_i \in \mathcal{A}_i$$

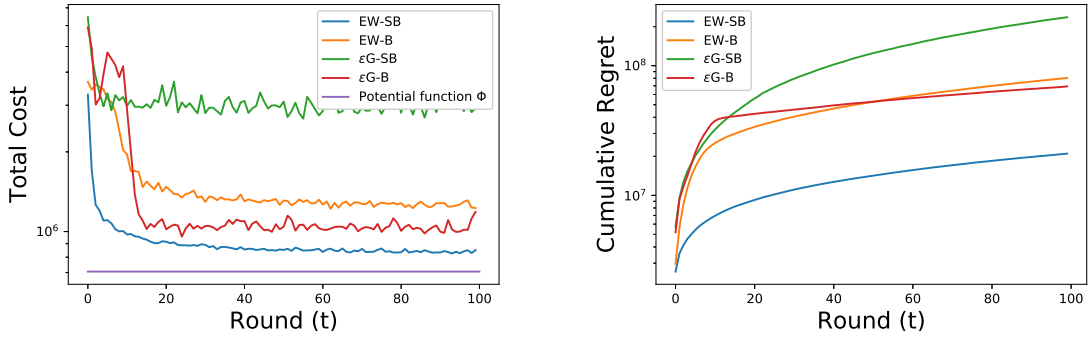
▷ Estimate feedback vector $\hat{\mathbf{r}}_\mathbf{i}^\mathbf{t}$

$W_i^{t+1} = W_i^t + \hat{\mathbf{r}}_\mathbf{i}^\mathbf{t}$ ▷ Update average rewards

Finally Figure 2.8 plots the individual costs for the players at the initial and the final (T) round. Clearly, a more uniform cost has emerged at time T for the 4 OD pairs and this compares well with the costs at equilibrium given by minimising Φ (indicated in red).

2.5 Chapter Summary

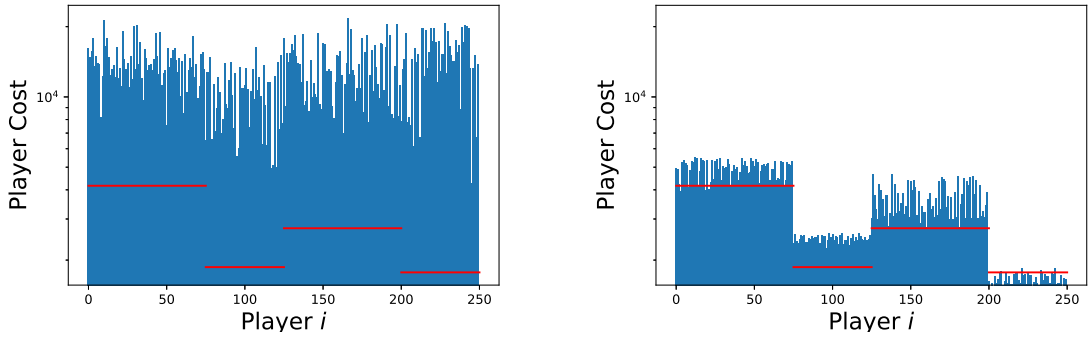
Algorithms for nonatomic selfish routing are still widely used in software such Emme [80] and PTV VISUM [137] for traffic planning means. The recent advances in bush-based methods present a major advance and efficiently make use of the concept of bushes and paired alternative segments. Their uptake in commercial software is



(a) Total Cost

(b) Cumulative Regret

Figure 2.7: Log-lin plots of total cost and cumulative regret for the 4 algorithms averaged over all test data.



(a) Player costs after initial round

(b) Player costs after T rounds

Figure 2.8: Log-lin plots illustrating the convergence of players costs for the 4 origin/destination pairs.

not universal due to their complexity, with software still preferring variants of the Frank-Wolfe method, reasons for this may be historical.

The gradient descent method implemented for the purpose of this thesis is a good option for analysis of mid-sized networks and is much faster than the Frank-Wolfe variants, although one should be mindful of the step size in networks with symmetric structure.

Algorithms for the atomic selfish routing games are sparse, perhaps due to their lack of a commercial application unlike their nonatomic counterpart. In this chapter a dynamic programming approach for parallel unweighted atomic selfish routing games was introduced which utilised the games potential function. For the standard

unrestricted network a tabu search metaheuristic and piecewise linear approximation method were proposed.

In addition some novel probabilistic algorithms based on the concept of bandit machines were outlined and experimental results demonstrated how these algorithms converge towards the equilibrium solution found by solving the corresponding game using the potential function. It is of interest that, with no knowledge of the underlying game and merely a probability distribution to assess the quality of the known paths, independent decision making has led to a reasonably stable state, mirroring how human decision making would happen and underlining the idea of user equilibrium.

2.5.1 Future Work

The major advances in shortest path technology present a burgeoning area of research and have large implications for nonatomic routing algorithms used in traffic planning. More work is required to analyse how much the shortest path technology can speed up these algorithms.

Whilst the concept of bounded rationality justified the idea of using a restricted path set for each user in an unweighted atomic routing game, a natural question to ask in the context of both nonatomic and atomic routing games is - how does restricting a commodity to a maximum of k paths affect both the speed and the quality of the solution?

The mixed-integer linear programming program with piecewise linear approximation as an objective provides a possible way to extend the routing game to include constraints on edges, vertices etc through the use of binary variables. Work is required to understand whether solving this is a viable option via conventional solvers using parallelisation. It is also worth considering models and methods which solve a mixed-integer nonlinear programming (MINLP) problem variant as recent research into solvers has greatly advanced, Section 5.1 provides more details.

Chapter 3

Multi-criteria Network Equilibrium: Efficiency and Suboptimality

Chapter Preface

Multi-criteria decision making is a powerful tool when considering optimisation of multiple and sometimes conflicting objectives. When dealing with the solutions to a multi-criteria problem, one must consider a set of solutions known as the set of Pareto optimal solutions.

This chapter presents work from the paper by O'Neill, Bagdasar, Berry, Popovici and Raja [131] (Appendix A.3) and its motivation is to investigate whether small changes to "free" network parameters result in a reduction of fuel consumption at user equilibrium, which in turn would result in monetary savings and reduced pollution. It also attempts to classify the suboptimality of the equilibrium solutions with respect to the Pareto front generated by a weighted sum model which is akin to the system optimal solution.

Chapter Keywords

Network Flow; User Equilibrium; Traffic Assignment; Multi-criteria Optimisation, Pareto Optimality, Price of Anarchy, Price of Stability

3.1 Introduction

Much work has been done in investigating the suboptimality of states of equilibrium arising from selfish routing. A key concept is the Price of Anarchy (PoA) (Section 1.2.1.5) which considers the ratio between the UE and SO, in effect it measures the degradation of the system state due to selfish behaviour. When the equilibrium solution is unique this measure is well-defined; however when multiple equilibria exist the best known equilibrium solution is compared to the system optimal solution, known as the Price of Stability (PoS) (Section 1.2.1.6). As stated in Section 1.2.2, the term *selfish routing* was coined by Roughgarden and Tardos [147] and the PoA was first introduced by Koutsoupias and Papadimitriou [89] to analyse the effects of selfish routing. The PoS was introduced by Anshelivich et al [5].

As aforementioned in Section 1.5, there are a number of difficulties in attempting to compare the equilibrium solutions given by a multi-criteria selfish routing problem and the corresponding social optimal solutions. Even if the assumptions given in Section 1.3.1, which result in a unique equilibrium solution, are adhered to, the PoA needs redefining when considering multi-criteria for the socially optimal solution as this will produce a Pareto set of optimal solutions. Therefore, there is not a unique value for the socially optimal problem and thus no unique PoA. If the assumptions in Section 1.3.1 are relaxed and there are now multiple equilibrium candidates, the aforementioned situation will become more difficult to define as there will now be a Pareto set of equilibrium solutions alongside a Pareto set of socially optimal solutions in n dimensional space (n being the number of criteria considered). It is important that the following questions are considered within the context of multi-criteria optimisation: What is a socially optimal solution?; What is an equilibrium solution?; How are these compared to assess the degradation of the network? Also, how are they measured?

This chapter is organised as follows. A simplified traffic assignment problem is introduced in Section 3.2.1 along with accompanying speed and fuel consumption estimation formulas. Proposed modelling approaches are discussed in Section 3.2.2, highlighting some of the associated issues. Section 3.2.3 outlines the models and algorithms used in this study and Section 3.2.4 discusses the computational results. Section 3.3 generalises the concept based on the PoA discussed and defined in Section

3.2.3.2. Finally Section 3.4 concludes the chapter, outlining potential future work.

3.2 A Study into a Multi-criteria Selfish Routing Problem Considering Fuel Consumption

3.2.1 Preliminaries

For convenience the nonatomic selfish routing game presented in Section 1.3.1 is recalled and assumption A3 is amended for the purposes of this Chapter.

3.2.1.1 Nonatomic Selfish Routing Game

Given a network $G = (V, E)$ and a set K of k OD pairs $\{(r_1, s_1), \ldots, (r_k, s_k)\}$. For each commodity (r_i, s_i) there is an amount of traffic d_i to be routed between r_i and s_i, the set of used paths by commodity (r_i, s_i) is denoted by Π_i. The set of all paths used for all commodities is then $\Pi = \bigcup_i \Pi_i$. Let f_π to to be the amount of flow routed along path π and x_e to be the amount of flow that induced onto edge e, $x_e = \sum_{\pi \in \Pi : e \in \pi} f_\pi$. The cost of an edge $t_e(x_e)$ is a function of the flow on the edge e, which is the aggregated flow from all paths that use edge e. A feasible flow $\mathbf{f}$ induces a flow $\mathbf{x} = (x_e)_{e \in E}$ and therefore the cost of a path π is $c_\pi = c_\pi(\mathbf{f}) = \sum_{e \in \pi} t_e(x_e)$.

An instance of a nonatomic selfish routing game is given by the triple (G, d, t) where $d = (d_j)_{j \in \{1,\ldots,k\}}$ is a vector of demands and $t = (t_e)_{e \in E}$ is a vector of edge travel time functions.

The fixed demand user equilibrium problem can be stated as the following optimisation problem [135]:

$$\text{minimise} \qquad U(\mathbf{x}) = \sum_{e \in E} \int_0^{x_e} t_e(\omega) d\omega \tag{3.1a}$$

subject to

$$\sum_{\pi \in \Pi_j} f_\pi = d_j, \qquad \forall j \in \{1, \ldots, k\} \tag{3.1b}$$

$$\sum_{\pi \in \Pi : e \in \pi} f_\pi = x_e, \qquad \forall e \in E \tag{3.1c}$$

$$f_\pi \geq 0, \qquad \forall \pi \in \Pi, \tag{3.1d}$$

subject to the three assumptions:

A1. The network is strongly connected.

A2. The traffic demand d_j is positive for all $j \in \{1, \ldots, k\}$.

A3. The travel time function $t_e : [0, \infty) \to (0, \infty)$ only depends on the flow on edge e, is twice differentiable, and satisfies the conditions $t_e^{'}(x_e) > 0$ and $t_e^{''}(x_e) \geq 0$, hence it is strictly increasing and convex.

To find a solution to Wardrop's second principle one can replace the objective function $U(\mathbf{x})$ with the objective function $T(\mathbf{x}) = \sum_{e \in E} x_e t_e(x_e)$, i.e. minimise the total travel time.

Assumption A3 ensures the uniqueness of the solution in terms of edge flows (see Section 1.3.3). It is common in the literature for a weaker variant of A3 to be used which guarantees uniqueness for the equilibrium problem, but not system optimal problem [135, 152].

3.2.1.2 User Equilibrium on a Single Commodity with Parallel Edges

Consider a multigraph G consisting of a single commodity (pair of vertices) connected by parallel edges. For this specific problem, the set of paths Π is equal to the set of edges E. This condition simplifies the optimisation problems and allows a solution via conventional convex optimisation, or in the discrete case, dynamic programming [9].

The graph given in Figure 3.1 consists of a set of m edges $E = \{1, 2, \ldots, m\}$, that connect, in parallel, the single commodity between the starting and terminal pair of vertices (s, t).

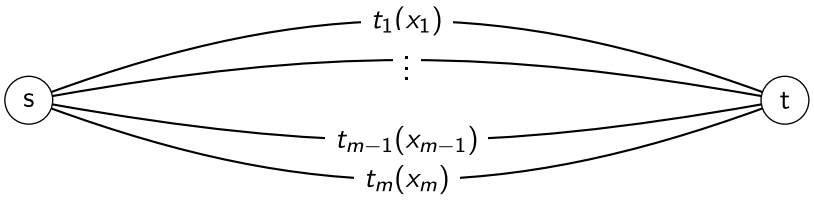


Figure 3.1: Single commodity with parallel edges.

For a demand r, the solution to the user equilibrium problem $\mathbf{x}^*_{\mathbf{UE}}$ displayed in Figure 3.1 can be found by solving the following optimisation problem:

$$\text{minimise} \qquad U(\mathbf{x}) \stackrel{\text{def}}{=} \sum_{e \in E} \int_0^{x_e} t_e(\omega) d\omega \tag{3.2a}$$

subject to

$$\sum_{e \in E} x_e = d \tag{3.2b}$$

$$x_e \geq 0, \qquad \forall e \in E. \tag{3.2c}$$

Replacing the objective (3.2a) with $T(\mathbf{x})$ minimises the total travel time and $\mathbf{x}^*_{\mathbf{SO}}$ is a solution to the system optimal problem.

3.2.1.3 Price of Anarchy

The PoA measures how suboptimal UE is compared to SO by comparing their respective total travel times for all vehicles. Let $\mathbf{x}^*_{\mathbf{UE}}$ be the optimal solution to UE and $\mathbf{x}^*_{\mathbf{SO}}$ be the optimal solution to SO, the price of anarchy is then defined as:

$$PoA = \frac{S(\mathbf{x}^*_{\mathbf{UE}})}{S(\mathbf{x}^*_{\mathbf{SO}})} \tag{3.3}$$

3.2.1.4 Travel Time Function

Traditionally, the most common choice for the travel time function has been a curve satisfying the assumption A3 given in (3.1), fitted by the Bureau of Public Roads [162]

$$t_e(x_e) = a_e \left(1 + 0.15 \left(\frac{x_e}{c_e}\right)^4\right), \tag{3.4}$$

where a_e is the free-flow time on a given edge e, c_e is the practical capacity as derived from the congestion conditions, and x_e denotes the current volume of traffic. The parameter values 0.15 and 4 are calibrated from the data.

3.2.1.5 Speed as a Function of Edge Flow

Denoting by d_e the length of edge e, the average speed at free-flow m_e is given by the formula $m_e = \frac{d_e}{a_e}$. The speed s_e as a function of the edge flow x_e is then given by

$$s_e(x_e) = \frac{d_e}{t_e(x_e)} = \frac{a_e m_e}{a_e \left(1 + b \left(\frac{x_e}{c_e}\right)^n\right)} = \frac{m_e}{1 + b \left(\frac{x_e}{c_e}\right)^n}. \tag{3.5}$$

Given that $t_e(x_e)$ is positive, continuous and strictly increasing, for a fixed d_e the function $s_e(x_e)$ is strictly decreasing on the interval $[0, \infty)$, from the maximum $s_e(0) = \frac{d_e}{s_e(0)}$ to the minimum $\lim_{x_e \to \infty} s_e(x_e) = 0$.

3.2.1.6 Fuel Consumption Curve

Consider a simplified fuel consumption model adapted from [122], which reflects the fundamental idea that in order to reduce fuel consumption, vehicles must drive within a given range, too slow or too fast resulting in increased fuel consumption. The free-flow speed values m_e (kph) in the model given in Table 3.1 correspond to common speed limits in the United Kingdom: 70, 60 and 40 miles per hour, respectively.

Edge (e)	m_e	d_e	c_e
1	110	160	600
2	95	130	500
3	65	80	300

Table 3.1: Edge travel time parameters.

Inspired by the general shape in Figure 1 in [122] and Figure 3 in [155], the fuel consumption curve is modelled by a strictly convex quadratic $y = Ax^2 + Bx + C$. The coefficients satisfy $A > 0$, $C > 0$, and $-2\sqrt{AC} < B < 0$, as in Figure 3.2. This function attains a minimum value $y^* = C - \frac{B^2}{4A}$, attained for $x^* = -\frac{B}{2A}$.

This model is largely consistent with the fuel consumption of an engine: larger for extreme values of speed, and optimal for moderate speed. For the purpose of the simulations, one assumes that all vehicles in the current model have the same parameters A, B, C. However, other parameter values would have to be considered for electric, hybrid, diesel, or petrol cars, of for engines of different sizes.

The speed $s_e(x_e)$ of a car travelling on edge e decreases from $m_e = s_e(0)$ (free-flow) to nearly 0, as the number of vehicles x_e increases. Correspondingly, the fuel consumption is given by

$$F_e(s_e(x_e)) = A[s_e(x_e)]^2 + B[s_e(x_e)] + C, \tag{3.6}$$

there have two cases. If $m_e \leq x^* = -\frac{B}{2A}$ (maximum allowed speed below optimum

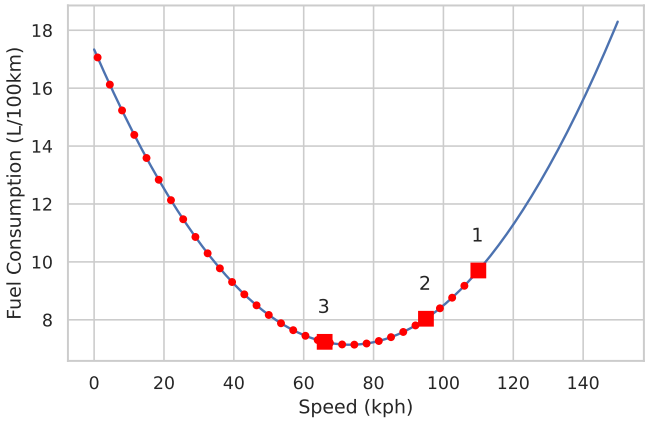


Figure 3.2: Fuel consumption curve $F_e(s_e)$ for $A = 0.0019$, $B = -0.2784$ and $C = 17.337$. The dotted lines indicate $F_e(s_e(x_e))$ (from right to left). The squares show $(m_e, F_e(m_e))$ for $m_1 = 110$, $m_2 = 95$ and $m_3 = 65$ kph (see Table 3.1).

speed for fuel consumption) as in Figure 3.2 squares 1 and 2, then the fuel consumption of individual vehicles will only increase with traffic. For $x^* = -\frac{B}{2A}$ the fuel consumption actually decreases as the traffic builds up to a moderate value, after which it increases as seen in Figure 3.2 square 3. The total fuel consumption on an edge e with x_e vehicles is then given by $x_e F_e(s_e(x_e))$, while the total fuel consumption of the network across all edges is

$$F(\mathbf{x}) = \sum_{e \in E} \frac{d_e}{100} \cdot x_e F_e(s_e(x_e)). \tag{3.7}$$

It should be noted that the purpose of these models and the fuel consumption curve given by Figure 3.2 is to investigate the questions listed in the introduction of this chapter. As aforementioned the quadratic curve is inspired by the general shape of Figure 1 in [122] and Figure 3 in [155] (full data and explanations are given in [122, 155] and the references therein) and is simplified to aid the computational approach to solving the models presented in this chapter. Given the absence of real data points for the simplified quadratic curve, an estimation of the errors is not possible and the quadratic should be seen as a means to exploring the ideas in this chapter.

To demonstrate the interplay between the travel time, speed and fuel consumption experienced by a single vehicle, the edge functions $t_e(x_e), s_e(x_e), F_e(s_e), x_e F_e(s_e)$ are plotted in Figs. 3.3a, 3.3b, 3.3c, 3.3d respectively, for the values of m_e, c_e and

d_e given in Table 3.1.

Figure 3.3c illustrates that the edges $1, 2$, both have the characteristic of exhibiting a minimum for fuel consumption for a demand $x_e > 0$ (the optimal speed zone which is optimal for fuel consumption is below the free-flow speed). On edge 3 fuel consumption strictly increases as vehicles can never make it into the optimal zone, hence the optimum fuel consumption is achieved for $x_e = 0$ (free-flow). In comparison the total fuel consumption (Figure 3.3d) along an edge is strictly increasing; however, it is noticeable from $x_1F_1(x_1)$ and $x_2F_2(x_2)$ that the rate of change slows for a demand $500 \leq x_e \leq 1250$, corresponding with the region in which the speed enters the optimal zone.

One must also be careful drawing conclusions for this simplified model as the saturation of the edge could, in reality (or a more complicated simulation), lead to increased congestion (and thus increased fuel consumption) because of braking, over-revving, etc. Also, as aforementioned, one would need to use different fuel consumption curves for different classes of vehicles (small cars, vans, trucks, etc).

3.2.2 Discussion of Potential Modelling Approaches

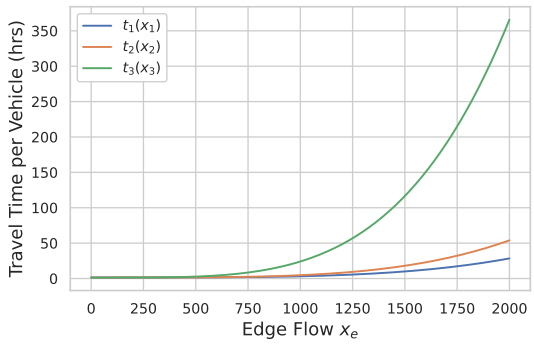
The following section looks at two possible approaches one may consider in relation to this problem: a weighted sum model (Section 3.2.2.1) and a constrained traffic assignment model (Section 3.2.2.2), for which the features, merits and flaws are discussed. Section 3.2.2.3 details two possible approaches one could use in order to overcome some of the issues raised.

3.2.2.1 Weighted Sum Model

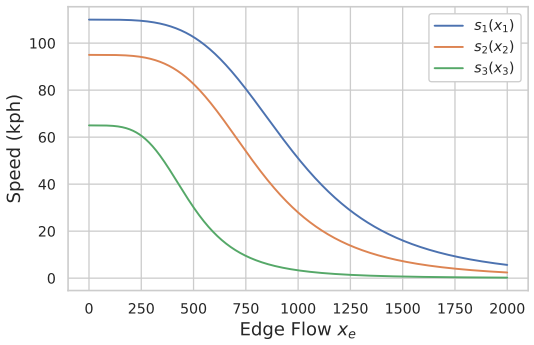
To examine the interplay between the total travel time $T(\mathbf{x})$ and fuel consumption $F(\mathbf{x})$, given the differences in scale, consider a normalised weighted sum model [163]. To this aim, given a scalar function $\phi : D \to (0, \infty)$, it will be convenient to consider the normalisation $\tilde{\phi} : D \to [0, 1)$ defined by

$$\tilde{\phi}(\mathbf{x}) = 1 - \frac{\inf_{\mathbf{y} \in D}(\phi(\mathbf{y}))}{\phi(\mathbf{x})}, \tag{3.8}$$

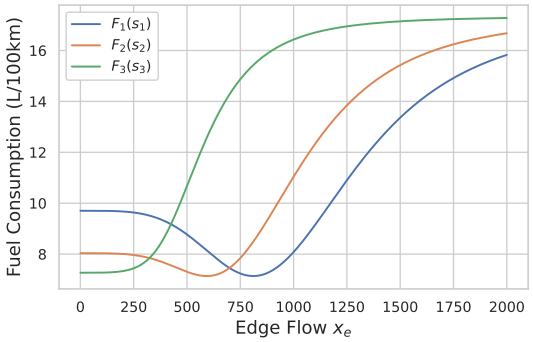
where it is understood that $\inf_{\mathbf{y} \in D}(\phi(\mathbf{y}))$ is computed a priori.



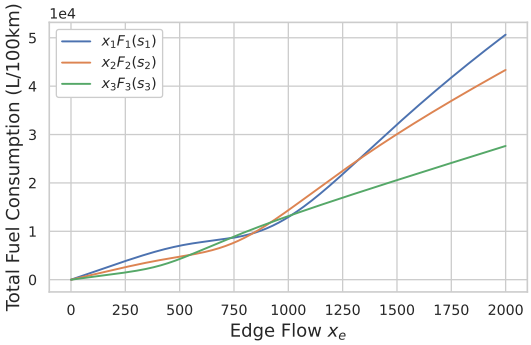
(a) Travel time per vehicle (hrs) - $t_e(x_e)$.



(b) Speed (kph) - $s_e(x_e)$.



(c) Fuel consumption per vehicle (L/100km) - $F_e(s_e(x_e))$.



(d) Total fuel consumption on edge (L/100km) - $x_eF_e(s_e(x_e))$.

Figure 3.3: Interplay between edge flow functions.

One may consider the weighted sum involving $\tilde{T}(\mathbf{x})$ and $\tilde{F}(\mathbf{x})$, given by

$$\lambda\tilde{T}(\mathbf{x}) + (1-\lambda)\tilde{F}(\mathbf{x}), \quad \lambda \in [0,1], \tag{3.9}$$

and try to minimise its value. Clearly, for $\lambda = 1$, the total travel time $T(\mathbf{x})$ is minimised, while for $\lambda = 0$ the total fuel consumption is minimised.

Another weighted sum involving normalised values of $U(\mathbf{x})$ (user equilibrium) and $F(\mathbf{x})$ (total fuel consumption) can be defined as

$$\lambda\tilde{U}(\mathbf{x}) + (1-\lambda)\tilde{F}(\mathbf{x}), \quad \lambda \in [0,1]. \tag{3.10}$$

Whilst this is valid as an objective, it ceases to make sense in the context of user equilibrium as the introduction of fuel consumption will have the effect of shifting the optimal solution away from user equilibrium (i.e. the travel times will not be equal according to Wardrop's first principle), it will in some senses act as central control. As aforementioned in the Introduction, this raises the question as to how

one might model user equilibrium with a competing objective; one idea that has been employed is by using the concept of VoT; however, this requires a conversion which is dependent on the opinions of individuals and careful consideration should be given as to whether one should include such normative (value) judgements.

3.2.2.2 Constraining by Total Fuel Consumption

An obvious extension to the problem is to include a constraint that limits fuel consumption.

$$\text{minimise} \qquad U(\mathbf{x}) \stackrel{\text{def}}{=} \sum_{e \in E} \int_0^{x_e} t_e(\omega) d\omega \tag{3.11a}$$

subject to

$$\sum_{e \in E} x_e = r \tag{3.11b}$$

$$\sum_{e \in E} F_e(s_e(x_e)) \leq C \tag{3.11c}$$

$$x_e \geq 0, \qquad \forall e \in E. \tag{3.11d}$$

The optimisation problem defined above can be viewed as a side constrained model [135], which satisfies a generalised Wardrop principle; however, difficulties are encountered with its interpretation as the constraints placed on the problem by the fuel consumption are not necessarily something an individual driver would consider when selfishly routing. In effect it will result in a new optimal point where travel times are now unequal, thus again Wardrop's first principle is violated.

3.2.2.3 Incentives to Change Behaviour

To overcome the issues presented in Sections 3.2.2.1 and 3.2.2.2, methods must be examined whereby individuals are coerced into making a different path choice which results in lower total fuel consumption. In a sense, similar to incentives in the market economy, somehow the selfish nature of individuals must be leveraged (preserving Wardrop's first principle) to reduce fuel consumption. Therefore, one must attempt to answer the question; why has anyone got any incentive to change routes?

Two possible approaches that can be considered:

- Change the network, e.g. add/remove new edges (i.e. Network Design).
- Adapt the controllable parameters of the network, e.g. change the free-flow speed limits of existing roads (edges).

In the remainder of this study, the second approach is considered; however, the first approach is a rich area of research with much work done in network design under budgetary constraints utilising such techniques as bi-level optimisation as aforementioned in the Section 1.5.

3.2.3 Models

To address the second approach outlined in Section 3.2.2.3, two models presented, the first a weighted sum model which can be used to analyse the interplay between total travel time and total fuel consumption; the second utilises a change in the travel time function, allowing a search to be undertaken for the set of free-flow speeds for which solutions satisfy user equilibrium.

3.2.3.1 Weighted Sum Model

As previously noted form a normalised weighted sum and minimise:

$$\begin{aligned} &\text{minimise} && \lambda\tilde{T}(\mathbf{x}) + (1-\lambda)\tilde{F}(\mathbf{x}), && \lambda \in [0,1] && \text{(3.12a)} \\ &\text{subject to} \\ & && \sum_{e\in E} x_e = r && && \text{(3.12b)} \\ & && x_e \geq 0, && \forall e \in E. && \text{(3.12c)} \end{aligned}$$

Given the definition in equation (3.8), the normalised minimum values of $\tilde{F}(\mathbf{x})$ and $\tilde{T}(\mathbf{x})$ are 0. The utopian (ideal) point [59, 172, 173] $\mathbf{z} \in \mathbb{R}^2$ is defined as

$$\mathbf{z} = \left(\min_{\mathbf{x}}(\tilde{F}(\mathbf{x})), \min_{\mathbf{x}}(\tilde{T}(\mathbf{x}))\right) = (0,0).$$

If $\mathbf{x}^*_\lambda$ is the optimal solution for a given value of λ to (3.12), then denote the solution which is nearest to the utopian point by $\mathbf{x}^*_{\lambda_0}$. This satisfies the condition

$$\mathbf{x}^*_{\lambda_0} = \arg\min_{\mathbf{x}} \left(\left\| \left(\tilde{F}(\mathbf{x}), \tilde{T}(\mathbf{x}) \right) - \mathbf{z} \right\|_2 \right) = \arg\min_{\mathbf{x}} \left(\left\| \left(\tilde{F}(\mathbf{x}), \tilde{T}(\mathbf{x}) \right) \right\|_2 \right). \quad \text{(3.13)}$$

An approximation of $\mathbf{x}^*_{\lambda_0}$ can be found by computing the set X_λ of N solutions to (3.12), namely $\mathbf{x}^*_\lambda$, $\forall\lambda \in [0, \frac{1}{N}, \frac{2}{N}, \ldots, 1]$. An approximation is then given by

$$\mathbf{x}^*_{\lambda_0} \approx \arg\min_{\mathbf{x}\in X_\lambda} \left(\left\| \left(\tilde{F}(\mathbf{x}), \tilde{T}(\mathbf{x}) \right) - \mathbf{z} \right\|_2 \right) = \arg\min_{\mathbf{x}\in X_\lambda} \left(\left\| \left(\tilde{F}(\mathbf{x}), \tilde{T}(\mathbf{x}) \right) \right\|_2 \right). \quad (3.14)$$

3.2.3.2 Measuring the Inefficiency of User Equilibrium against the Weighted Sum Model

The inefficiency of the user equilibrium solution $\mathbf{x}^*_{\mathbf{UE}}$ can also measured for a given configuration of the network by comparing it to the solution that is nearest, in Euclidean distance, to the solution within the Pareto set $\mathbf{x}^*_{\lambda_0}$ (note that normalised functions are used to allow comparison across network configurations). Define the measure ρ to be the ratio

$$\rho = \frac{\left\| \left(\tilde{F}(\mathbf{x}^*_{\lambda_{UE}}), \tilde{T}(\mathbf{x}^*_{\lambda_{UE}}) \right) - \mathbf{z} \right\|_2}{\left\| \left(\tilde{F}(\mathbf{x}^*_{\lambda_0}), \tilde{T}(\mathbf{x}^*_{\lambda_0}) \right) - \mathbf{z} \right\|_2}. \quad (3.15)$$

The minimum value of this ratio is 1 and is achieved when $\mathbf{x}^*_{\lambda_{UE}} = \mathbf{x}^*_{\lambda_0}$.

As this measure is analogous to the PoA/PoS in a single criteria context, similar behaviour is expected in a multi-criteria context. However, unlike the single criteria scenario, equilibrium solutions at low demand and high demand may not converge with the system optimal solution.

The following explanation provides an insight into why this is the case. It is clear that in an sparsely populated network (i.e. little demand) users will not have an incentive to switch paths as the shortest path will not change. In the case of an over saturated network (i.e. very high demand above the capacity), users will not have an available option to switch as all available paths will be congested and thus at equilibrium. Without intervention, the equilibrium solution will likely remain suboptimal with respect to any criteria that is in opposition of the total travel time. Clearly, the objective of user equilibrium is for each user to selfishly minimise their travel time, i.e. there may exist paths that are more costly in terms of travel time but satisfy the other criteria, e.g. save fuel, cut down emissions.

While situations in which $\mathbf{x}^*_{\lambda_{UE}} \approx \mathbf{x}^*_{\lambda_0} \implies \rho \approx 1$ are likely to occur more frequently at low or high demand due to either users not wanting to switch or having no choice, the additional condition required is that the network is designed

such that the path choices for equilibrium are good path choices for all criteria examined. In reality some criteria are always likely to be suboptimal and a value of $\rho \approx 1$ will scarcely be attainable. A more reasonable interpretation is that low values of ρ suggest that a network is configured in such a way that the equilibrium solution is a good solution to all the criteria examined. Section 3.2.4.2 explores this reasoning and provides an example which shows that this is not necessarily the case and one should be careful in the interpretation of ρ when comparing different networks.

3.2.3.3 Adjusting the edge free-flow speed

Define the optimisation problem by adjusting the travel time function to allow different values of the free-flow time a_e (note the distance d_e is fixed and cannot change). Utilising the relationship $m_e = \frac{d_e}{a_e}$, then $a_e = \frac{d_e}{m_e}$ and the travel time function can be written as

$$t_e(x_e) = \frac{d_e}{m_e}\left(1 + b\left(\frac{x_e}{c_e}\right)^n\right). \tag{3.16}$$

The values m_e can now be manipulated and, for a given value the optimisation problem given by 3.1 can be solved. The optimisation problem is considered as a sub-problem of the main search which seeks to minimise the total fuel consumption by manipulating the values of m_e.

Algorithm 6 Steps for generating global minimum UE solutions

Step 1 $\rightarrow$ Set m_e, $\forall e \in E$

Step 2 $\rightarrow$ Solve user equilibrium sub-problem

Step 3 $\rightarrow$ Store solution

Step 4 $\rightarrow$ Repeat 1-3 until all desired combinations of $\{m_e\}_{e\in E}$ have been explored.

In Step 2 of algorithm 6, m_e is fixed and the sub-problem is just a specific case of 3.1 and has a unique global solution.

It should be noted that this method is not tractable in larger networks as the number of configurations is $|E|^k$, where k is the number of choices m_e for a given edge e; however, it is easily parallelisable and does allow exploration into the pattern of behaviour.

3.2.4 Results and Discussion

In the first part of this section results are presented that are generated for the weighted sum model which demonstrates the interplay between travel time and fuel consumption. Results are then presented, utilising the same network, for the adjusting of the edge free-flow speed limits m_e to analyse the impact this has on the total travel time and total fuel consumption for solutions solved for user equilibrium.

Results were generated using the 3 edge parallel network, with values of d_e and c_e given by Table 3.1, and for a demand value $r = 1500$, i.e. 1500 vehicles should be routed across the network. The fuel consumption curve (3.6) had values of $A = 0.0019$, $B = -0.2784$ and $C = 17.337$ and is depicted in Figure 3.2.

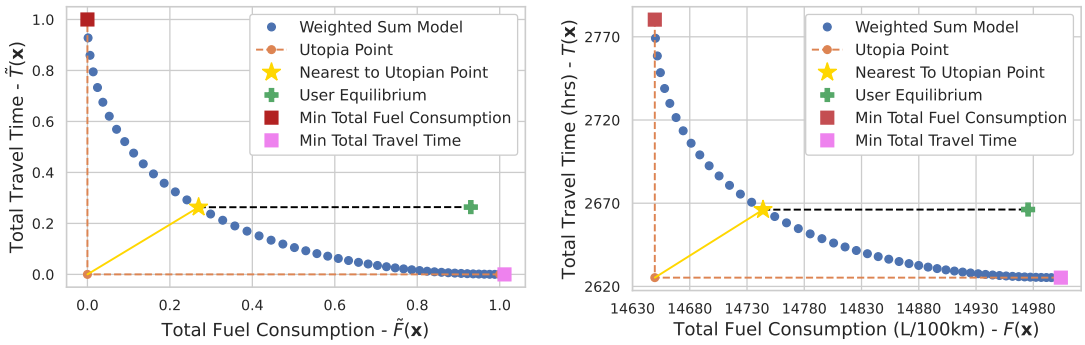
3.2.4.1 Weighted Sum Model

The results of the optimisation for values of $\lambda \in [0, 1]$ and the three configurations of $\mathbf{m} = (m_1, m_2, m_3)$, $\mathbf{m}^1, \mathbf{m}^2, \mathbf{m}^3$ are given in Table 3.2. Figure 3.4 shows a comparison of the normalised results for $\mathbf{m}^1$ and the results in the original scale. Figure 3.5 plots results for $\mathbf{m}^1, \mathbf{m}^2, \mathbf{m}^3$ in the original scale, in addition to the weighted sum model, the minimum total travel time, minimum fuel consumption and user equilibrium are also plotted. The point $(F(\mathbf{x}^*_{\lambda 0}), T(\mathbf{x}^*_{\lambda 0}))$ is also included as a means of measuring how far the user equilibrium is from a more idealised point in the Pareto set.

$\mathbf{m}$	$\mathbf{x}$	x_1	x_2	x_3	$T(\mathbf{x})$	$F(\mathbf{x})$	λ_0	$\phi(\mathbf{x})$	ρ
$\mathbf{m}^1$	$\mathbf{x}^*_{\lambda_0}$	715.02	450.7	334.28	2666.13	14744.0	0.3	11120.64	1.33
	$\mathbf{x}^*_{UE}$	661.79	444.67	393.54	2666.21	14975.43	-	-	-
$\mathbf{m}^2$	$\mathbf{x}^*_{\lambda_0}$	499.22	585.28	415.51	2470.71	13957.74	0.14	12349.56	2.26
	$\mathbf{x}^*_{UE}$	425.83	583.19	490.98	2622.48	14077.74	-	-	-
$\mathbf{m}^3$	$\mathbf{x}^*_{\lambda_0}$	732.13	495.34	272.53	2941.01	15287.75	0.34	11089.86	2.09
	$\mathbf{x}^*_{UE}$	764.47	567.52	168.01	3044.31	15441.731	-	-	-

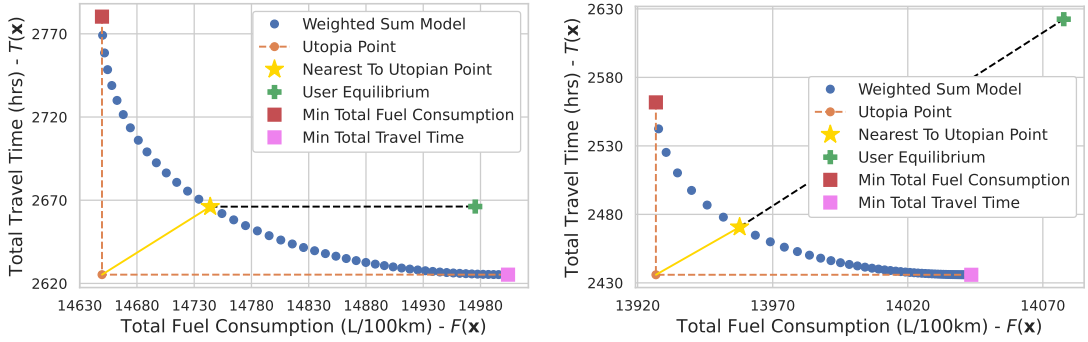
Table 3.2: Comparison of weighted sum model solutions for $\mathbf{m}^1 = (110, 80, 65)$, $\mathbf{m}^2 = (95, 95, 95)$ and $\mathbf{m}^3 = (110, 80, 40)$.

The Pareto set has extreme values corresponding to the minimum values of $T(\mathbf{x})$ and $U(\mathbf{x})$ and in each case the user equilibrium solution is dominated by

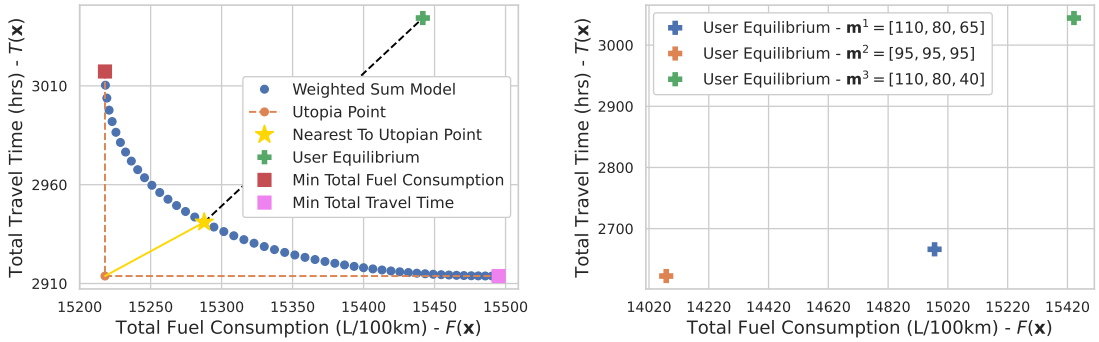


(a) Normalised plot for $\mathbf{m}^1 = (110, 95, 65)$. (b) Original scale plot $\mathbf{m}^1 = (110, 95, 65)$.

Figure 3.4: Comparison of normalised and original scale plots.



(a) $\mathbf{m}^1 = (110, 95, 65)$. (b) $\mathbf{m}^2 = (95, 95, 95)$.



(c) $\mathbf{m}^3 = (110, 80, 40)$. (d) $\mathbf{m}^1$ vs $\mathbf{m}^2$ vs $\mathbf{m}^3$.

Figure 3.5: Weighted sum model for 3 different configurations of $\mathbf{m}$.

solutions of the weighted sum model. This is not surprising as the average time drivers experience by selfishly routing (user equilibrium) can never do better than the minimum total travel time found by minimising the total travel time or the minimum fuel consumed by minimising fuel consumption. Whilst it can be the case that for certain demand levels, i.e. extremely low or extremely high, the total travel

time for selfishly routing is equal to the minimum total travel time; the solution to user equilibrium, at best would correspond to a point in the Pareto set.

3.2.4.2 Inefficiency of User Equilibrium against the Weighted Sum Model

One should be careful not to conclude that a configuration where the user equilibrium solution is close to this point is better (ρ is smaller) as they are different problems for each configuration. On inspection of Figs. 3.5a, 3.5b and 3.5c, the user equilibrium solution $\mathbf{x}^*_{\mathbf{UE}}$ that is furthest from the solution $\mathbf{x}^*_{\lambda_0}$ is given by configuration $\mathbf{m}^2$ with the largest value of $\rho = 2.26$ (Figure 3.5b); however, Figure 3.5d clearly shows that this solution dominates both configurations $\mathbf{m}^1$ and $\mathbf{m}^3$ whose value of ρ is smaller than the one given by $\mathbf{m}^2$. To summarise, whilst ρ measures the inefficiency of user equilibrium when compared with system optimal within a given network, clearly it does not provide a means as to evaluating the quality of the solution with respect to the objectives $T(\mathbf{x})$ and $F(\mathbf{x})$.

3.2.4.3 Adjusting the Edge Free-flow Speed

A simple iterative search was used to generate a set of user equilibrium solutions, furthermore, the Pareto set for the solutions that are efficient for total travel time and fuel consumption were identified. For each solution, the variance of the edge travel time functions, which have positive flow (i.e. those used), was also generated to demonstrate the validity of the user equilibrium claim. The maximum value of the variance over all solutions was 3.87×10^{-10}.

The results are displayed in Figure 3.6 and clearly show the trade-off between total travel time and total fuel consumption. Points 1, 2, 3 on the graph indicate the increase in allowable speeds on the edges; at point 1 for low speeds the total fuel consumption and total travel time are both high, as the speeds increase and approach point 2, the optimal zone of the fuel consumption curve is entered, thus the total fuel consumption is minimised. Finally, as point 3 is approached, speeds increase and the optimal zone is exited, allowing the total travel time to be lowered, but severely impacting the total fuel consumption.

Two additional points are plotted showing how a reasonable adjustment in the values of m_e can result in a new user equilibrium solution that strictly dominates in

terms of both total travel time and total fuel consumption. Figs. 3.7a and 3.7b plot

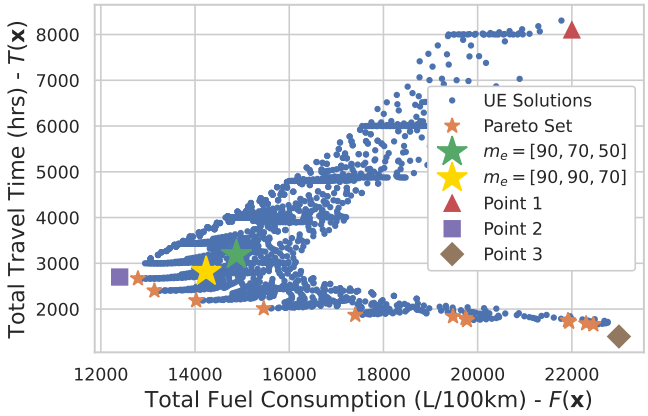
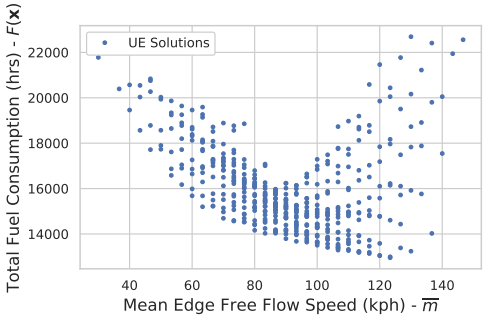
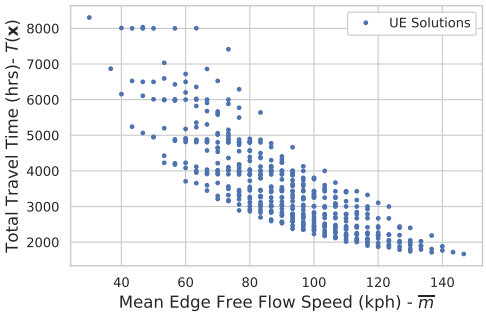


Figure 3.6: Total fuel consumption vs total travel time for varying free-flow speeds m_e.

the total fuel consumption and the total travel time against the mean edge speed $\overline{m} = \frac{1}{|E|}\sum_{e \in E} m_e$, respectively. Each of the figures are intuitive with the expected behaviour, in the case of Figure 3.7a, the total fuel consumption, it can be seen that this mirrors the fuel consumption curve $F_e(s_e)$, when the mean speed is too low or too high there is an increase in the total fuel consumption, the best results are obtained between the values of $\overline{m} = 60$ and $\overline{m} = 80$. In Figure 3.7b, the total travel time can be seen to decrease as the speed limit is progressively lifted.



(a) Mean edge free-flow speed $\overline{m}$ vs Total fuel consumption



(b) Mean edge free-flow speed $\overline{m}$ vs Total travel time.

Figure 3.7: Decomposition of user equilibrium solutions plotted against mean edge speed $\overline{m}$.

3.3 Measuring the Inefficiency of Equilibrium

Motivated by the ideas expressed in Section 3.2.3.2 and defined in equation (3.15), a more general outline of the inefficiency measure for multi-criteria nonatomic selfish routing games is given.

Given a congestion game (G, d, t) for which there exists a set of equilibrium solutions X_E, and a set of n functions by which the quality of the solution can be measured, the general multi-criteria optimisation problem is stated as,

$$\begin{aligned} &\min(f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_n(\mathbf{x})) \\ &\text{s.t.} \quad \mathbf{x} \in X, \end{aligned} \tag{3.17}$$

where X is the set of feasible set of solutions.

A feasible solution $\mathbf{x_1} \in X$ dominates $\mathbf{x_2} \in X$ if:

1. $f_i(\mathbf{x_1}) \leq f_i(\mathbf{x_2})$, for all $i \in 1, 2, \ldots, n$, and
2. $f_i(\mathbf{x_1}) < f_i(\mathbf{x_2})$ for at least one index $j \in 1, 2, \ldots, n$

The set of Pareto optimal solutions contains all feasible solutions $\mathbf{x}^*$ which are not dominated.

The ideal solution is defined to be $\mathbf{z} = (\inf_{\mathbf{x}\in X} f_1(\mathbf{x}), \inf_{\mathbf{x}\in X} f_2(\mathbf{x}), \ldots, \inf_{\mathbf{x}\in X} f_n(\mathbf{x}))$. Let $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_n(\mathbf{x}))$, be the n-dimensional vector of functions evaluated at $\mathbf{x}$ and $d(\mathbf{p}, \mathbf{q})$ be a distance measure between the n-dimensional vectors $\mathbf{p}$ and $\mathbf{q}$.

Therefore, the point closest to $\mathbf{z}$ as measured by $d(\mathbf{p}, \mathbf{q})$ is $\mathbf{x}^*_\mathbf{z} = \operatorname{argmin}_{\mathbf{x}\in X} d(\mathbf{f}(\mathbf{x}), \mathbf{z})$ and ρ_d can be defined for a given user equilibrium solution $\mathbf{x_E} \in X_E$ as the ratio of - the distance between the ideal point and user equilibrium; and the distance between the ideal point and the feasible solution closest to the ideal point. Figure 3.8 illustrates the concept for a two function multi-criteria problem using Euclidean distance. The inefficiency of an equilibrium solution $\mathbf{x_E}$, $\rho_d(\mathbf{x_E})$ is defined as

$$\rho_d(\mathbf{x_E}) = \frac{d(\mathbf{f}(\mathbf{x_E}), \mathbf{z})}{d(\mathbf{f}(\mathbf{x}^*_\mathbf{z}), \mathbf{z})}. \tag{3.18}$$

The definition given in Section 3.2.3.2 and given by equation (3.15) is recovered by letting $d(\mathbf{p}, \mathbf{q}) = \|\mathbf{p} - \mathbf{q}\|_2$, $\mathbf{f}(\mathbf{x}) = (\tilde{F}(\mathbf{x}), \tilde{T}(\mathbf{x}))$ and $\mathbf{x_E}$ be the unique solution to the optimisation problem (3.2).

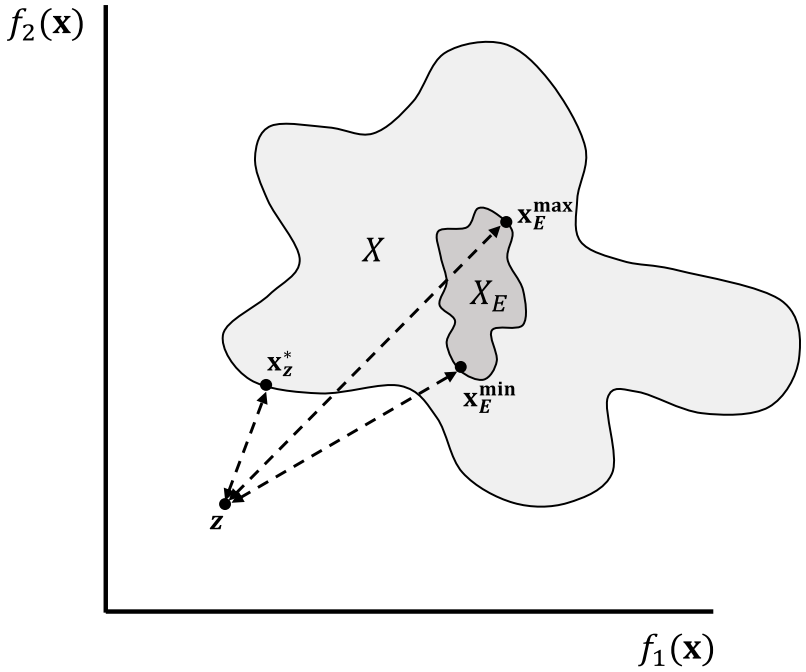


Figure 3.8: Illustration of the ratios described by equation (3.18) using Euclidean distance between points.

It is important to note that in Section 3.2.3.2 the solution to equilibrium $\mathbf{x_E}$ is unique and therefore ρ_d is uniquely defined. In the more general case, whereby multiple equilibrium solutions exist, the PoA and PoS as defined in Sections 1.2.1.5 and 1.2.1.6 can be mirrored by considering the following Pareto set of solutions.

1. The Pareto set of equilibrium solutions $X_E^{min} \subseteq X_E$ that are not minimally dominated, i.e. A feasible solution $\mathbf{x_1} \in X$ minimally dominates if

   (a) $f_i(\mathbf{x_1}) \leq f_i(\mathbf{x_2})$, for all $i \in 1, 2, \ldots, n$, and

   (b) $f_i(\mathbf{x_1}) < f_i(\mathbf{x_2})$ for at least one index $j \in 1, 2, \ldots, n$

2. The Pareto set of equilibrium solutions $X_E^{max} \subseteq X_E$ that are not maximally dominated, i.e. A feasible solution $\mathbf{x_1} \in X$ maximally dominates if

   (a) $f_i(\mathbf{x_1}) \geq f_i(\mathbf{x_2})$, for all $i \in 1, 2, \ldots, n$, and

   (b) $f_i(\mathbf{x_1}) > f_i(\mathbf{x_2})$ for at least one index $j \in 1, 2, \ldots, n$

Thus, one can measure any equilibrium point within either set using $\rho_d(\mathbf{x_E})$. Considering the PoA (Section 1.2.7) which considers the maximum equilibrium the fol-

lowing can be defined as a candidate for multi-criteria,

$$\rho_d^{max} = \rho_d(\mathbf{x_E} \in X_E^{max})$$

and for the PoS (Section 1.2.8),

$$\rho_d^{min} = \rho_d(\mathbf{x_E} \in X_E^{min}).$$

As above it is possible to give these a unique candidate by choosing the closest solution to $\mathbf{z}$ as measured by $d(\mathbf{p}, \mathbf{q})$. That is define the minimum equilibrium candidate $\mathbf{x_E^{min}} = \underset{\mathbf{x} \in X_E^{min}}{\operatorname{argmin}}\, d(\mathbf{f}(\mathbf{x}), \mathbf{z})$ and for the maximum candidate $\mathbf{x_E^{max}} = \underset{\mathbf{x} \in X_E^{max}}{\operatorname{argmax}}\, d(\mathbf{f}(\mathbf{x}), \mathbf{z})$. This is visually illustrated in Figure 3.8.

The choice of the ideal point z as a point to which to measure from is a natural candidate; however, it should be noted that in some situations the 'ideal' solution to (3.17) (or the one chosen in practice) may be different and careful consideration would be needed in defining means of measuring the inefficiency of equilibrium.

3.4 Chapter Summary

In this chapter a bi-criteria extension of a simplified single OD pair connected with parallel edges to alleviate fuel consumption was presented alongside a discussion of the issues with the multi-criteria approach of converting criteria to Value of Time, i.e. when such a conversion can be deemed difficult to make sense of within the context of selfish routing.

Further, two approaches were presented, the first, a weighted sum model, allowed the interplay between the competing objectives of total fuel consumption and total travel time to be explored and a means of measuring the inefficiency of the equilibrium solutions with the weighted sum model demonstrated; the second, through the manipulation of the free-flow speed of an edge, demonstrated the Pareto set of solutions and the possibility of non-invasive traffic planning to reduce the total fuel consumption whilst still maintaining a respectable total travel time under the notion of selfish routing, i.e. adjusting the existing speed limits to reach the desired goal.

Finally, a more general definition of the PoA and the PoS in the context of multi-criteria problems was discussed and concepts illustrated.

3.4.1 Future Work

In future work, to explore the second approach further, one can use a larger test network that is not limited to parallel edges, e.g. Sioux Falls [96]. This would require the development of a more tractable approach to allow a further investigation into the effect that the changes to free-flow speeds through either a meta-heuristic approach to bi-level optimisation or a MILP whereby a subset of edges would be selected that represented an optimal solution to the problem. A basic implementation of this approach can be found at [129]. The basic nonatomic selfish routing formulation could be expanded to include edge-capacity constraints [125] and emissions modelling via pricing schemes [104].

Finally, the concept presented in Section 3.3 requires further analysis and investigation within both the context of congestion games and the wider area of game theoretic concepts.

Chapter 4

Nonatomic Selfish Routing: Assessing Demand-Based Importance Measures

Chapter Preface

In applications such as traffic routing, there is a need to assess the network resilience and identify components that may have a detrimental impact on the routing of network flow. As the travel cost in selfish routing games is dependent on the equilibrium flow, standard network science measures which do not account for network flow do not necessarily provide insight into which components of the network are important.

This chapter analyses appropriate existing measures for selfish routing games based on travel cost and proposes an alternate measure. It also queries the necessity of these measures and their use.

Chapter Keywords

Importance Criticality, Robustness, Vulnerability, Resilience, Network Science

4.1 Introduction

The pervasiveness of networks within our daily lives naturally leads to the question of robustness of a network in performing its main function. In the case of nonatomic selfish routing games, this can be framed as assessing the degradation of the network's ability to route demand when changes are made to the network, i.e. can the importance of network components be predicted ahead of time? It is also natural to consider when changes improve the ability to route demand.

In view of the statement above, two key questions arise: what is meant by assessing the ability to route demand?; what is meant by changes to the network?

When considering the first question, in the case of nonatomic selfish routing games and their equilibrium, there is an obvious (albeit not unique) choice, the total congestion (e.g. total travel time). The principal idea of equilibrium in a nonatomic selfish routing game is that commodities are routed such that they seek to minimise their own personal total congestion and adhere to Wardrop's first principle.

> The journey times on all the routes actually used between an origin and a destination are equal, and less than those which would be experienced by a single vehicle on any unused route.

Therefore, it is natural to consider the change in total travel time at equilibrium due to some change in the network.

In this chapter, the focus will be on the state of equilibrium in a nonatomic selfish routing game and the associated total travel time found from solving the optimisation problem outlined in Section 1.3.1. Whilst the focus is on equilibrium, this work could be applied to the state of system optimal if the situation so required. It is also noted that the consideration of the total travel time is not particularly special, and this work could be applied to anything that can be represented as a congestible function, like server latency etc.

The second question has a number of candidates, namely: demand profile, edge congestion functions and the underlying network topology, i.e. vertices and edges. Whilst changes to the demand and edge congestion functions are valid areas of investigation, large changes to the demand profile, or improvements to edge capacity etc are not overly common compared with possible outages of the network vertices

and edges, that is when the vertices or edges that are no longer functioning, e.g. road closures, computer vertex failures, electrical cables. Therefore, the focus here is on assessing how a network holds up when there are disruptions to the underlying network topology.

It seems intuitive that an edge that carries a lot of flow is critical to the network, that is, does the removal of the edge hinder the users of the network in terms of their travel time (average travel time is just a function of the total travel time).

Assessing vertices is also important and there are two types of vertex that need to be considered. Vertices that either emit or absorb flow (belonging to a commodity) and those that do not. Those vertices that do not emit or absorb flow can be measured in the same manner as the edges; understanding what to measure in terms of the vertices that belong to a commodity is important and more challenging owing to the question of what to do with the demand associated with them.

This chapter is organised as follows. In Section 4.2 some network science measures, which are not demand-based, are introduced and demonstrated via a case study of all UK major cities and towns (excluding London due to its size). These measures are also analysed to ascertain whether they can be utilised to predict the importance of network components. Section 4.3 outlines and analyses existing demand-based measures for network equilibrium flows and details the issues present in these measures. Section 4.4 proposes an alternate measure and related theorems and a statistical analysis of relevant measures is given to ascertain the usefulness of these measures when applied to nonatomic selfish routing games. Section 4.5 examines some paradoxical effects. Finally Section 4.6 utilises the proposed measure to assess whether the positioning of external demand on a grid-based network has an impact on the importance of the edges and vertices. All analysis undertaken in Section 4.3, 4.4 and 4.6 was done using the Python libraries Pandas [159] and Seaborn [78, 168] and full data and implementation of results can be found at [129]. Equilibrium solutions for networks analysed were solved using the gradient projection algorithm outlined in Section 2.2.3.2 to an AEC of 10^{-5} or a maximum of 500 iterations.

4.2 Network Science Measures

Network science is a continually burgeoning field that studies the means to statistically classify particular phenomena within networks, such as their topology. Measures exist such as centrality, degree and clustering, which can be useful to rank edges and vertices and provide a good insight into pertinent questions such as network connectivity, network resilience, importance of vertices (e.g. Google's PageRank) and the spread of information or disease [106, 149].

4.2.1 Local Measures

The following measures act to describe the structure at a local (micro) level.

- Clustering coefficient - *how close its neighbours are to being a clique*
- Weighted clustering coefficient - *as above but taking edge weighting into account*
- Centrality measures (normalised)
  - Degree - *how many edges are adjacent to the node*
  - Closeness - *how close a vertex is to any other vertex in the network*
  - Betweenness - *how many shortest paths pass through the node*
- Eccentricity - *maximum shortest path between a vertex and any other node, i.e the distance of a vertex from the vertex most distant to it*

4.2.2 Global Measures

The following measures act to describe the structure at a global (marcro) level.

- Average degree - *mean of all the vertex degrees*
- Average clustering coefficient - *mean of all the clustering coefficients*
- Average weighted clustering coefficient - *mean of all the weighted clustering coefficients*
- Degree distribution - *fraction of vertices in the network with degree k*

- Radius - *minimum of all vertex eccentricities*
- Diameter - *maximum of all vertex eccentricities, i.e. the maximum of all shortest paths*

4.2.3 Relevant Measures

Measures which are used in analysis are defined in this section; for all these measures assume that, for a commodity j with OD pair (u, v), the shortest path distance $d(u, v)$ is the weighted path given by,

$$d(u, v) = \min_{\pi \in \Pi_j} \sum_{e \in \pi} t_e(x_e), \tag{4.1}$$

where Π_j is the set of paths available between u and v.

A full explanation of the measures presented can be found in [23, 25].

4.2.3.1 Vertex Measures

Degree Centrality

Definition 4.2.1 (Degree Centrality)**.** *For a graph $G = (V, E)$, the degree centrality for the vertex v is the number of edges incident.*

$$C_D(v) = deg(v).$$

In the case of a directed graph, the number of edges incident is the sum of the incoming and outgoing edges.

Definition 4.2.2 (Normalised Degree Centrality)**.** *For a graph $G = (V, E)$, the normalised degree centrality for the vertex v is the fraction of vertices it is connected to.*

$$\hat{C}_D(v) = \frac{C_D(v)}{|V| - 1}.$$

Closeness Centrality

Definition 4.2.3 (Closeness Centrality)**.** *For a graph $G = (V, E)$, the closeness centrality for vertex v is the reciprocal of the sum of the shortest path distances for the $n - 1$ reachable vertices for vertex v,*

$$C_C(v) = \frac{1}{\sum_u d(u, v)},$$

where $d(u, v)$ is the shortest path distance for vertices u and v.

Definition 4.2.4 (Normalised Closeness Centrality)**.** *For a graph $G = (V, E)$, the normalised closeness centrality for vertex v is the reciprocal of the average shortest path distance for the $n - 1$ reachable vertices for vertex v,*

$$\hat{C}_C(v) = (n-1) \cdot C_C(v).$$

Betweenness Centrality

Definition 4.2.5 (Betweenness Centrality)**.** *For a graph $G = (V, E)$, the betweenness centrality for vertex v is the fraction of all-pairs shortest paths passing through vertex v,*

$$C_B(v) = \sum_{s,t \in V : s \neq t \neq v} \frac{\sigma_{st}(v)}{\sigma_{st}},$$

where σ_{st} is the number of shortest paths from vertex s to vertex t and $\sigma_{st}(v)$ is the number of shortest paths passing through vertex v.

Definition 4.2.6 (Normalised Betweenness Centrality)**.** *For a directed graph $G = (V, E)$, the normalised betweenness centrality for vertex v is the fraction of all-pairs shortest paths passing through vertex v,*

$$\hat{C}_B(v) = \frac{C_B(v)}{(n-1)(n-2)},$$

where $(n-1)(n-2)$ is the number of s, t pairs in the graph G

Note that this allows comparison between strongly connected directed graphs. However, in a directed graph where there exists no path between some pair of vertices, s and t, $\sigma_{st} = 1$ and $\sigma_{st}(v) = 0$. In this case one should be careful of comparing networks.

Clustering Coefficient

Definition 4.2.7 (Clustering Coefficient)**.** *For a directed graph $G = (V, E)$ the clustering coefficient is given by,*

$$C_W(v) = \frac{1}{C_D(v)(C_D(v) - 1) - 2C_D^{\Leftrightarrow}(v)} T(v),$$

where T(v) is the set of directed triangles through vertex v and $C_D^{\Leftrightarrow}(v)$ is the number of neighbouring vertices for which a bi-directional edge exists, i.e. edge (u, v) and (v, u) are present.

Note that the denominator counts all possible directed triangles that can be formed; in the case of a undirected graph this is just the number of pairs of vertices that are incident to v, $\frac{C_D(v)(C_D(v)-1)}{2}$. In the case of a directed graph, this is replaced by the denominator and $2C_D^{\leftrightarrow}(v)$ is a correction factor which accommodates for false triangles which have been counted by $C_D(v)(C_D(v)-1)$ [61].

Eccentricity

The eccentricity of a vertex v is the maximum distance from v to all other vertices in the network G,

$$\epsilon(v) = \max_{u \in V} d(v, u).$$

4.2.3.2 Edge Measures

Edge Betweenness Centrality

Definition 4.2.8 (Edge Betweenness Centrality)**.** *For a graph $G = (V, E)$, the edge betweenness centrality for edge e is the fraction of all-pairs shortest paths passing through edge e,*

$$C_B(e) = \sum_{s,t \in V : s \neq t \neq v} \frac{\sigma_{st}(e)}{\sigma_{st}},$$

where σ_{st} is the number of shortest paths from vertex s to vertex t and $\sigma_{st}(e)$ is the number of shortest paths passing through edge e.

Definition 4.2.9 (Normalised Betweenness Centrality)**.** *For a directed graph $G = (V, E)$, the normalised edge betweenness centrality for edge e is the fraction of all-pairs shortest paths passing through edge e,*

$$\hat{C}_B(e) = \frac{C_B(e)}{(n-1)(n-2)},$$

where $(n-1)(n-2)$ is the number of s, t pairs in the graph G

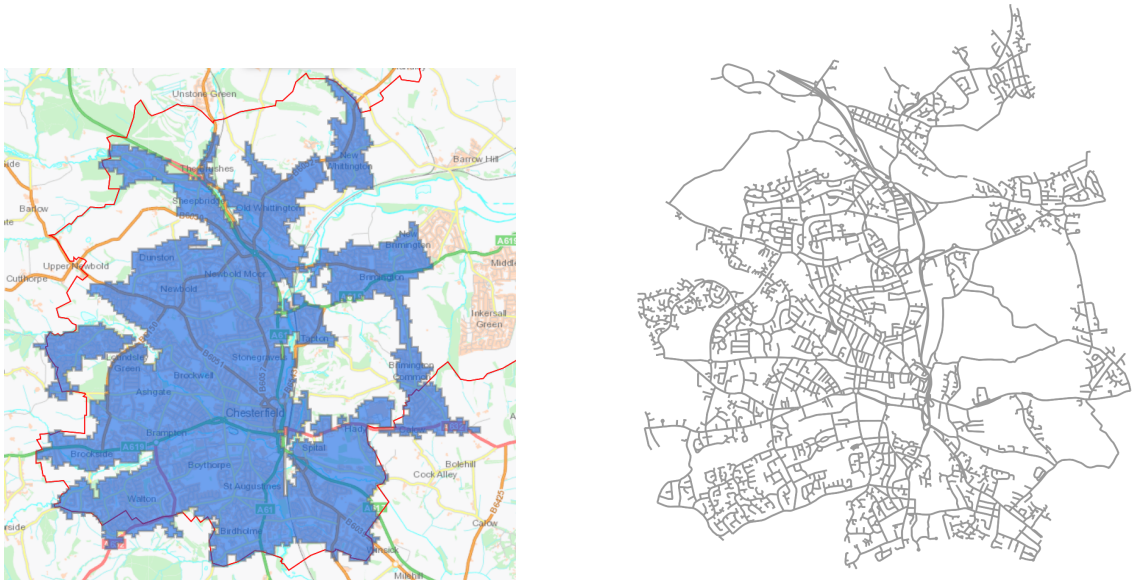
4.2.4 A Case Study Using OpenStreetMap

To showcase the aforementioned measures and also provide insight into the structure and size of one of the most widely used applications of nonatomic selfish routing games, urban traffic networks (traffic planning purposes), an analysis of 111 major cities and towns in England and Wales is given.

The data for this study was extracted from OpenStreetMap (OSM) via the python libraries NetworkX (NX) and OSMNX, a Python package to retrieve, model, analyze, and visualize street networks from OSM [24]. The borders used for each major city and town is based on the built-up areas (BUA) 2011 census definitions [128]. This provided better boundaries for urban areas when compared with the traditional local authority district (LAD). The convex hull of the boundary was used to extract a connected directed graph of the roads which was then used in the statistical analysis. All data can be found at [129]

A major omission from the dataset is London as it is an outlier because of its density and size and dramatically skews the results. Figures 4.1a and 4.1b illustrate the BUA boundary for the town of Chesterfield and the corresponding extracted network from OSM used for analysis. Figure 4.2 illustrates four local vertex measures for Chesterfield which are coloured low (light green) to high (dark blue). Table 4.1 summarises a comparison between the town of Basildon and Chesterfield. Table 4.2 displays the overall central tendency and dispersion measures for all towns and cities and Figure 4.3 presents a Pearson correlation matrix assessing the linear association between the central tendency and dispersion measures. For a detailed explanation of the Pearson correlation coefficient see Chapter 7 of [26]. Finally, Figure 4.4 provides examples of degree distributions for selected towns and cities.

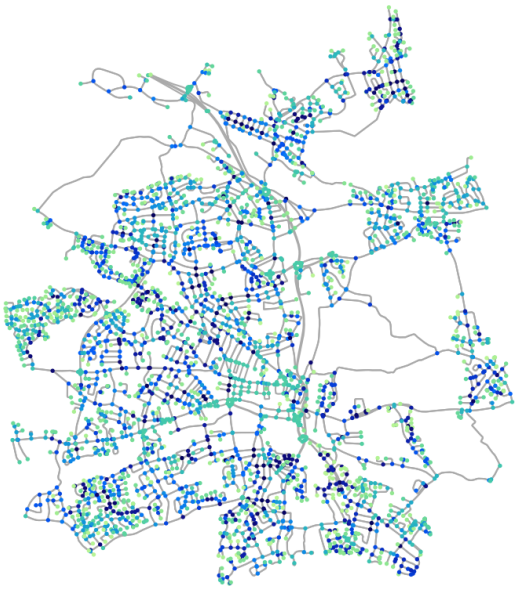
4.2.4.1 Example City Analysis



(a) Chesterfield BUA.

(b) Chesterfield road network.

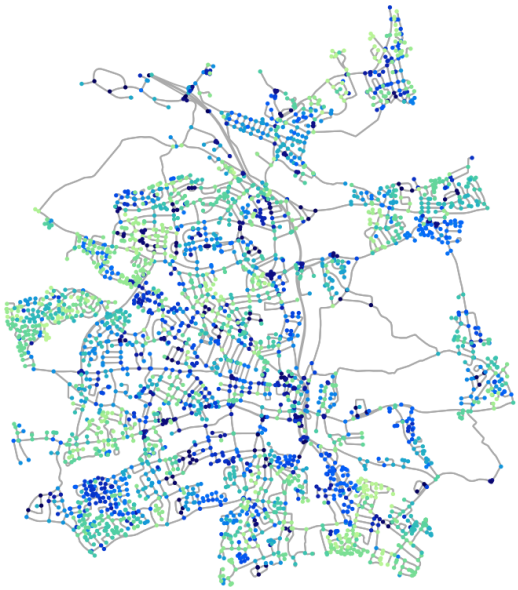
Figure 4.1: Chesterfield BUA and network extracted via OSNMX.



(a) Degree centrality.

(b) Closeness centrality.

(c) Betweenness centrality.



(d) Weighted clustering coefficient.

Figure 4.2: Vertex measures for Chesterfield.

City	Avg neighbor degree avg	Avg betweenness centrality	Avg closeness centrality	Avg clustering coefficient	Avg clustering coefficient (weighted)	Avg degree centrality
Basildon	2.5589	0.01073	0.0002	0.0291	0.0006	0.0012
Chesterfield	2.6950	0.01078	0.0003	0.02549	0.0009	0.0014

City	Edge density ($\frac{\text{km}}{\text{km}^2}$)	Avg edge length	Avg Node Degree	No. of edges (m)	No. of nodes (n)	Node density (km)
Basildon	15661.3703	90.7818	4.1397	6990	3377	83.3459
Chesterfield	15548.0810	89.8760	4.5108	7197	3191	76.7023

Table 4.1: Global averages of measures for Basildon vs Chesterfield.

4.2.4.2 Analysis on all England and Wales Cities

	μ	σ	min	median	max	$\frac{\sigma^2}{\mu}$
Avg of avg neighbour degree	2.666	0.064	2.507	2.660	2.849	0.002
Convex hull area (km^2)	74,592.521	160,442.301	18,345.047	43,979.944	1,696,460.132	345,098.029
Avg clustering coefficient	0.034	0.007	0.023	0.034	0.054	0.001
Avg clustering coefficient (weighted)	0.001	0.000	0.000	0.001	0.002	0.000
Avg degree centrality	0.001	0.000	0.000	0.001	0.002	0.000
Edge density (km^2)	17,707.888	3,074.378	11,384.039	17,491.264	26,825.076	533.762
Avg edge length	84.923	9.112	61.009	85.148	108.510	0.978
Avg Node Degree	4.461	0.184	4.047	4.459	4.954	0.008
No. of edges (m)	15,249.384	29,777.625	4,466.000	9,109.000	310,578.000	58,147.069
No. of nodes (n)	6,766.277	12,629.505	1,978.000	4,075.000	131,429.000	23,573.436
Node density (km^2)	95.194	22.058	51.928	93.658	152.199	5.111

Central Tendency and Dispersion measures could be used for comparisons between larger areas (e.g. countries)

Table 4.2: Central tendency measures for UK.

Correlation Matrix

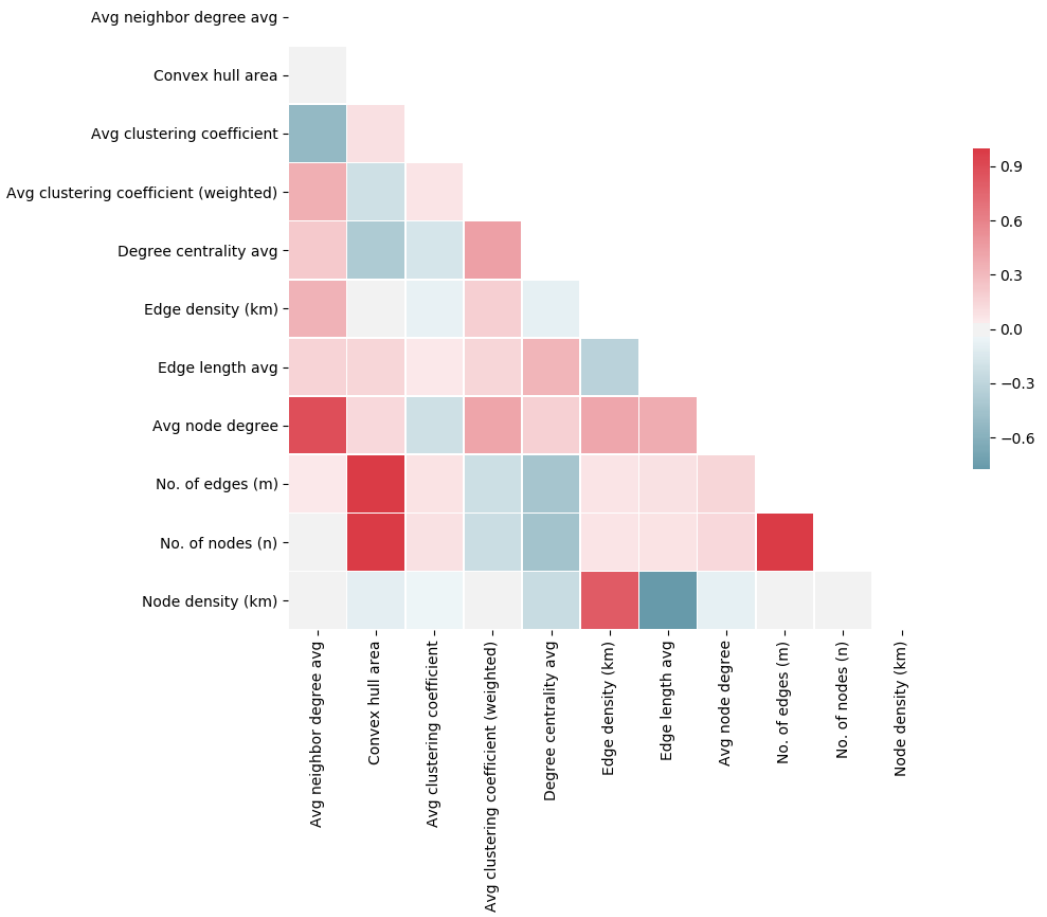


Figure 4.3: Pearson correlation matrix for selected measures for UK BUA's.

Degree Distributions

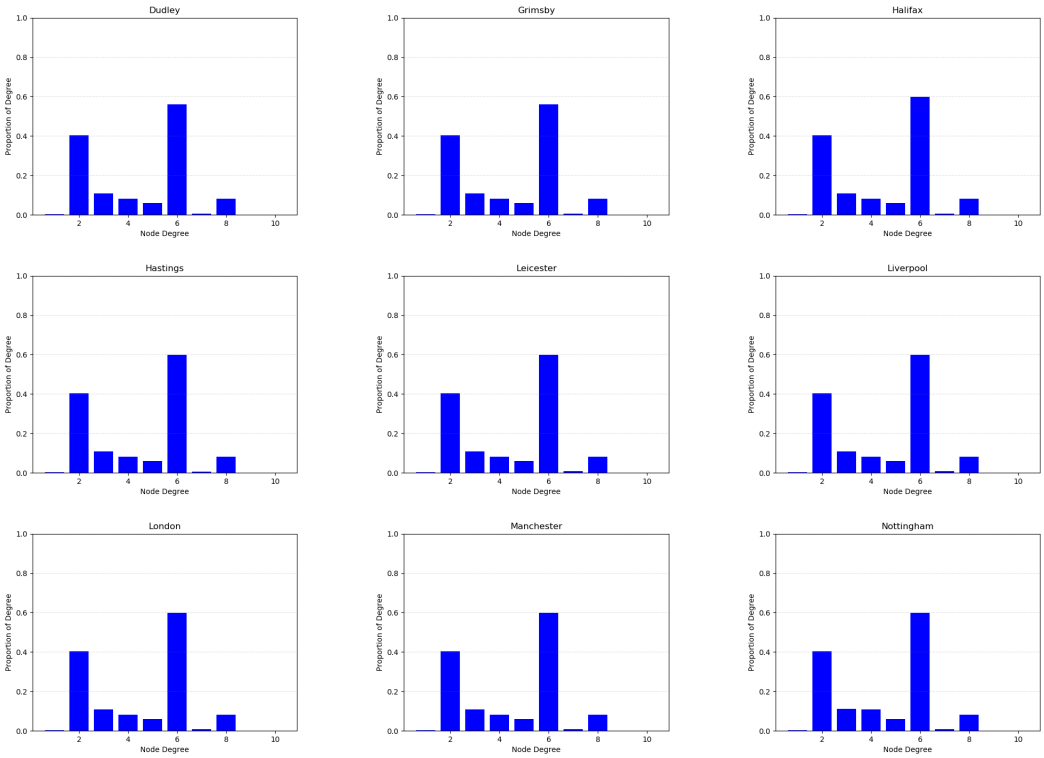
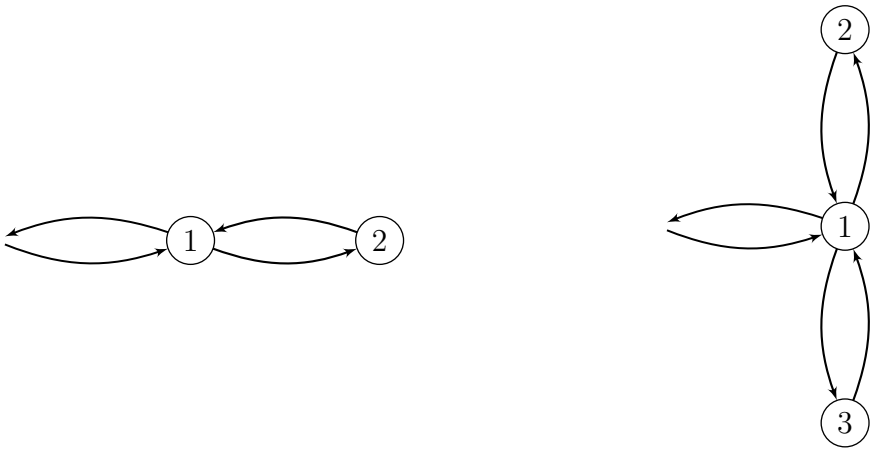


Figure 4.4: Degree distributions of selected cities.



(a) Vertex 2 is a terminating vertex with a degree of 2

(b) Vertex 1 is a three-way junction with a degree of 6.

Figure 4.5: Explanation of vertex degree values of 2 and 6.

4.2.4.3 Conclusions and Future Work

Table 4.2 and Figure 4.4 illustrate that the UK road networks are sparse networks when considering the mean vertex degree and other such measures. On inspection of the degree distributions, it is apparent that the most frequent vertex degrees are 2 and 6, and this is supported by the mean vertex degree $\mu = 4$. The explanation for the frequency of a value of 2 is due to the ending of roads as illustrated in Figure 4.5a. One can postulate that the value of 6 is also easily explained by the presence of three-way junctions (Y junction or T junction) as illustrated in Figure 4.5b. It is also notable that a value of 8 is less common which would be attributable to a four-way junction (intersection).

Whilst networks that have organically grown, such as those in the UK road network, are somewhat unstructured, the analysis here supports a suggestion that there is a pseudo grid-like topology. It also establishes the mean size of the network equilibrium problem for urban transportation networks in the UK with a mean value of 15,249.38 and 6,766.28 for the number of edges and nodes, respectively (see Table 4.2). Section 5.3.6 outlines the current state of shortest path technology in large network problems, easily dealing with ∼200K vertices and ∼1M edges. This puts the size of the problem into context and demonstrates that at some point in the future there may be less of a reliance on the need for bespoke methods as outlined in Chapter 2.

It is possible, given the data that has been extrapolated, to classify the road network using supervised or unsupervised learning. In particular, cluster analysis could be used to attempt to classify networks into categories within the UK, or even globally. Of note are networks which are designed and therefore exhibit much greater structure. Examples include grid plan networks such as those used in the US (e.g. Manhatten) or in the UK (e.g. Milton Keynes).

Additional questions can also be asked with regard to selfish routing, in conjunction to these classifications, as to whether a more structured network has benefits with regards to robustness and traffic management.

4.2.5 Statistical Analysis of Network Science Measures on Sioux Falls

This section analyses how the local centrality network science measures presented in Section 4.2.3 correlate with the change in total travel time ΔT, when either an edge or a vertex is removed, to assess whether these measures can be used as a predictor for ranking which edges or vertices are important to the network.

Analysis is done for the Sioux Falls network (Appendix C.3) and the network proposed by Dial [57] (Appendix C.4) and includes the following measures. Note that, given the analysis in Section 4.2.4, urban transportation networks are sparse and, therefore, clustering measures are not included in this analysis; however, in denser networks they may prove useful. Three of these measures require the edge weight (cost) and are computed for three different cases, $i = 1, 2, 3$ representing:

1. Edges have equal cost
2. Edge costs are the congestible edge functions computed on an empty network
3. Edge costs are the congestible cost of edge functions when the network is at equilibrium.

Edge Measures

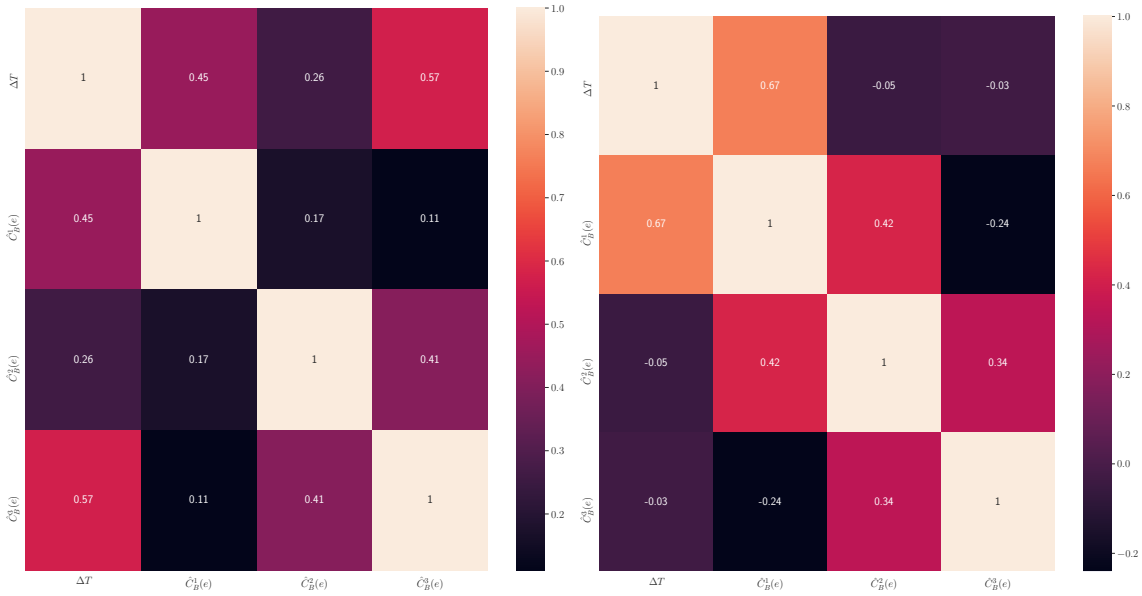
- $\hat{C}_B^i(e)$ - Normalised Edge Betweenness Centrality

Vertex Measures

- $\hat{C}_B^i(v)$ - Normalised Betweenness Centrality
- $\hat{C}_C^i(v)$ - Normalised Closeness Centrality
- $\hat{C}_D(v)$ - Normalised Degree

Figures 4.6a and 4.7a display the Pearson correlation matrix for Sioux Falls for the removal of a vertex and the removal of an edge, respectively. Figures 4.6b and 4.7b display the Pearson correlation matrix for Dial's Network. Figures 4.6a and 4.6b indicates a low positive correlation between $\hat{C}_B^1(e)$, $\hat{C}_B^3(e)$ and ΔT on removal of an

edge. Similarly for the removal of a node, Figures 4.7a and 4.7b suggest that $\hat{C}_B^2(v)$, $\hat{C}_B^3(v)$ and $\hat{C}_C^3(v)$ correlate well with ΔT.

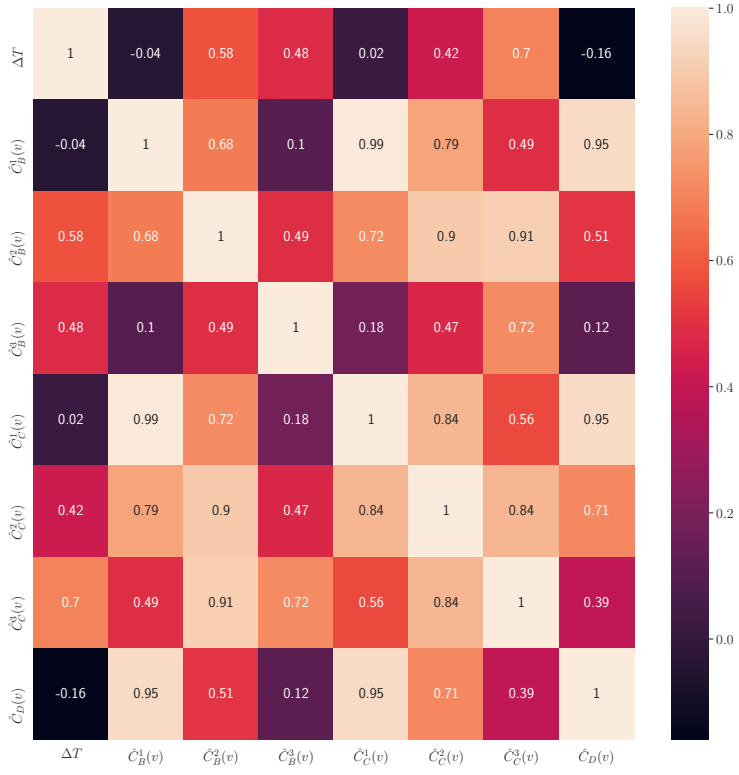


(a) Sioux Falls network.

(b) Dial's network.

Figure 4.6: Pearson correlation matrix for network science measures (removal of an edge e).

(a) Sioux Falls network.



(b) Dial's network.

Figure 4.7: Pearson correlation matrix for network science measures (removal of a vertex v).

4.3 Demand-based Measures

Section 4.2.5 motivates the need for measures that correlate better with both the change in total travel time ΔT and the flow on an edge x_e. Whilst the measures demonstrated in Section 4.2.4 can potentially provide answers for many questions involving networks, they do not take into account the resulting flow present on a network due to the routing of demand. Therefore, they are unlikely to be able to fully answer questions around robustness and criticality.

It is, therefore, important to consider measures that take demand and flow into account and what the desirable properties of these measures should be. This section outlines these considerations and presents existing demand-based measures that provide a means of assessing the importance of a component with regards to ΔT and the flow in the network.

4.3.1 What Should a Reasonable Measure Involving Congestible Flow Show?

The following are six desirable properties of an effective measure:

1. Reflects the importance of the removal of a set of vertices/edges from a network;
2. Normalised to allow comparison between different networks;
3. Bounded;
4. Weighted by the demand profile to allow comparison with different demand loadings;
5. Identify if the removal of the set of vertices/edges has resulted in a disconnect in the demand;
6. Reflects changes to demand.

4.3.2 Existing Demand-based Measures

As aforementioned in Section 1.6, a comprehensive overview of such measures can be found in the survey done by Jafino, Kwakkel and Verbraeck (2019) [81].

The measures presented in this section make use of the shortest path distance, $d(u, v)$, between a pair of vertices u and v given by (4.1).

4.3.2.1 Importance

The importance is defined in different ways [84, 102]; however, it can be loosely defined as the impact that the removal of a given component or set of components have on a network.

More precisely, it will measure the change in a given network measure $E_M(G)$ for the network G to the revised network G'.

Definition 4.3.1 (Normalised Importance of a Set of Network Vertices and Edges)**.** *Given a nonatomic selfish routing game (G, d, t) and a subgraph (G', d, t), let S represent the set of removed vertices and edges from $G = (V, E)$ and $G' = (V', E')$,*

$$S = \{V \setminus V' \cup E \setminus E'\}.$$

The importance of the set S, $I(S)$ is measured by the relative network efficiency drop after S is removed from the network G:

$$I(S) = \frac{\Delta E_M}{E_M} = \frac{E_M(G, d) - E_M(G', d)}{E_M(G, d)},$$

where $E_M(G, d)$ and $E_M(G', d)$ are the network efficiency measures for the nonatomic selfish routing game (G, d, t) before and after the removal of S.

It is important to note that if the denominator of $E_M(G, d)$ and $E_M(G', d)$ is the same, then it plays no part in the measure. Therefore, it is important to clarify if and how a denominator changes.

4.3.2.2 Latora and Marchiori (L-M) Measure

Latora and Marchiori [94] introduced a measure to assess how efficiently a network exchanges information. Studies of the measures are provided on neural networks, man-made communication (internet) and transportation systems (Boston subway transportation system). The first two are considered as unweighted graphs, i.e. edges do not have a weight (distance), whereas the transportation system is a weighted graph that uses the physical distance of the geographical locations between stations. The L-M measure is defined as follows.

Definition 4.3.2 (L-M Measure)**.** *The network efficiency measure $E_{LM}(G)$ for a given network $G = (V, E)$, is defined as:*

$$E_{LM}(G) = \frac{1}{n(n-1)} \sum_{u,v \in V : u \neq v} \frac{1}{d(u,v)},$$

where $n = |V|$. If there is no path between u and v, $d(u,v) = \infty$.

If vertices are removed in a subgraph G' then the original number of vertices n is still used.

Note that this is a more general definition and does not take into account the demand profile of a nonatomic selfish routing game. Therefore, the point at which shortest paths $d(u,v)$ are computed must be determined. To compare with the other measures, $d(u,v)$ will be computed after the network has been loaded and is at equilibrium.

4.3.2.3 Demand-weighted Total Travel Time

The following measure was proposed by Zhu in 2006 [174] and is fairly straightforward. That is, weight the total travel time by the demand routed in the network.

Definition 4.3.3 (Zhu et al)**.** *The network efficiency measure $E_Z(G, d)$ for a given network $G = (V, E)$, is defined as:*

$$E_Z(G, d) = \frac{\sum_{k \in K} \lambda_k d_k}{\sum_{k \in K} d_k},$$

where for a given commodity $k = (u, v)$, $\lambda_k = d(u, v)$, i.e. the shortest path for commodity k.

At equilibrium, all demand for a commodity $k \in K$ will be routed on the shortest path and, therefore, the numerator is equal to the total travel time,

$$\sum_{k \in K} \lambda_k d_k = \sum_{e \in E} x_e t_e(x_e).$$

Thus, at equilibrium $E_Z(G, d)$ can be seen as measuring the demand weighted total travel time of the network.

As per the L-M measure, the above is computed after the network has been loaded and in the case that a commodity k is disconnected, the measure is undefined, $E_Z = \infty$.

Given that it is improbable (Braess Paradox) that the removal of a set of network vertices/edges S results in an improvement to the overall total travel time. The normalised importance for $E_Z(G,d)$ is defined as,

$$I_Z(S) = \frac{\Delta E_Z}{E_Z} = \frac{E_Z(G',d) - E_Z(G,d)}{E_Z(G,d)} = \frac{\sum_{c\in K} \lambda'_k d_k - \sum_{c\in K} \lambda_k d_k}{\sum_{c\in K} \lambda_k d_k}$$

thus as $E_Z(G',d) \to \infty$, the measure is unbounded above,

$$\lim_{E_Z(G',d)\to\infty} \frac{E_Z(G',d) - E_Z(G,d)}{E_Z(G,d)} = \lim_{E_Z(G',d)\to\infty} \frac{E_Z(G',d)}{E_Z(G,d)} - 1 = \infty.$$

4.3.2.4 Edge Importance Adapted from Jenelius, Petersen, and Mattson (2006)

The original definitions given in [84] are limited to edges, but can be extended to fit with the notion of removing a set of vertices/edges from the network.

Definition 4.3.4 (Global Importance)**.**

$$I_G(S) = \frac{1}{n_K} \sum_{k\in K} \left(\lambda_k(G') - \lambda_k(G)\right),$$

where $\lambda_k(G)$ is the shortest path for commodity k for network G and n_k is the number of commodities $|K|$.

The global importance is the difference between the total sum of the shortest paths weighted by the number of commodities. A major omission to this measure is that it does not account for the demand in the network and, therefore, it is not considered for analysis.

Definition 4.3.5 (Demand-Weighted Importance)**.**

$$I_{DW}(S) = \frac{\sum_{k\in K} d_k \left(\lambda_k(G') - \lambda_k(G)\right)}{\sum_{k\in K} d_k},$$

where $\lambda_k(G)$ is the shortest path for commodity k for network G.

The demand-weighted importance is merely the difference between the total travel time at equilibrium, it shares the numerator with the Zhu importance (I_Z), but is weighted by the total demand for the network. Whilst similar to the Zhu importance, this measure is not normalised and does not adhere to the second desirable property outlined in Section 4.3.1 and is not part of the subsequent analysis.

Definition 4.3.6 (Relative Unsatisfied Demand)**.**

$$I_{UD}(S) = \frac{\sum_{k \in K} u_k(G')}{\sum_{k \in K} d_k},$$

where $u_k(G')$ is the resulting unsatisfied demand in G' from removing a set S of network vertices/edges from G.

The relative unsatisfied demand is an important concept and will be invaluable in determining the reasons for improvements in total travel time and ensuring the delivery of demand. It measures the fraction of the demand that has not been routed due to a disconnect in the network or removal of a vertex attached to a commodity.

4.3.2.5 Nagurney-Qiang (N-Q) Measure

Nagurney and Qiang introduced their demand-based measure in 2008 [119] in an attempt to provided an importance ranking measure which considered the demands and flows on a network. Further considerations were given in their text on fragile networks [120].

Definition 4.3.7 (N-Q Measure)**.** *The network efficiency measure $E_{NQ}(G, d)$ for a given network $G = (V, E)$, is defined as:*

$$E_{NQ}(G, d) = \frac{\sum_{k \in K} \frac{d_k}{\lambda_k}}{n_K},$$

where for a given commodity $k = (u, v)$, $\lambda_k = d(u, v)$, i.e. the shortest path for commodity k and n_k is the number of commodities in the network $|K|$.

If a commodity $k \in K$ becomes disconnected, all demand of the commodity k, d_k is assigned to a path of infinite length, $\lambda_k = \infty$.

If a commodity is removed, all demand of the commodity k, d_k is assigned to a path of infinite length, $\lambda_k = \infty$ and n_K is the original number of commodities in the network $|K|$

The last part of the definition which specifies the case for a disconnect of a commodity is necessary to ensure that the measure is well defined. As the $\lim_{\lambda_k \to \infty} \frac{d_k}{\lambda_k} = 0$, a disconnected commodity can be seen to contribute nothing to the measure.

$E_{NQ}(G, d)$ can be seen to represent the average demand per unit cost over the set of commodities. The bigger the increase to the values of λ_k, $\forall k \in K$ in G', the smaller the value of $E_{NQ}(G', d)$. By definition as $E_{NQ}(G', d) \to 0$, $I_{NQ}(S) \to 1$.

4.3.3 Some Motivating Examples

To demonstrate the importance measures I_{LM}, I_Z, I_{NQ} and I_{UD} and provide insight, the following examples are given for the removal of either a single vertex or a single edge. The change in total travel time is reported as ΔT, with a value of $\Delta T > 0$ corresponding to a worsening of the total travel time and $\Delta T < 0$ corresponding to an improvement of the total travel time. The unsatisfied demand I_{UD} is also given. As the section develops, issues with the existing demand-based measures are mentioned and, motivated by this, solutions to these issues are sought. When removing an edge, a comparison will also be made with the flow on an edge x_e at equilibrium in the original network (no removal of edges). In the case of a vertex v, a comparison with all the flow incident to v in the original network with be made. Let E_v be the set of edges incident to vertex v, then the flow incident to vertex v is given by $x_e^v = \sum_{e \in E_v} x_e$.

4.3.3.1 Braess Network

Figure 4.8 displays the Braess network which consists of single commodity $(1, 4)$. A full description of the network and edge cost functions can be found in Appendix C.1.

The Braess network is an important example as the adding of an edge can worsen the travel time at equilibrium, $\Delta T > 0$. The converse is also true, i.e. starting with the edge present and removing the edge results in a improvement of the total travel time ($\Delta T < 0$) at equilibrium. Therefore, a measure should be able to detect this. Results are summarised in Tables 4.3 and 4.4.

u	v	x_e	I_{LM}	I_Z	I_{NQ}	I_{UD}	ΔT
1	2	4000.00000	0.25345	0.06250	0.05882	0.00000	20000.00000
1	3	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	3	4000.00000	-0.42070	-0.18750	-0.23077	0.00000	-60000.03815
2	4	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
3	4	4000.00000	0.25345	0.06250	0.05882	0.00000	20000.00000

Table 4.3: Edge importance measures for Braess network.

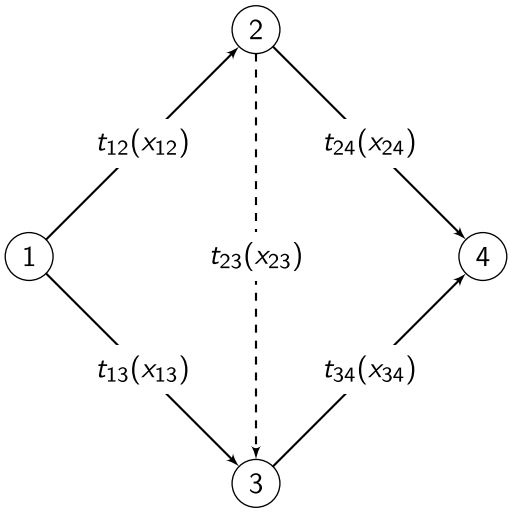


Figure 4.8: Braess network.

v	x_e^v	I_{LM}	I_Z	I_{NQ}	I_{UD}	ΔT
1	4000.00000	1.00000	N/A	1.00000	1.00000	-320000.00000
2	8000.00000	0.47567	0.06250	0.05882	0.00000	20000.00000
3	8000.00000	0.47567	0.06250	0.05882	0.00000	20000.00000
4	4000.00000	1.00000	N/A	1.00000	1.00000	-320000.00000

Table 4.4: Vertex importance measures for Braess network.

In Table 4.3 note that the removal of an edge results in a value of $\Delta T > 0$ and is reflected in the measures I_{LM}, I_Z and I_{NQ} which are all positive. Similarly, when $\Delta T < 0$ each of these measures is negative, i.e. the removal of edge $(u, v) = (2, 3)$ results in a improvement of the total travel time at equilibrium $\Delta T = -60000 < 0$. It is also of note that $I_{UD} = 0$ for all edges, and, thus, the resulting travel times are based on the same demand being routed (i.e. no demand has been removed due to a disconnected commodity).

Inspection of Table 4.4 provides a few key insights. First the removal of vertices 1 and 4 have disconnected the single commodity so all demand is no longer routed, $I_{UD} = 1$. This results in a total travel time of 0 and, thus, there is (if viewed purely by value) an improvement in travel time. Whilst this may seem somewhat a fabricated or fictitious scenario, it is in fact a reasonable phenomenon. One can easily imagine a situation whereby the removal of a large building, e.g. a hospital or school, results in demand being removed from the network and, thus, travel times

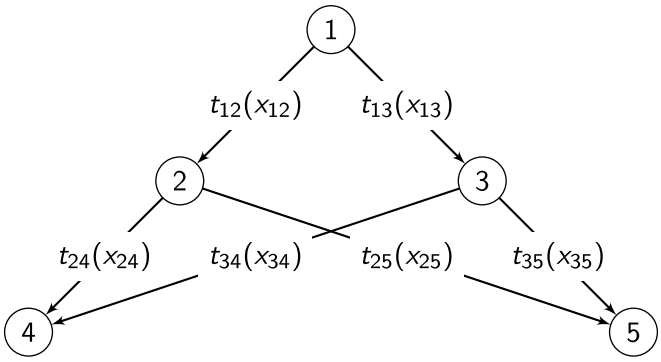


Figure 4.9: Network from Qiang and Nagurney [139].

improve.

The measures (excluding I_Z which is not defined for vertex removal), while reflecting that these vertices are indeed of the utmost importance (i.e. in this case they sever all demand for the network), do not provide an indication that the total travel time has actually improved as they did with the removal of the edge $(2, 3)$ in Table 4.3. Indeed, it can be argued that the measures should reflect the importance for both cases, improvement and worsening of the total travel time.

4.3.3.2 One Origin, Multiple Destinations

The following example (Figure 4.9) is from Qiang and Nagurney [139] and allows confirmation of the results obtained. A full description of the network and edge cost functions can be found in Appendix C.2. It also flags another issue with the removal of vertices that result in a disconnected commodity. The network has two commodities $(1, 4)$ and $(1, 5)$ with respective demands $d_{14} = 100$ and $d_{15} = 20$. Results are summarised in Tables 4.5, 4.6, 4.7 and 4.8.

u	v	x_e	I_{LM}	I_Z	I_{NQ}	I_{UD}	ΔT
1	2	60.00000	0.73030	1.00000	0.50000	0.00000	12400.00105
1	3	60.00000	0.73030	1.00000	0.50000	0.00000	12400.00105
2	4	50.00001	0.49446	0.67742	0.16304	0.00000	8400.00105
2	5	10.00000	0.51189	0.02419	0.04220	0.00000	299.99987
3	4	49.99999	0.49446	0.67742	0.16304	0.00000	8400.00105
3	5	10.00000	0.51189	0.02419	0.04220	0.00000	299.99987

Table 4.5: Edge importance measures for network in Figure 4.9.

u	v	x_e **Rank**	I_{LM} **Rank**	I_Z **Rank**	I_{NQ} **Rank**	ΔT **Rank**
1	2	1	1	1	1	1
1	3	1	1	1	1	1
2	4	2	4	2	2	2
2	5	3	3	3	3	3
3	4	2	4	2	2	2
3	5	3	2	3	3	3

Table 4.6: Edge importance measure rankings for network in Figure 4.9.

v	x_e^v	I_{LM}	I_Z	I_{NQ}	I_{UD}	ΔT
1	120.00000	1.00000	N/A	1.00000	1.00000	-12399.99895
2	120.00000	0.73030	1.00000	0.50000	0.00000	12400.00105
3	120.00000	0.73030	1.00000	0.50000	0.00000	12400.00105
4	100.00000	-0.51663	N/A	0.16304	0.83333	-11999.99971
5	20.00000	0.69667	N/A	0.16304	0.16667	-2400.00014

Table 4.7: Vertex importance measures for network in Figure 4.9.

v	x_e^v **Rank**	I_{LM} **Rank**	I_Z **Rank**	I_{NQ} **Rank**	I_{UD} **Rank**	ΔT **Rank**
1	2	1	N/A	1	1	4
2	1	2	1.00000	2	4	1
3	3	2	1.00000	2	4	1
4	4	4	N/A	4	2	3
5	5	3	N/A	3	3	2

Table 4.8: Vertex importance measure rankings for network in Figure 4.9.

Table 4.5 and Table 4.6 compare the measures given by the removal of individual edges against ΔT. Table 4.6 clearly shows how the edges have been ranked by the measures and illustrates the issue of using the L-M measure with nonatomic selfish routing games, that is the rankings are incorrect when compared with ΔT.

The removal of vertices $1, 4$ and 5 (see Table 4.7) result in a disconnect of commodities. Vertex 1 disconnects both commodities $(1, 4)$ and $(1, 5)$, i.e. $I_{UD} = 1$, whilst vertex 4 disconnects $(1, 4)$ and vertex 5 disconnects $(1, 5)$. Comparing the removal of vertices 4 and 5, clearly the removal of vertex 4 removes more demand from the network, $I_{UD} = 0.8333 > 0.1667$, and the total travel time is also improved to a greater extent, $\Delta T = -11999.9997 < -2400.0001$. Again, there are issues with the measures, I_{LM} has opposite signs and I_{NQ} provides an equal importance value of $I_{NQ} = 0.16304$. None of the measures fully capture all the information necessary to distinguish that: demand has been removed; the travel time has improved; the difference in how much the travel time has improved. Put simply, it is impossible to determine the cause of the effect by the values given by the measures.

4.3.3.3 Multiple Origins, Multiple Destinations

To examine the measures on a larger network with multiple commodities, the Sioux Falls network (Appendix C.3) is used and the effect of removing each edge and each vertex is examined.

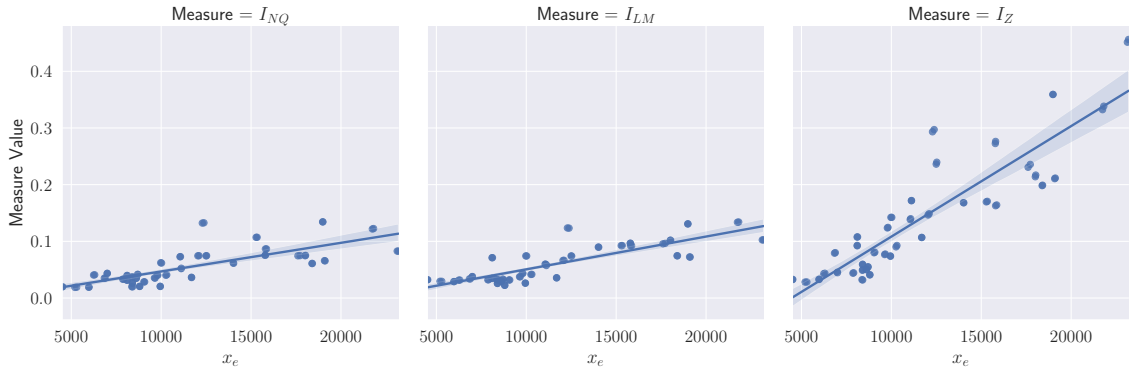
Figures 4.10, 4.11, 4.12, 4.13 present the results for removal of edges and Figures 4.14, 4.15, 4.16, 4.17 the results for the removal of vertices. Figures 4.10, 4.11, 4.16, 4.17 display the value given by the measures, whereas Figures 4.12, 4.13, 4.14, 4.15 display the ranking of the component. Note that I_Z is undefined for the removal

of a vertex with an associated demand. In the case of the Sioux Falls network this applies to all vertices, hence no results can be given for I_Z for the removal of vertices.

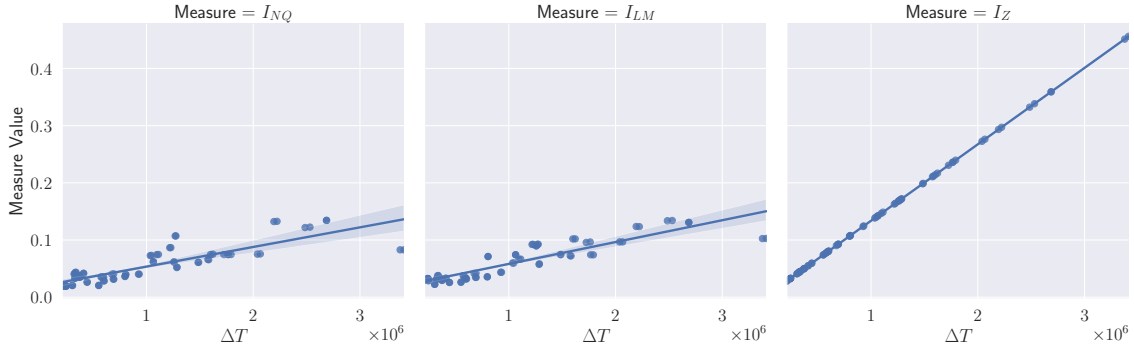
Inspection of Figures 4.11 and 4.13 show that all three measures I_{NQ}, I_{LM} and I_Z exhibit a positive correlation with the flow on an edge x_e with correlation values of 0.76, 0.84 and 0.89, and correlation rank values of 0.9, 0.82 and 0.82, respectively from the removal of an edge e. The measures also exhibit a strong correlation with the change in total travel time ΔT, with correlation values of 0.85, 0.9 and 1, and correlation rank values of 0.88, 0.91 and 1, respectively. In particular, I_Z is perfectly correlated with the change in total travel time ΔT - this is as expected as for equilibrium I_Z is just a form of normalised ΔT that differs by a constant factor (see Theorem 4.4.1). It is also of note that the flow x_e and the change in total travel time ΔT have a strong correlation. and that the amended version of I_{LM} also has a higher correlation value than the I_{NQ} measure. Figure 4.10c and 4.12c also show that for I_Z the spread of measure is wider and less bunched than I_{LM} and I_{NQ}. This may account from some of the errors in correctly correlating the rank.

Figures 4.15 and 4.17 display the correlation results from removing a vertex v. I_{NQ} and I_{LM} have positive correlation values of 0.89 and 0.69, respectively for the flow incident to the vertex v - x_e^v. With regards to the total travel time ΔT the values are 0.72 and 0.94, respectively. This suggests that while I_{NQ} is a good predictor of the flow incident to a vertex, it is less successful as a predictor of the change in total travel time due to the removal of a vertex when compared with I_{LM}.

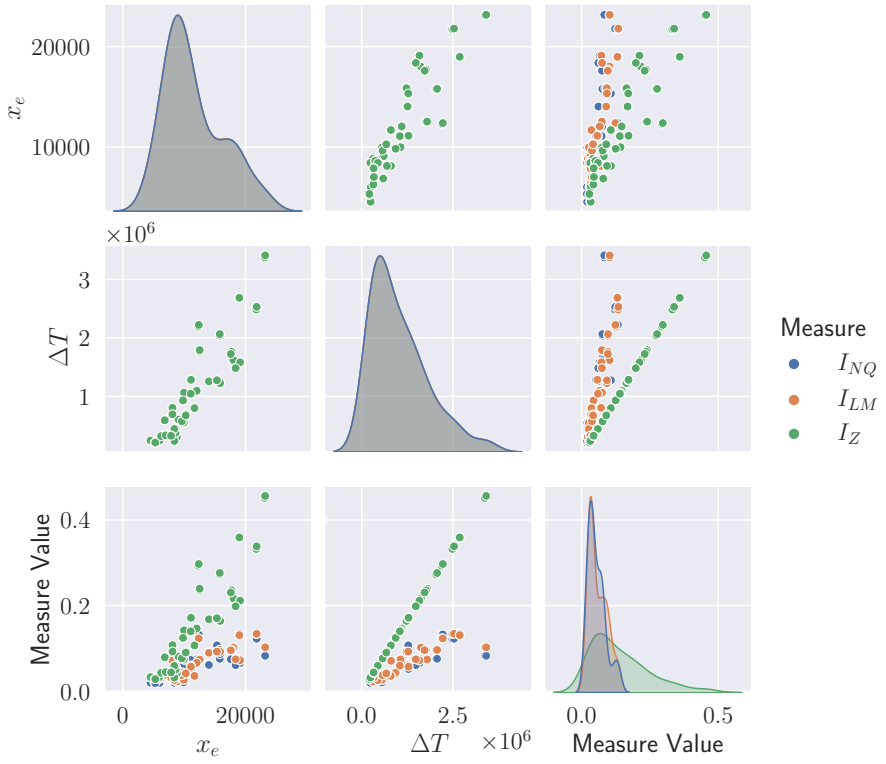
Therefore, the I_{NQ} measure does not appear to provide a strong enough case to distinguish itself from the standard I_{LM} measure.



(a) Linear regression for x_e vs I_{NQ}, I_{LM} and I_z measures.



(b) Linear regression plot for ΔT vs I_{NQ}, I_{LM} and I_z measures.



(c) Pair plot for measures.

Figure 4.10: Comparison of I_{NQ}, I_{LM} and I_z for the removal of an **edge** e.

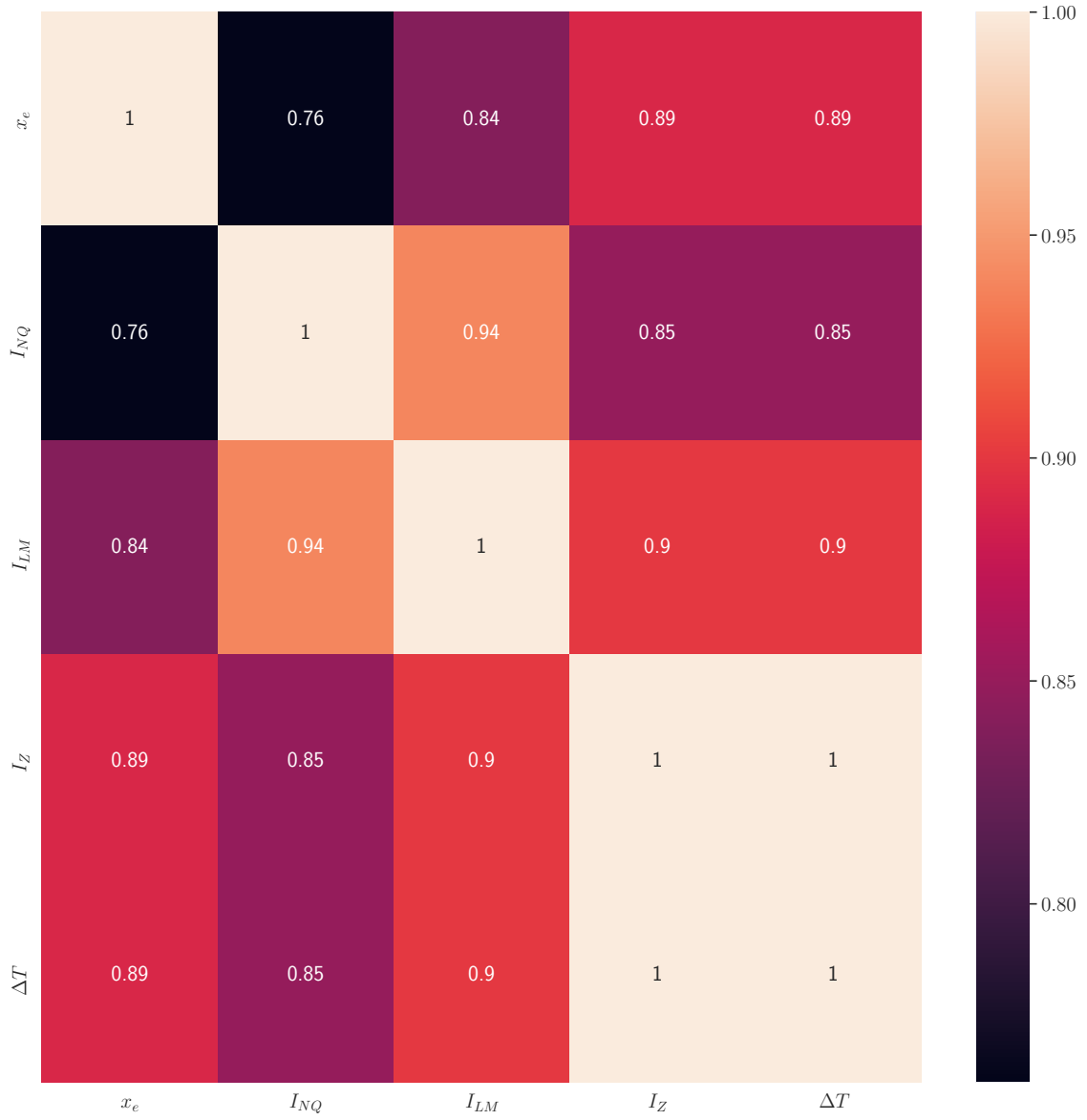
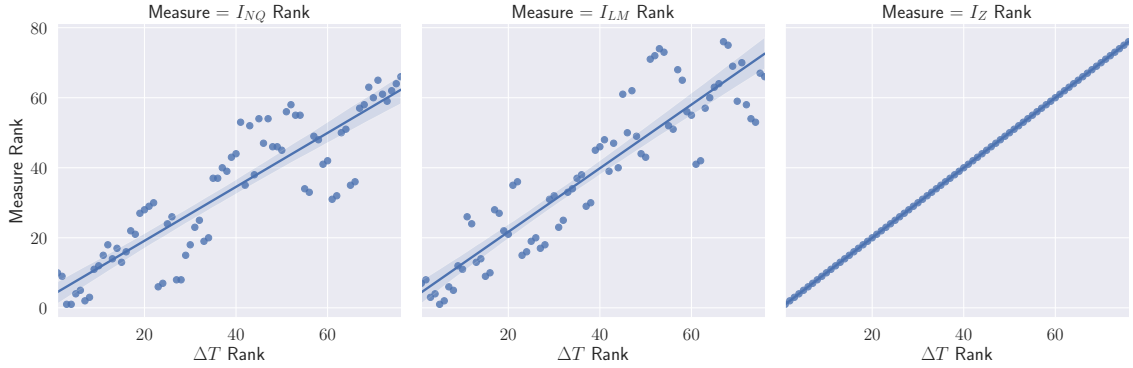
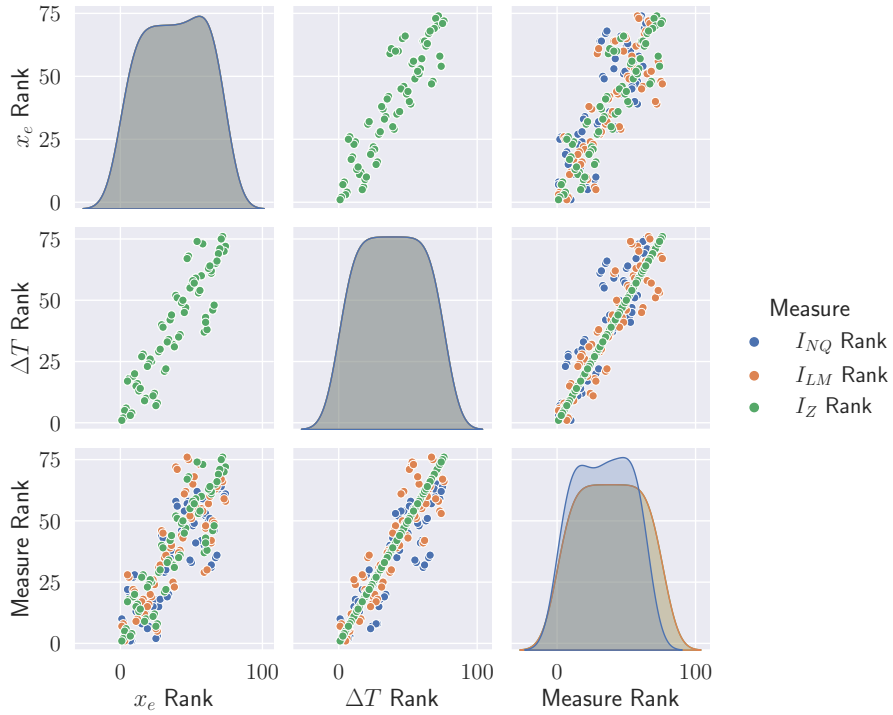


Figure 4.11: Pearson correlation matrix for removal of an **edge** e.

(a) Linear regression for x_e vs I_{NQ}, I_{LM} and I_z measures.



(b) Linear regression plot for ΔT vs I_{NQ}, I_{LM} and I_z measures.



(c) Pair plot for edge measures.

Figure 4.12: Comparison of I_{NQ}, I_{LM} and I_z for the removal of an **edge** e (Ranking).

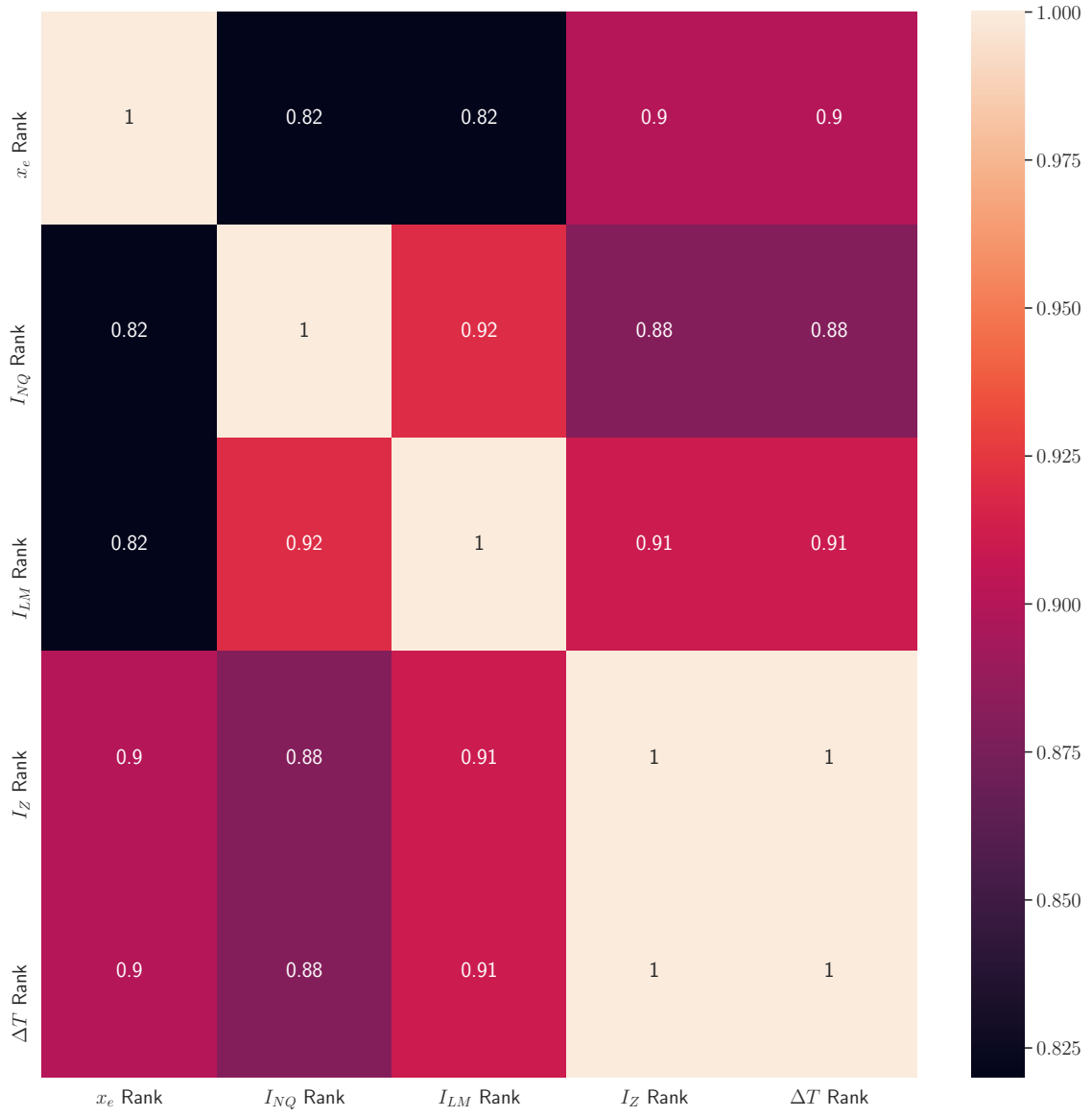
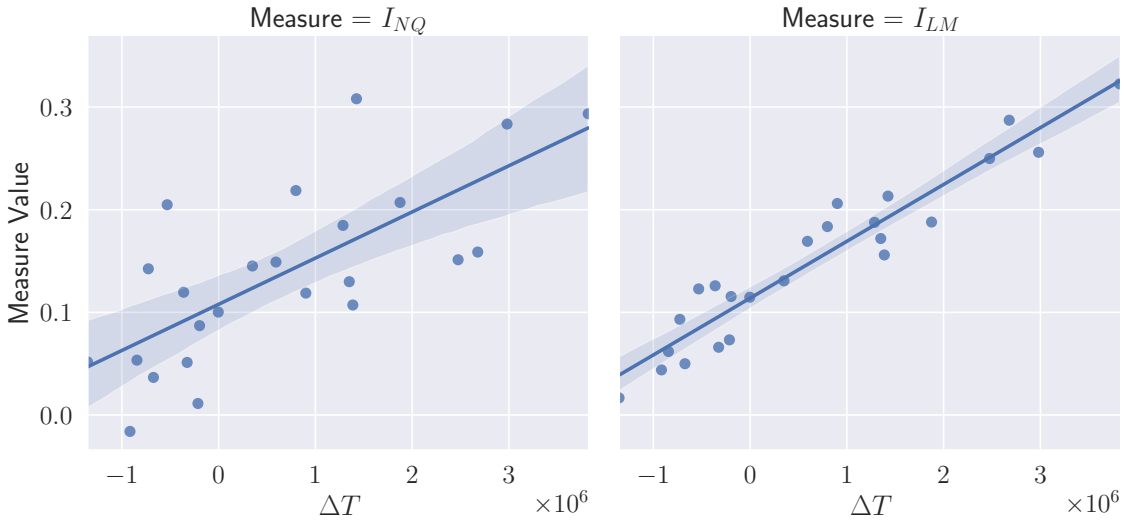
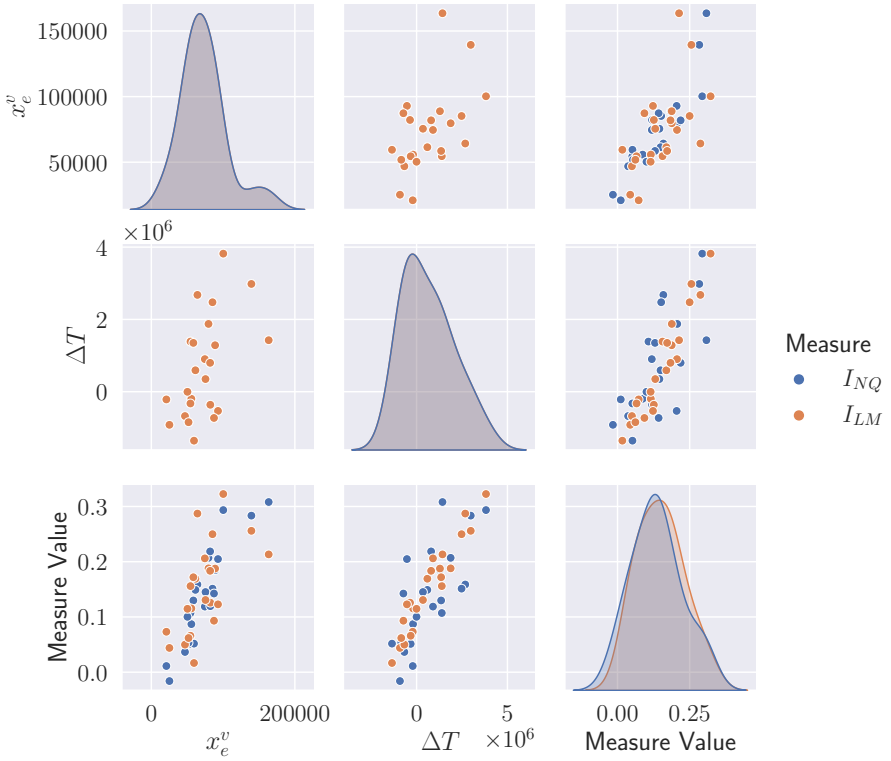


Figure 4.13: Pearson correlation matrix for removal of an **edge** e (Ranking).



(a) Linear regression for ΔT vs I_{NQ}, I_{LM} and I_z vertex measures.



(b) Pair plot for vertex measures.

Figure 4.14: Comparison of I_{NQ}, I_{LM} and I_z measures for the removal of a **vertex** v.

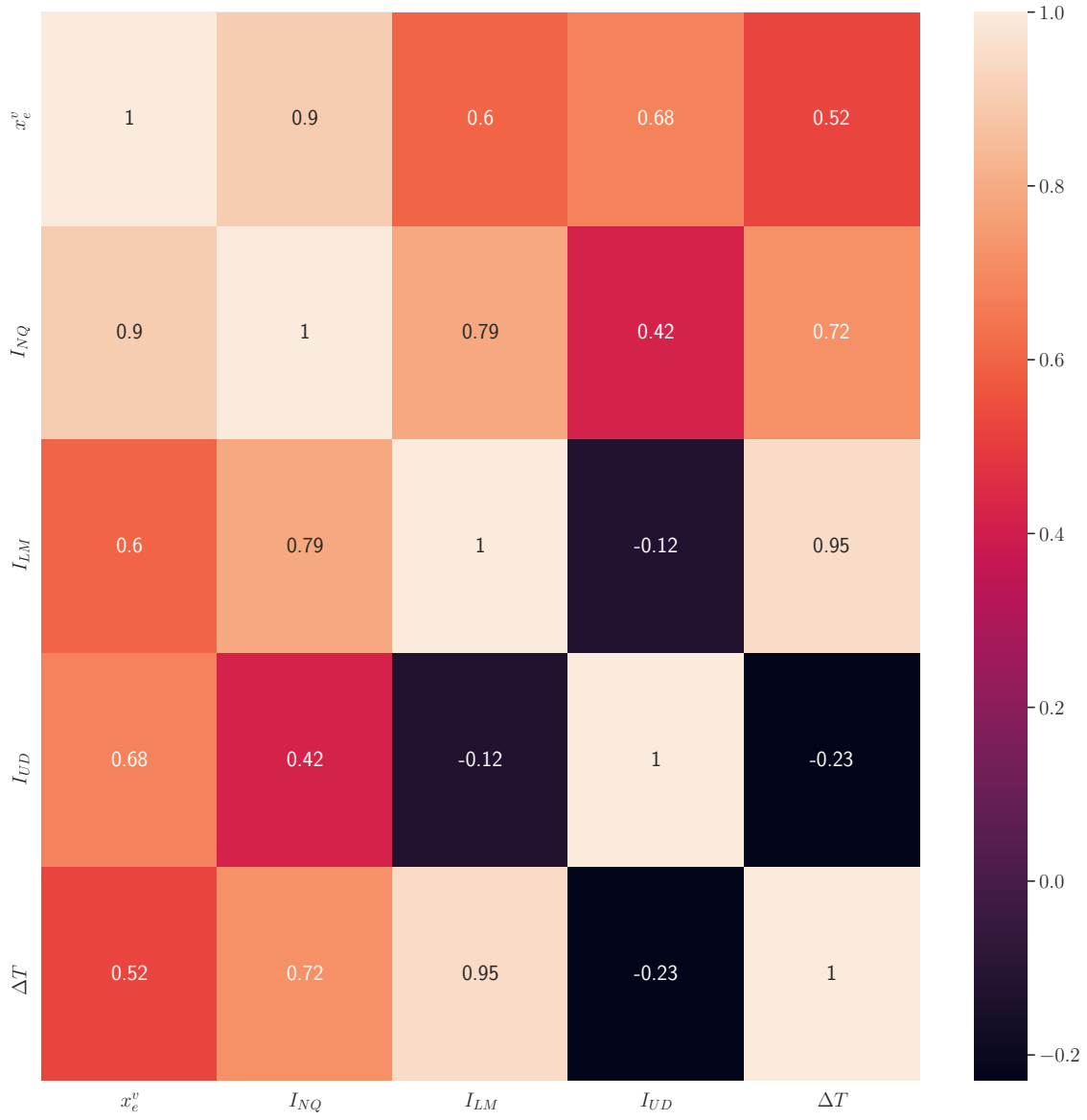
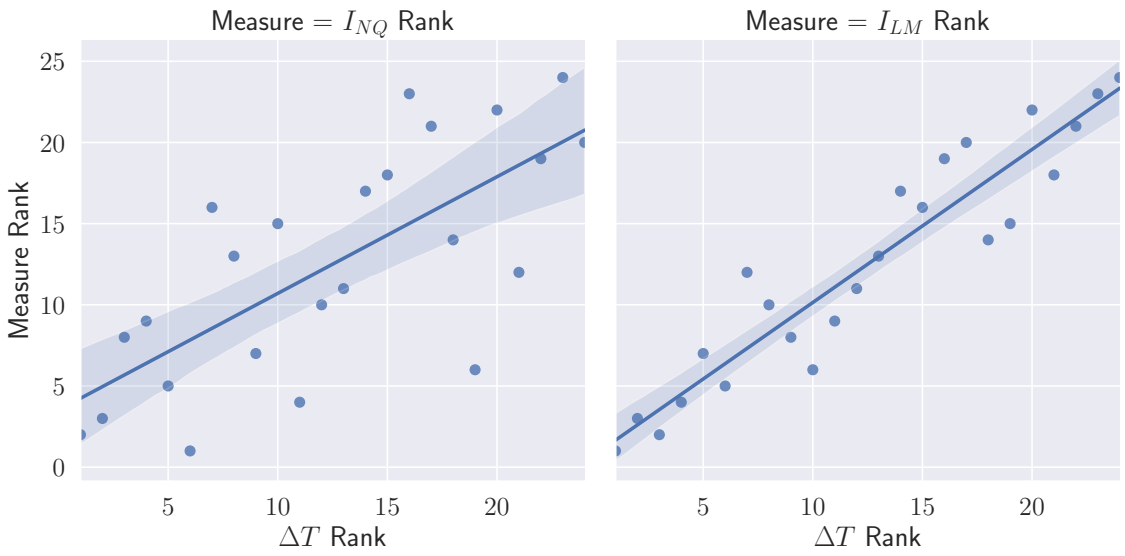
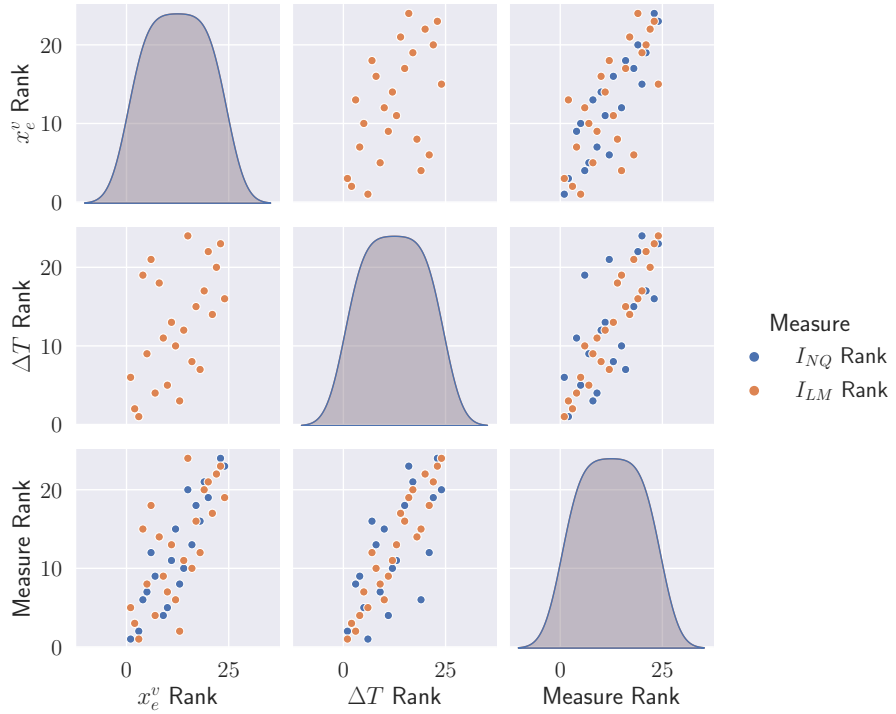


Figure 4.15: Pearson correlation matrix for removal of a **vertex** v.



(a) Linear regression for ΔT vs I_{NQ}, I_{LM} and I_z vertex measures.



(b) Pair plot for vertex measures.

Figure 4.16: Comparison of I_{NQ}, I_{LM} and I_z for the removal of a **vertex** v (Ranking).

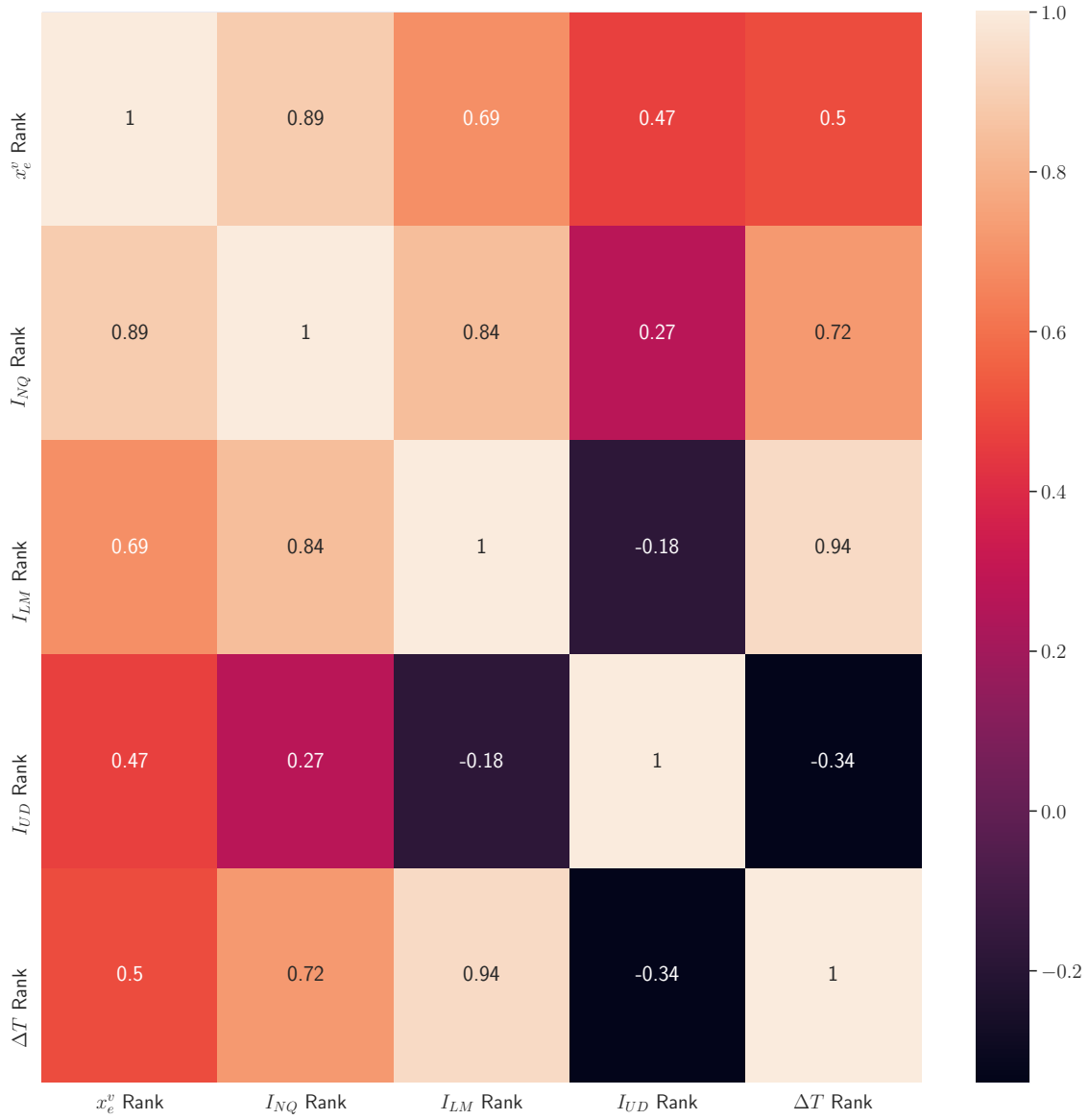


Figure 4.17: Pearson correlation matrix for removal of a **vertex** v (Ranking).

4.3.4 Summary of Issues Pertaining to Demand-based Measures

Section 4.3.3 identified issues with the I_{LM}, I_Z and I_{NQ} measures. The following list summarises the issues:

1. If there is a disconnect in the network, i.e. a commodity can no longer route the specified demand, then I_Z is undefined.

2. $I_Z(S) = \frac{\Delta E}{E} = \frac{E_Z(G',d) - E_Z(G,d)}{E_Z(G,d)}$ is not bounded above.

   i.e. $\lim_{E_Z(G',d)\to\infty} I_Z(S) = \frac{E_Z(G',d) - E_Z(G,d)}{E_Z(G,d)} = \frac{E_Z(G',d)}{E_Z(G,d)} - 1 = \infty$

3. $I_{NQ}(S)$ and $I_{LM}(S)$ are not bounded below.

   i.e. $\lim_{E_M(G',d)\to\infty} I(S) = \frac{E_M(G,d)-E_M(G',d)}{E_M(G,d)} = 1 - \frac{E_M(G',d)}{E_M(G,d)} = -\infty$

4. Other than I_{UD} measures do not identify that demand has been removed.

5. Do not correctly reflect an improvement or worsening of total travel time in cases where there is a disconnect in the network.

6. Do not correctly reflect the magnitude of the improvement of total travel time due to a change in demand, e.g. the example given by the removal of vertices 4 and 5 in Section 4.3.3.2.

7. Denominator is not relevant when used by the normalised importance (Definition 4.3.1).

Comparing these with the desirable properties listed in Section 4.3.1, there are a number of deficiencies that need addressing.

4.4 An Improved Measure

Section 4.3.4 motivates the need for the development of a normalised measure which does not suffer with the issues presented in Section 4.3.4.

The new measure will be the reciprocal of the total travel time and, with some additional definitions, will resolve the aforementioned issues. The reciprocal is chosen to allow the definition of the normalised importance to be defined as it is defined for E_{NQ} and E_{LM}, that is, there is not a requirement for a switch in sign.

The measure will also not be weighted by the number of commodities, possible connections or demand, as these are effectively a constant factor in the definitions of other demand-based measures. That is, it does not contribute to the normalised importance as noted in Section 4.3.2.1.

Definition 4.4.1 (Reciprocal of the Total Travel Time)**.** *The network efficiency measure $E_R(G,d)$ for a given network $G = (V,E)$, is defined as:*

$$E_R(G,d) = \frac{1}{\sum_{k\in K} \lambda_k d_k},$$

where for a given commodity $k = (u, v)$, $\lambda_k = d(u, v)$, i.e. the shortest path for commodity k.

In the case that a commodity $k \in K$ becomes disconnected, all demand of the commodity k, d_k is assigned to a path of no length, $\lambda_k = 0$.

In the case that all commodities are disconnected in G' then $E_R(G', d) = \frac{1}{\epsilon}$.

The rationale for $E_R(G', d) = \frac{1}{\epsilon}$ is that in the case that all commodities are disconnected, $\sum_{k \in K} \lambda_k d_k = 0$ and the measure would be undefined. This also has the nice property that

$$I = \frac{E_R(G, d) - E_R(G', d)}{E_R(G, d)} = \lim_{\epsilon \to 0} \frac{E_R(G, d) - \frac{1}{\epsilon}}{E_R(G, d)} = -\infty.$$

The total travel time, $\sum_{k \in K} \lambda_k d_k = 0$, is viewed, at least from a purely sign-based perspective, as an improvement. That is, the value of $-\infty$ corresponds to the maximum improvement possible in the total travel time. Ideally this would be bounded below by -1 to allow a more reasonable comparison across networks.

To address this issue an updated definition of the normalised importance is presented.

4.4.1 Revised Normalised Importance

Definition 4.4.2 (Revised Normalised Importance of a Set of Network Vertices and Edges)**.** *Given a nonatomic selfish routing game (G, d, t) and a subgraph (G', d, t), let S represent the set of removed vertices and edges from $G = (V, E)$ and $G' = (V', E')$,*

$$S = \{V \setminus V' \ \cup E \setminus E'\}.$$

The revised importance of the set S, $I(S)$ is measured by the relative network efficiency drop after S is removed from the network G:

$$\hat{I}(S) = \frac{\Delta E}{E} = \frac{E(G, d) - E(G', d)}{\max(E(G, d), E(G', d))},$$

where $E(G, d)$ is the network efficiency measure for the nonatomic selfish routing game (G, d, t).

For the reciprocal measure,

$$\hat{I} = \frac{E_R(G, d) - E_R(G', d)}{\max(E_R(G, d), E_R(G', d))} = \lim_{\epsilon \to 0} \frac{E_R(G, d) - \frac{1}{\epsilon}}{\frac{1}{\epsilon}} = -1.$$

4.4.2 A Few Theorems

The following section outlines some theorems about the chosen existing measures outlined in Section 4.3.2 and the newly defined measure given in the previous section.

Theorem 4.4.1. *At equilibrium $I_Z(S)$ differs from ΔT by a constant factor of $-\frac{1}{\sum_{k\in K}\lambda_k d_k}$.*

Proof.

$$\hat{I}_Z(S) = \frac{\frac{\sum_{k\in K}\lambda'_k d_k}{\sum_{k\in K} d_k} - \frac{\sum_{k\in K}\lambda_k d_k}{\sum_{k\in K} d_k}}{\frac{\sum_{k\in K}\lambda_k d_k}{\sum_{k\in K} d_k}} = \frac{\sum_{k\in K}\lambda'_k d_k - \sum_{k\in K}\lambda_k d_k}{\sum_{k\in K}\lambda_k d_k} = -\frac{1}{\sum_{k\in K}\lambda_k d_k}\Delta T$$

□

Theorem 4.4.2. *The revised normalised importance $\hat{I}_Z(S)$ of $E_Z(G,d)$ is equal and opposite in sign to the normalised importance $\hat{I}_R(S)$ of $E_R(G,d)$ when defined, i.e. all commodities are connected when the set S is removed from G.*

Proof. Let $\hat{I}_Z(S)$ and $\hat{I}_R(S)$ denote the importance of $E_Z(G,d)$ and $E_R(G,d)$ respectively.

$$\hat{I}_Z(S) = \frac{\frac{\sum_{k\in K}\lambda_k d_k}{\sum_{k\in K} d_k} - \frac{\sum_{k\in K}\lambda'_k d_k}{\sum_{k\in K} d_k}}{\max\left(\frac{\sum_{k\in K}\lambda_k d_k}{\sum_{k\in K} d_k}, \frac{\sum_{k\in K}\lambda'_k d_k}{\sum_{k\in K} d_k}\right)},$$

and,

$$\hat{I}_R(S) = \frac{\frac{1}{\sum_{k\in K}\lambda_k d_k} - \frac{1}{\sum_{k\in K}\lambda'_k d_k}}{\max\left(\frac{1}{\sum_{k\in K}\lambda_k d_k}, \frac{1}{\sum_{k\in K}\lambda'_k d_k}\right)}.$$

Considering the two cases:

Case 1:

If

$$\frac{\sum_{k\in K}\lambda_k d_k}{\sum_{k\in K} d_k} > \frac{\sum_{k\in K}\lambda'_k d_k}{\sum_{k\in K} d_k},$$

then,

$$\hat{I}_Z(S) = \frac{\frac{\sum_{k\in K}\lambda_k d_k}{\sum_{k\in K} d_k} - \frac{\sum_{k\in K}\lambda'_k d_k}{\sum_{k\in K} d_k}}{\frac{\sum_{k\in K}\lambda_k d_k}{\sum_{k\in K} d_k}} = \frac{\sum_{k\in K}\lambda_k d_k - \sum_{k\in K}\lambda'_k d_k}{\sum_{k\in K}\lambda_k d_k}.$$

Noting that,

$$\frac{a-b}{a} = \frac{\frac{1}{b} - \frac{1}{a}}{\frac{1}{b}}$$

$$\hat{I}_Z(S) = \frac{\frac{1}{\sum_{k\in K}\lambda'_k d_k} - \frac{1}{\sum_{k\in K}\lambda_k d_k}}{\frac{1}{\sum_{k\in K}\lambda'_k d_k}} = -\left(\frac{\frac{1}{\sum_{k\in K}\lambda_k d_k} - \frac{1}{\sum_{k\in K}\lambda'_k d_k}}{\frac{1}{\sum_{k\in K}\lambda'_k d_k}}\right) = -\hat{I}_R(S),$$

where,

$$\frac{1}{\sum_{k\in K}\lambda'_k d_k} > \frac{1}{\sum_{k\in K}\lambda_k d_k} \implies \frac{\sum_{k\in K}\lambda_k d_k}{\sum_{k\in K} d_k} > \frac{\sum_{k\in K}\lambda'_k d_k}{\sum_{k\in K} d_k}.$$

Case 2

If

$$\frac{\sum_{k\in K}\lambda_k d_k}{\sum_{k\in K} d_k} < \frac{\sum_{k\in K}\lambda'_k d_k}{\sum_{k\in K} d_k},$$

then,

$$-\hat{I}_Z(S) = \hat{I}_R(S)$$

follows by substituting into the steps for case 1. □

Theorem 4.4.3. *(Nagurney 2007) If positive demands exist for all vertices in G and the demand for each of these pairs is equal to 1 and $\lambda_{uv} = d(u,v)$ then $E_{NQ}(G,d)$ and $E_{LM}(G)$ are equivalent.*

Proof. Let n be the number of vertices in G, then the number of commodities is given by $n(n-1)$. Furthermore, by assumption that $d_k = 1$, $\forall k \in K$ then $E_{LM}(G)$ is given as follows:

$$E_{LM}(G) = \frac{1}{n(n-1)} \sum_{u,v\in V:u\neq v} \frac{1}{d(u,v)} = \frac{\sum_{u,v\in V:u\neq v}\frac{1}{d(u,v)}}{n_K} = \frac{\sum_{k\in K}\frac{d_k}{\lambda_k}}{n_K} = E_{NQ}(G)$$

□

Theorem 4.4.4. *If the network G only has one commodity c_1, then the normalised importance of $\hat{I}_{NQ}(S)$ and $\hat{I}_R(S)$ are the either the same or opposite in sign (due to the disconnect of the demand).*

Proof. As the network has a single commodity c_1, there is a single demand d_1.

Assume that there exists a shortest path for c_1 in both G and G', then:

Case $\lambda_1 > \lambda'_1$:

$$\hat{I}_R(S) = \frac{\frac{1}{\lambda_1 d_1} - \frac{1}{\lambda'_1 d_1}}{\max(\frac{1}{\lambda_1 d_1}, \frac{1}{\lambda'_1 d_1})} = \frac{\frac{1}{\lambda_1 d_1} - \frac{1}{\lambda'_1 d_1}}{\frac{1}{\lambda_1 d_1}} = \frac{\frac{1}{\lambda_1} - \frac{1}{\lambda'_1}}{\frac{1}{\lambda_1}} = \frac{\frac{d_1}{\lambda_1} - \frac{d_1}{\lambda'_1}}{\frac{d_1}{\lambda_1}} = \hat{I}_{NQ}(S)$$

Case $\lambda_1 < \lambda'_1$:

follows as above.

Case $\lambda_1 = \lambda_1'$:

$$\hat{I}_R(S) = 0 = \hat{I}_{NQ}(S)$$

Assume that there no longer exists a shortest path for c_1 in G', then by definition $E_R(G', d) = \frac{1}{\epsilon}$ and $\lim_{\lambda_1' \to \infty} E_{NQ}(G', d) = 0$. Thus,

$$\hat{I}_R(S) = \frac{E_R(G,d) - E_R(G',d)}{\max(E_R(G,d), E_R(G',d))} = \lim_{\epsilon \to 0} \frac{E_R(G,d) - \frac{1}{\epsilon}}{\frac{1}{\epsilon}} = -1$$

$$\hat{I}_{NQ}(S) = \frac{E_{NQ}(G,d) - E_{NQ}(G',d)}{\max(E_{NQ}(G,d), E_{NQ}(G',d))} = \lim_{\lambda_1' \to \infty} \frac{E_{NQ}(G,d) - 0}{E_{NQ}(G,d)} = 1$$

□

Theorem 4.4.5. *$\hat{I}_R(S) < 0$ if and only if the total travel time has been improved.*

Proof. This follows trivially from the definition of $\hat{I}_R(S)$.

$\Longleftarrow$

$\hat{I}_R(S) < 0 \implies E_R(G', d) > E_R(G, d)$ and therefore $\frac{1}{\sum_{k \in K} \lambda_k' d_k} > \frac{1}{\sum_{k \in K} \lambda_k d_k} \implies \sum_{k \in K} \lambda_k' d_k < \sum_{k \in K} \lambda_k d_k$. i.e. the total travel time for G' is less than that of G.

$\Longrightarrow$

Assume that the total travel time for G' is less than that of G, then:

$\sum_{k \in K} \lambda_k' d_k < \sum_{k \in K} \lambda_k d_k \implies \frac{1}{\sum_{k \in K} \lambda_k' d_k} > \frac{1}{\sum_{k \in K} \lambda_k d_k} \implies E_R(G', d) > E_R(G, d) \implies \hat{I}_R(S) < 0$ □

The consequence of theorem 4.4.5 is that something has caused the total travel time to improve despite the removal of network components. There are two possibilities, either a commodity has been disconnected and thus the corresponding demand has been assigned to a fictitious path, i.e. it has not contributed to the total travel time, or the removal of S from G has resulted in a Braess Paradox.

A stronger condition for theorem 4.4.5 would have been to claim that if a network commodity is removed, then $\hat{I}_R(S) < 0$; however, as the Section 4.5 shows, this need not necessarily be the case.

4.4.3 Comparison and Analysis Against Existing Measures

Using the same networks given in Section 4.3.3, the $\hat{I}_R$ measure is compared against the other demand-based measures.

4.4.3.1 Braess Network

Results for the Braess network given in Figure 4.8 are summarised in Tables 4.9 and 4.10. Whilst the results given in Table 4.9 (removal of an edge) are similar to the other measures, the results in Table 4.10, (removal of a vertex) illustrate the first difference. The removal of vertices 1 and 4 result in the same sign for $\hat{I}_R$ as ΔT, this is important as these are the vertices associated with the single commodity and thus disconnect the network (reflected by $I_{UD} = 1$). Thus, the $\hat{I}_R$ has indicated not only that these vertices of the utmost importance, but also that they disconnect the network and demand is removed.

u	v	x_e	I_{LM}	I_Z	I_{NQ}	$\hat{I}_R$	I_{UD}	ΔT
1	2	4000.00000	0.25345	0.06250	0.05882	0.05882	0.00000	20000.00000
1	3	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
2	3	4000.00000	-0.42070	-0.18750	-0.23077	-0.18750	0.00000	-60000.03815
2	4	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
3	4	4000.00000	0.25345	0.06250	0.05882	0.05882	0.00000	20000.00000

Table 4.9: Edge importance measures for Braess network given in Figure 4.8.

v	x_e^v	I_{LM}	I_Z	I_{NQ}	$\hat{I}_R$	I_{UD}	ΔT
1	4000.00000	1.00000	N/A	1.00000	-1.00000	1.00000	-320000.00000
2	8000.00000	0.47567	0.06250	0.05882	0.05882	0.00000	20000.00000
3	8000.00000	0.47567	0.06250	0.05882	0.05882	0.00000	20000.00000
4	4000.00000	1.00000	N/A	1.00000	-1.00000	1.00000	-320000.00000

Table 4.10: Vertex importance measures for Braess network given in Figure 4.8.

4.4.3.2 One Origin, Multiple Destinations

Results for the network given in Figure 4.9 are summarised in Tables 4.11, 4.12, 4.13 and 4.14. From inspection of the tables, $\hat{I}_R$ correctly ranks the change in total travel time ΔT and, on the removal of vertices which may disconnect commodities, has a sign that matches ΔT. It also addresses the issue of not correctly reflecting the magnitude of the improvement of total travel time due to a change in demand,

namely the values given in Table 4.13 reflect how much demand has been removed, i.e they correlate with I_{UD}.

u	v	x_e	I_{LM}	I_Z	I_{NQ}	$\hat{I}_R$	I_{UD}	ΔT
1	2	60.00000	0.73030	1.00000	0.50000	0.50000	0.00000	12400.00105
1	3	60.00000	0.73030	1.00000	0.50000	0.50000	0.00000	12400.00105
2	4	50.00001	0.49446	0.67742	0.16304	0.40385	0.00000	8400.00105
2	5	10.00000	0.51189	0.02419	0.04220	0.02362	0.00000	299.99987
3	4	49.99999	0.49446	0.67742	0.16304	0.40385	0.00000	8400.00105
3	5	10.00000	0.51189	0.02419	0.04220	0.02362	0.00000	299.99987

Table 4.11: Edge importance measures for network given in Figure 4.9.

u	v	x_e **Rank**	I_{LM} **Rank**	I_Z **Rank**	I_{NQ} **Rank**	$\hat{I}_R$ **Rank**	ΔT **Rank**
1	2	1	1	1	1	1	1
1	3	1	1	1	1	1	1
2	4	2	4	2	2	2	2
2	5	3	3	3	3	3	3
3	4	2	4	2	2	2	2
3	5	3	2	3	3	3	3

Table 4.12: Edge importance measure rankings for network given in Figure 4.9.

v	x_e^v	I_{LM}	I_Z	I_{NQ}	$\hat{I}_R$	I_{UD}	ΔT
1	120.00000	1.00000	N/A	1.00000	-1.00000	1.00000	-12399.99895
2	120.00000	0.73030	1.00000	0.50000	0.50000	0.00000	12400.00105
3	120.00000	0.73030	1.00000	0.50000	0.50000	0.00000	12400.00105
4	100.00000	-0.51663	N/A	0.16304	-0.96774	0.83333	-11999.99971
5	20.00000	0.69667	N/A	0.16304	-0.19355	0.16667	-2400.00014

Table 4.13: Vertex importance measures for network given in Figure 4.9.

v	x_e^v **Rank**	I_{LM} **Rank**	I_Z **Rank**	I_{NQ} **Rank**	$\hat{I}_R$ **Rank**	I_{UD} **Rank**	ΔT **Rank**
1	2	1	N/A	1	4	1	4
2	1	2	1.00000	2	1	4	1
3	3	2	1.00000	2	1	4	1
4	4	4	N/A	4	3	2	3
5	5	3	N/A	3	2	3	2

Table 4.14: Vertex importance measures for network given in Figure 4.9.

4.4.3.3 Multiple Origins, Multiple Destinations

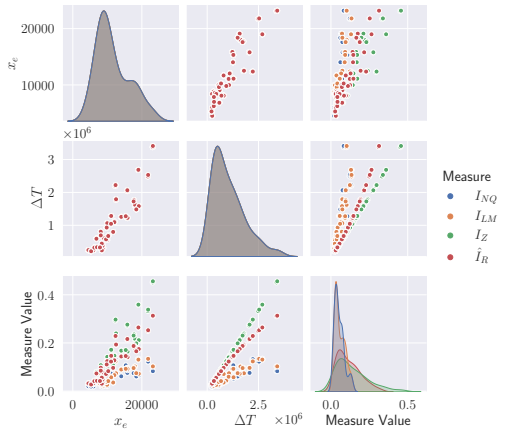
Results for the Sioux Falls network (Appendix C.3) are summarised in Figures 4.18, 4.19 and 4.20. Figures 4.18 and 4.19 show correlation plots for the removal of edges and vertices respectively. Figure 4.20a and 4.20b display the importance of each of the edges and vertices, respectively. Finally, the edge importance values are displayed on the network itself in Figure 4.21 with the bidirectional edges (10,15) and (4,11) being the most important and least important edges, respectively, i.e. removal of these edges have the largest and smallest impact on the total travel time ΔT.

For the removal of an edge, Figure 4.18 shows a perfect correlation between $\hat{I}_R$ and ΔT for the value measure and the ranking given by the measure. $\hat{I}_R$ also exhibits a strong correlation with the flow on an edge, x_e.

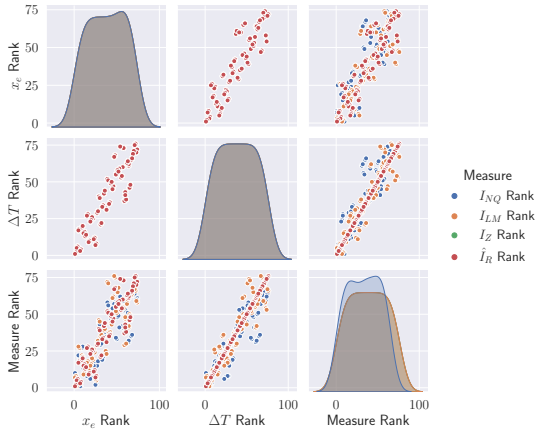
Similarly, for the removal of a vertex, Figure 4.19 shows a perfect correlation between $\hat{I}_R$ and ΔT for the value measure and the ranking given by the measure. $\hat{I}_R$ is exhibits a strong correlation with x_e^v.

There are two key observations. First, $\hat{I_R}$ has higher correlation values than both the I_{LM} and I_{NQ}. This raises the question as to what benefit the I_{NQ} measure provides over the change in total travel time ΔT or a measure which provides a similar correlation such as I_Z or $\hat{I}_R$. Second, the measure is defined for the removal of a vertex and, therefore, can provide a means of measurement whereas I_Z cannot.

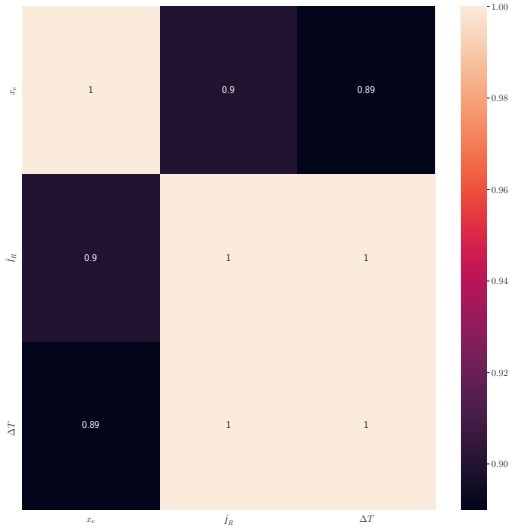
Figure 4.20b displays both positive and negative values of $\hat{I}_R$. There are 13 vertices (positive value of $\hat{I}_R$) whose removal worsen the total travel time and 11 vertices (negative value of $\hat{I}_R$) whose removal improves the total travel time.



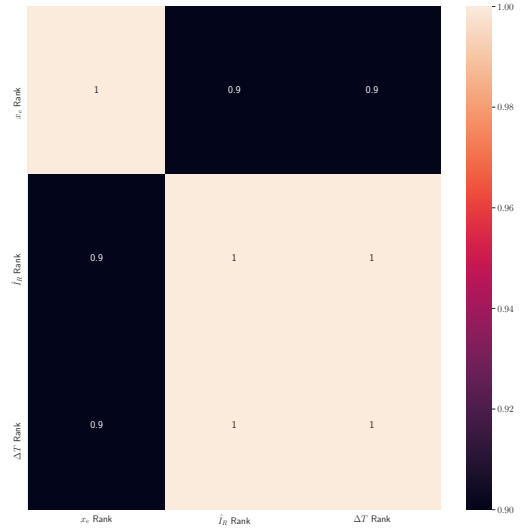
(a) Pair plot (value).



(b) Pair plot (rank).

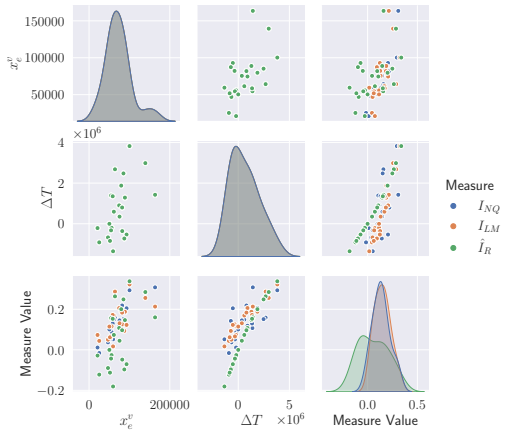


(c) Pearson correlation matrix (value).

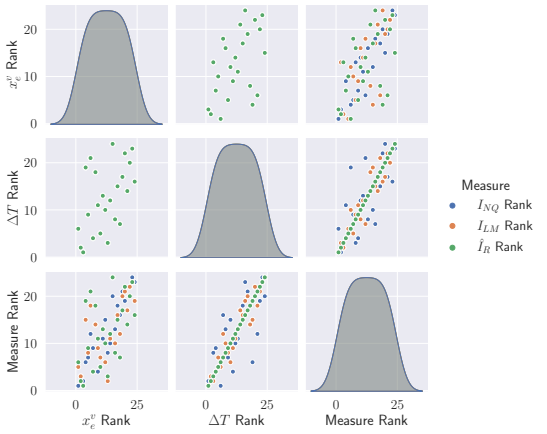


(d) Pearson correlation matrix (rank).

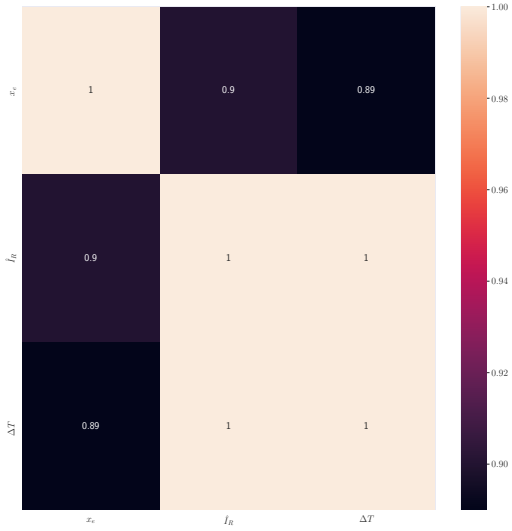
Figure 4.18: Comparison of $\hat{I}_R$ measure for the removal of an edge e.



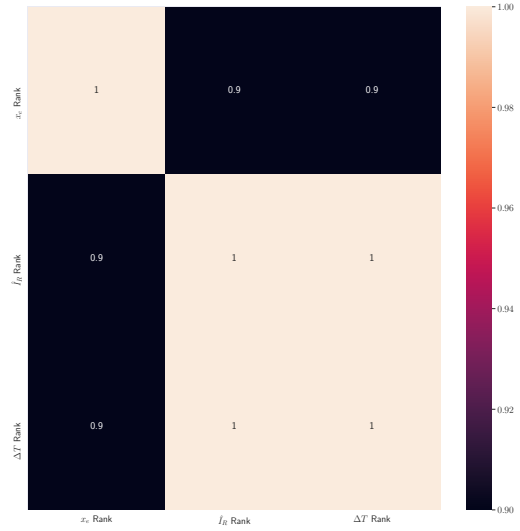
(a) Pair plot (value).



(b) Pair plot (rank).

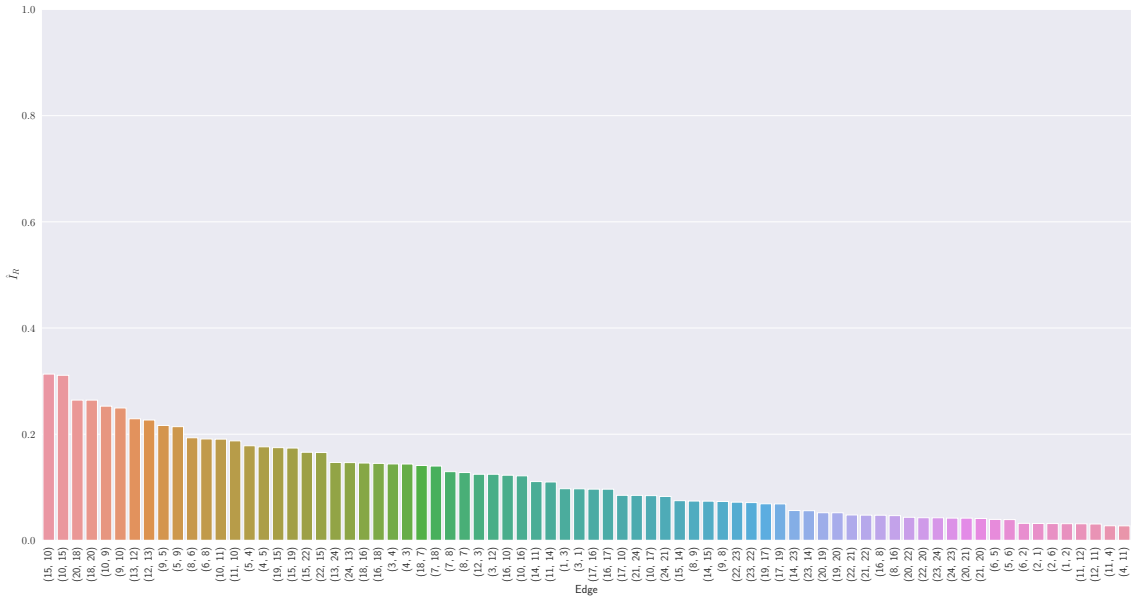


(c) Pearson correlation matrix (value).

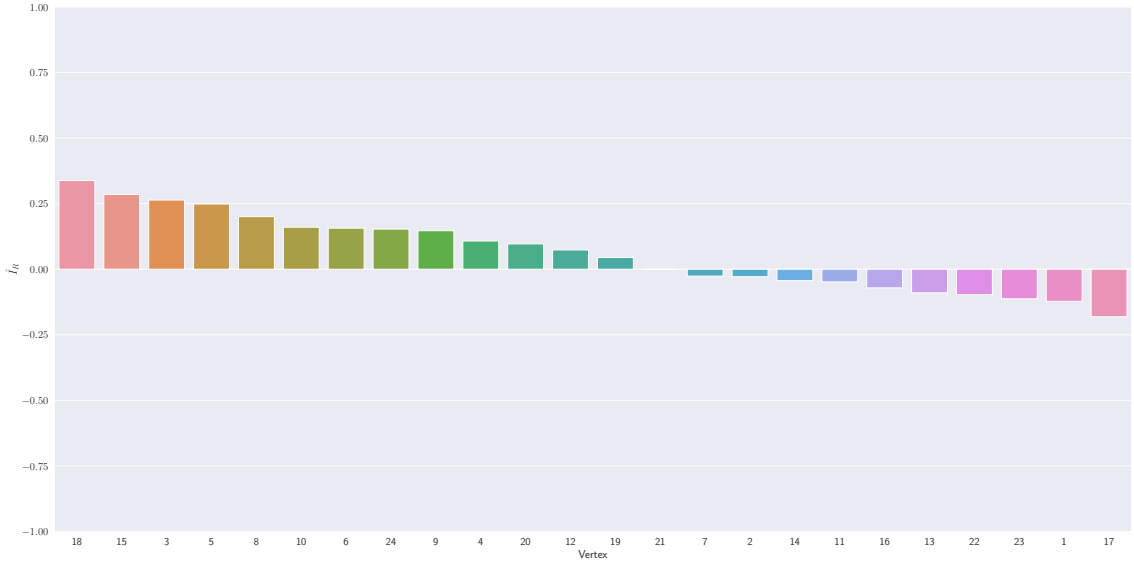


(d) Pearson correlation matrix (rank).

Figure 4.19: Comparison of $\hat{I}_R$ measure for the removal of a vertex v.



(a) Edges ordered by $\hat{I}_R$.



(b) Vertices ordered by $\hat{I}_R$.

Figure 4.20: Edge and vertex importance measures for Sioux Falls.

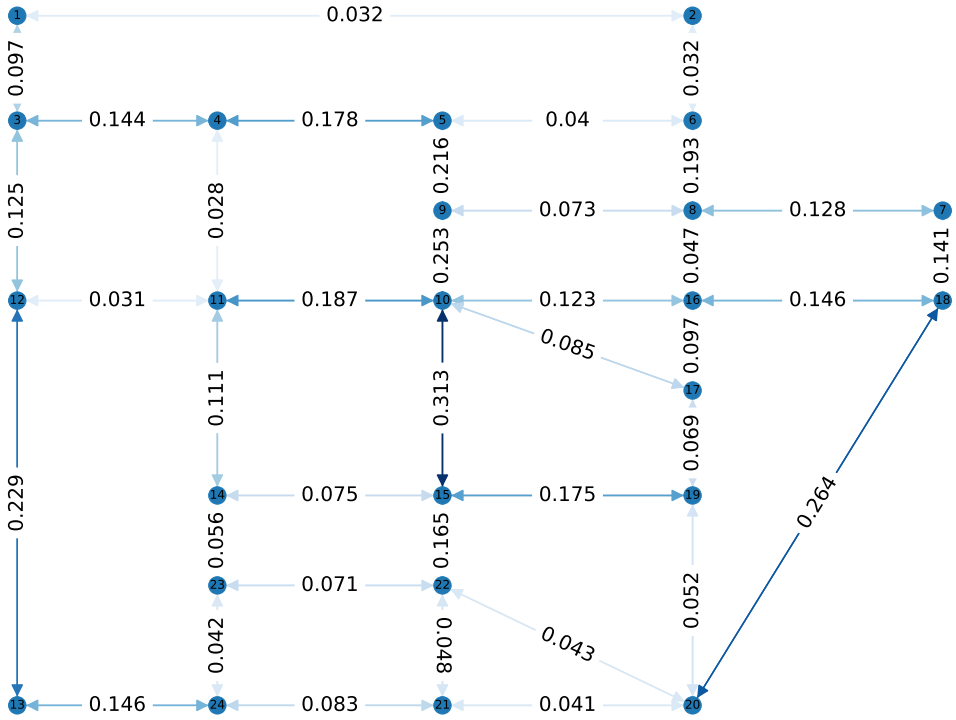


Figure 4.21: Sioux Falls - Network edges coloured by importance $\hat{I}_R$.

4.5 A Paradoxical Effect

Based on the motivating examples (Section 4.4.3) and intuition, it is easy to make the assumption that, when disconnecting a commodity and thus the resulting demand, the total travel time of the network will be reduced. Whilst most of time this may the case it is not true for all nonatomic selfish routing games. The following example (Figure 4.22) demonstrates this, although it is noted that it is the rerouting of the demand for commodity $(1, 2)$ that causes the travel time to increase irrespective of the size of the demand for commodity $(3, 2)$. It does however counter such a statement as "A disconnected commodity results in a decrease in total travel time and thus $\hat{I}_R(S) < 0$".

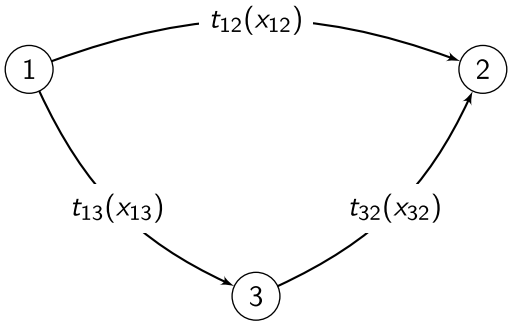


Figure 4.22: Amended Pigou network.

The example has two commodities $(1,2)$ and $(3,2)$ with respective demands, $d_{12} = 10$ and $d_{32} = 10$. The cost functions of the edges are $t_{12}(x_{12}) = x_{12}$, $t_{13}(x_{13}) = \frac{x_{13}}{10}$ and $t_{23}(x_{23}) = \frac{x_{23}}{10}$. Given that there is only one path between OD pair $(3,2)$ the flow on the path must be $f_{32} = 10$ and thus the edge cost $t_{32}(x_{32})$ is now given by $t_{32}(x_{32} + 10)$. For OD pair $(1,2)$, let x represent the amount of flow on the lower path π_{12}^- consisting of vertices $1-3-2$ and $d_{12}-x$ be the amount routed on path π_{12}^+ consisting of vertices $1-2$, then the equilibrium costs can be computed by solving the equation:

$$\begin{aligned} c_{12}^+ &= c_{12}^- \\ t_{12}(d_{12} - x) &= t_{13}(x) + t_{32}(x + 10) \\ 10 - x &= \frac{x}{10} + \frac{x + 10}{10} \\ x &= \frac{15}{2} \end{aligned}$$

Equilibrium occurs when the flow on path π_{12}^+ is $f_{12}^+ = \frac{5}{2}$, the flow on path π_{12}^- is $f_{12}^- = \frac{15}{2}$ and the flow on path π_{32} is $f_{32} = \frac{15}{2} + 10 = \frac{35}{2}$. The cost of the paths is therefore $c_{12}^+ = c_{12}^- = \frac{5}{2}$ and $c_{32} = \frac{7}{4}$.

Therefore, the total cost is

$$\sum_{k \in K} \lambda_k d_k = c_{12}^+ d_{12} + c_{32} d_{32} = \frac{5}{2} \cdot 10 + \frac{7}{4} \cdot 10 = \frac{85}{2}.$$

Consider the consequence of the removal of edge $(3,2)$. Upon removal of this edge, OD pair $(3,2)$ is disconnected and d_{32} can no longer be routed. More worryingly it now disconnects the lower path π_{12}^- between OD pair $(1,2)$ and thus all demand d_{12} is routed on the upper path π_{12}^+ incurring a cost of $c_{12}^+ = 10$. The total travel time

in the network is now given by

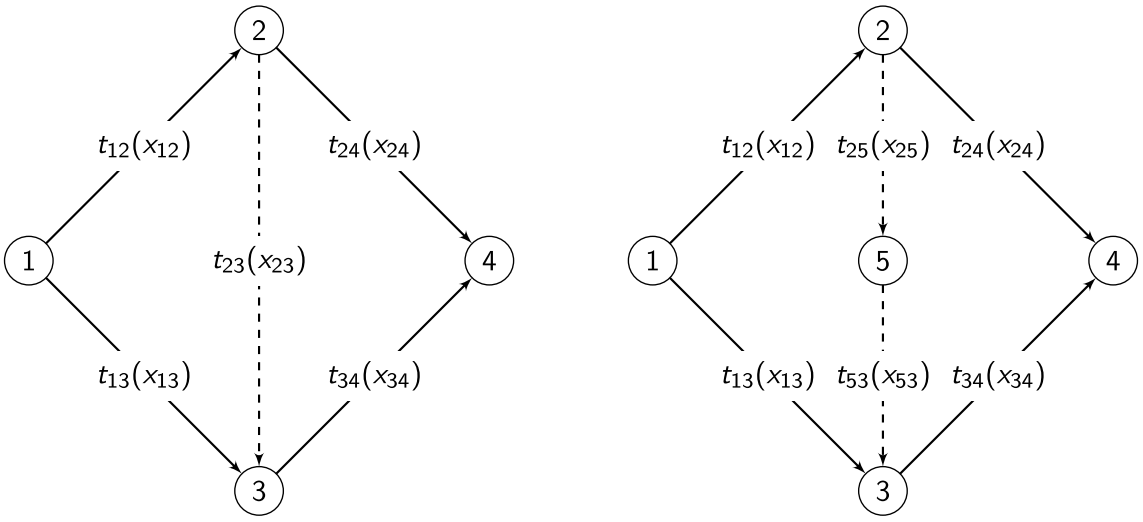
$$\sum_{k \in K} \lambda_k d_k = c_{12}^{+} d_{12} = 10 \cdot 10 = 100.$$

Clearly, the removal of demand of the network (due to the disconnect of the commodity) has resulted in a worsening of the total travel time.

4.5.1 Removal of a Vertex Without Associated Demand

A similar question to pose is whether the removal of a vertex (and the edges incident) with no associated demand always results in a worsening of the total travel time?

It is trivial to produce an example that rejects this by a simple amendment to the Braess network. Consider the Braess network and amended Braess network given in Figure 4.23. Clearly the removal of vertex 5 results in the removal of edges $(2, 5)$ and $(5, 3)$ and the disconnect of vertices 2 and 5. This is the same as the removal of edge $(2, 3)$ in the original network which results in an improvement of the total travel time. Therefore, the removal of vertex 5 in the amended network results in an improvement of the total travel time.



(a) Braess network.

(b) Amended Braess network.

Figure 4.23: Amended Braess network.

4.6 Results: Impact of Changes on Importance

The following section considers two possible scenarios that can impact a network and demonstrates the use of the measure proposed in Section 4.4 to identify critical edges and vertices.

4.6.1 Positioning of External Demand

To demonstrate the use of the proposed measure, consideration is given to the scenario in which demand enters a network at a single point and leaves at another single point with results presented and analysed. A motivating example is the case of Manhatten Island which has a fairly uniform grid like structure and just a few major entry/exit points. It can also be viewed as external demand which passes through a network.

The network used for this analysis can be found in Appendix C.5 and is a grid network with uniform congestion functions. Two scenarios are analysed, first demand enters and exits from the middle, i.e. a single commodity $(11, 15)$. Second, from the corners, i.e. a single commodity $(1, 25)$. These are depicted in Figures 4.26a and 4.26b, respectively.

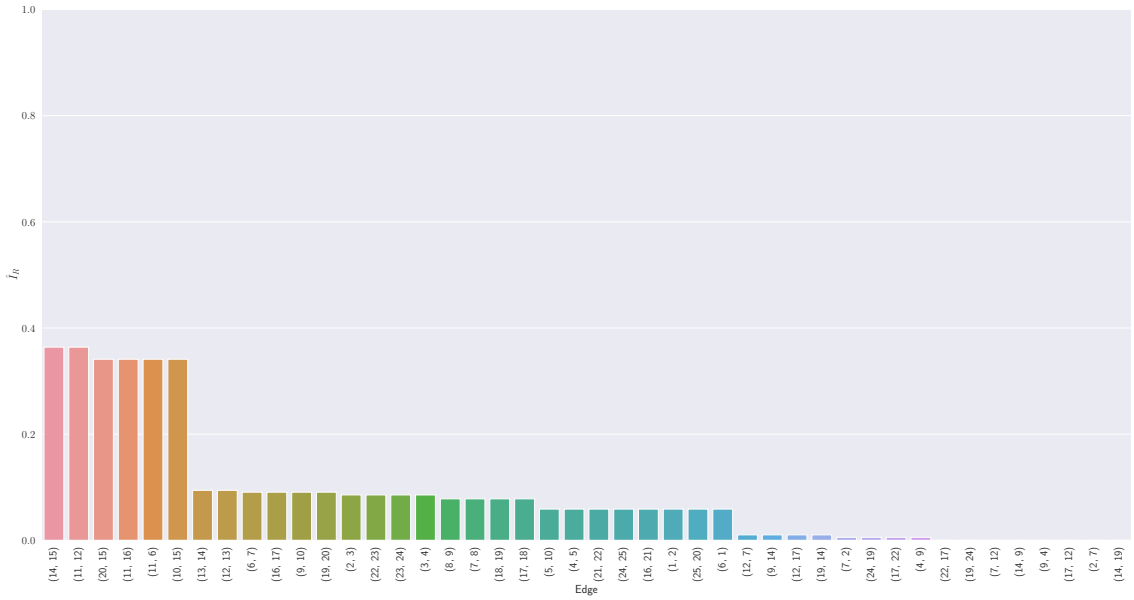
Figures 4.24, 4.25 display the edge and vertex importance measure in order of their ranking $\hat{I}_R$ for the middle and corner situations respectively. Note edges with an importance of 0 are not shown. Figure 4.26 displays the network with edge labels corresponding to their importance, the edges have also been shaded from light to dark to visually represent the importance. Note that the values present in Figure 4.26 are the values generated by $\hat{I}_R$ and are not normalised. Whilst there may be a case for normalising the values for ease of analysis within a single network, when comparing the two similar networks, the non-normalised values provide more insight into how the change in commodity location affects the importance of the edges.

The corner-to-corner network has only two entry edges and two exit edges and due to the uniformity of the grid these have equal importance, there is also a symmetry across the network, with edges further towards the opposite corners having a lower importance, this symmetry can be explained by two observations. First, the shortest path hop length is 8 and any path that that gets closer in Euclidean dis-

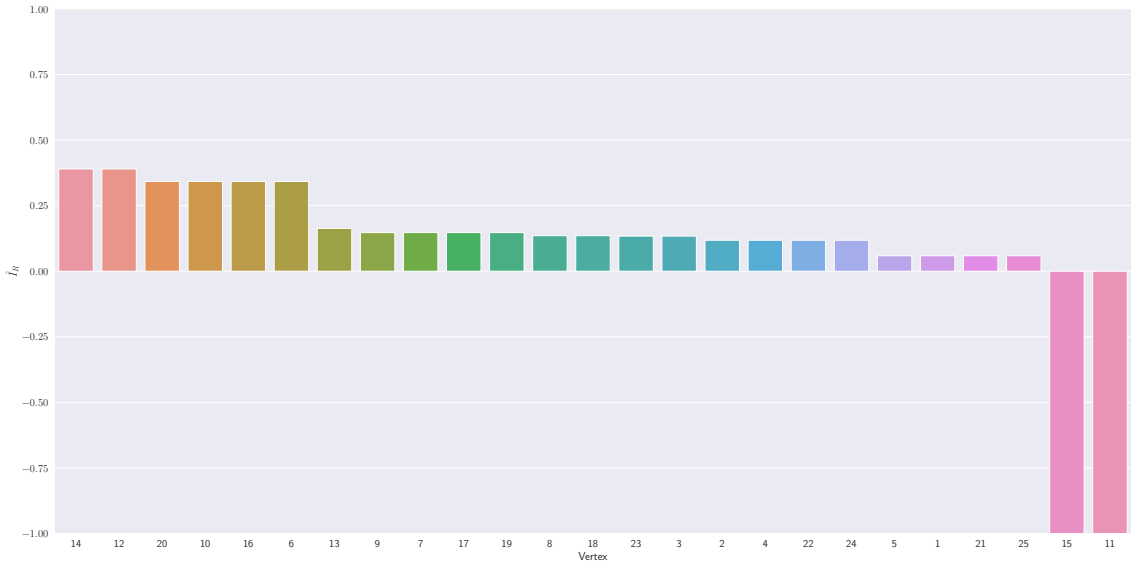
tance to the exit results in a hop length of 8. Second, edges have a certain number of paths of length 8 that pass through them. For example, the two extreme cases are the entry and exit edges, i.e. half the paths must pass through edge (1,2), (1,6), (20,25) and (24,25) and, the edges incident to the corner vertices 5 and 21, i.e. (4,5), (5,10), (16,21) and (21,22), which have only one path of hop length 8 (traversing the outer vertices) that passes through them. Therefore, the removal of edges that are used by smaller number of shortest paths will have less of an impact on the total travel time.

The side-to-side network has three edges for entry and exit and for the given demand these are almost of equal importance as the number of paths passing through these edges are almost uniformly distributed. Edges which are parallel to the centre path 11-12-13-14-15 have an importance value that decreases the further from this path they are. This is due to the increase in hop length of the paths that deviate from the centre path. It is also apparent from Figures 4.24a and 4.25a that the magnitude of the importance values differ for the entry/exit edges of both networks with the four most important edges in the corner-to-corner network having a higher importance value than the six most important edges in the side-to-side network. This is easily explained as the importance generated by the demand is effectively distributed across more edges in the side-to-side network and thus lower.

On inspection of Figures 4.24b and 4.25b it is apparent that the two vertices attached to the single commodity in both networks result in a negative value of $\hat{I}_R$ due to the disconnect in the commodity and the removal of the demand for the network.

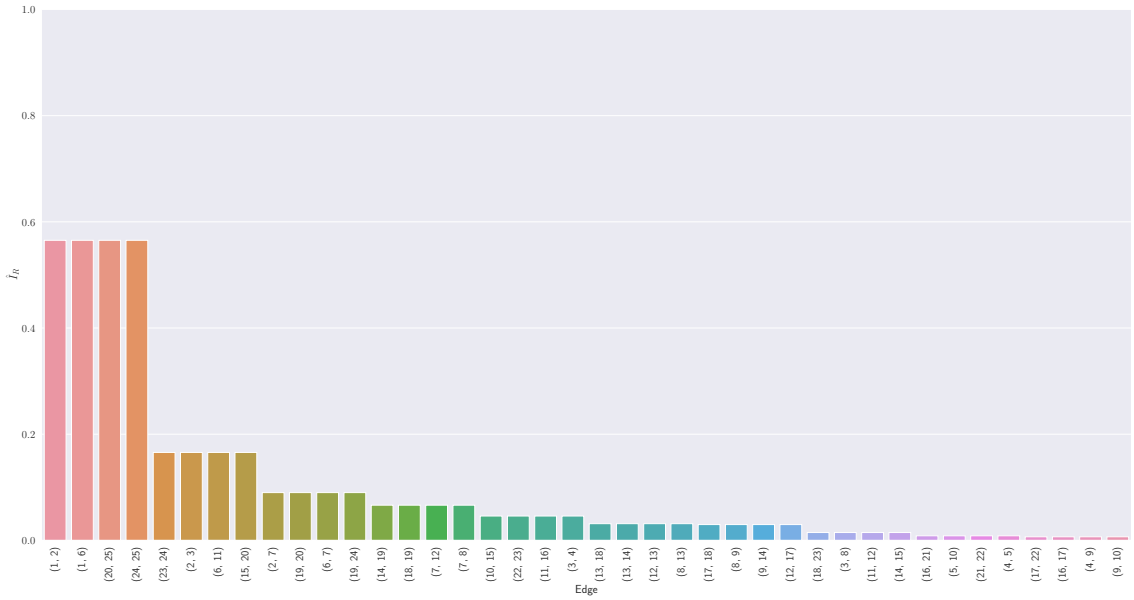


(a) Edges ordered by $\hat{I}_R$ for side-to-side network.

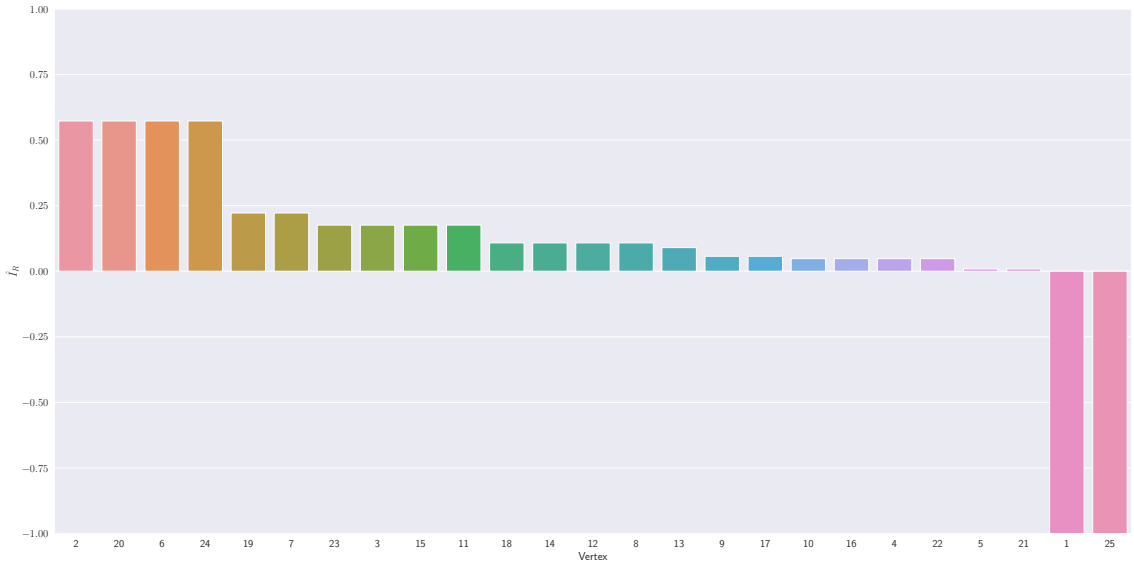


(b) Vertices ordered by $\hat{I}_R$ for side-to-side network.

Figure 4.24: Edge and vertex importance measures for side-to-side external demand.

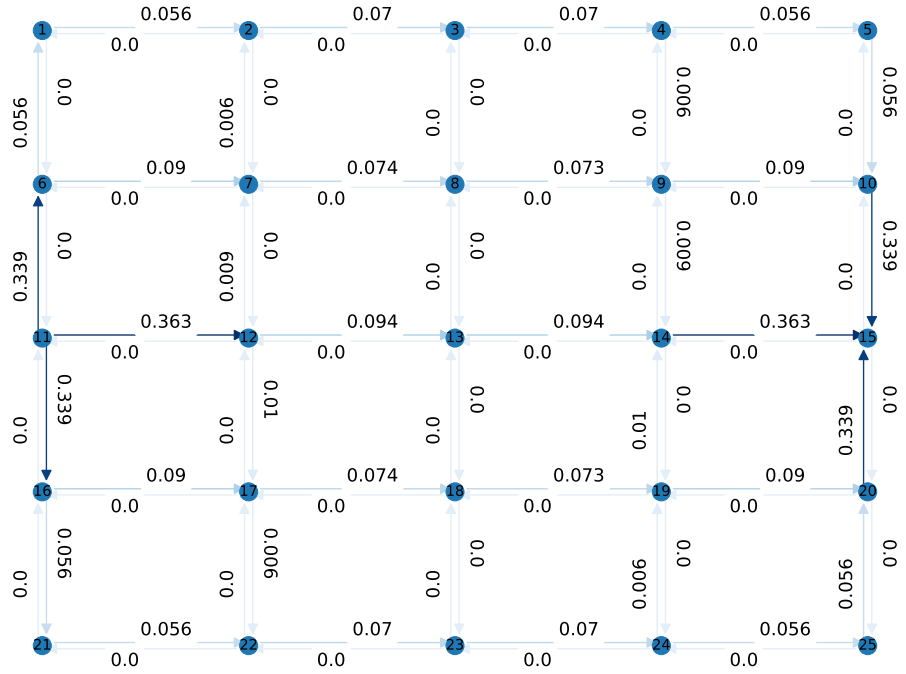


(a) Edges ordered by $\hat{I}_R$ for corner-to-corner network.

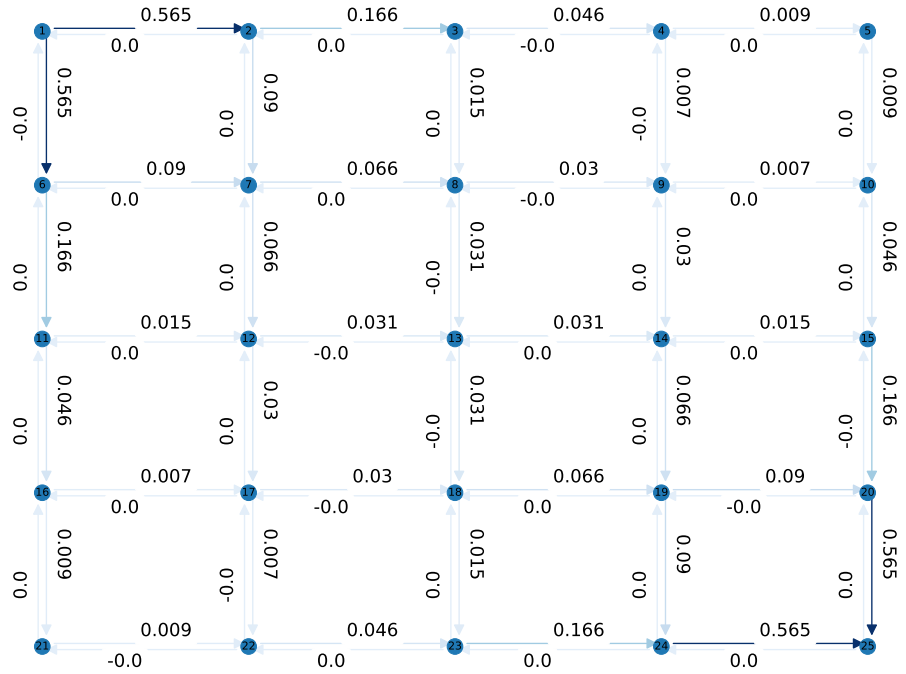


(b) Vertices ordered by $\hat{I}_R$ for corner-to-corner network.

Figure 4.25: Edge and vertex importance measures for corner-to-corner external demand.



(a) Side to side.



(b) Corner to corner.

Figure 4.26: Network edges coloured by importance $\hat{I}_R$.

4.6.2 Changes to Localised Demand

Clearly, a dramatic change in the demand for every commodity will result in a different flow pattern for the routed demand. It is worth, though, examining the situation when localised demand is reduced/increased; does this change the order of importance?

Figure 4.27 displays the results of varying demand on the network found in Appendix C.5 when it is enters and exits from the side. There a couple of observations. First, clearly the order is predominantly the same, with the same sets of edges incurring the same importance and the differences between these sets maintained, i.e. the same edges are generally important irrespective of the demand. Second, not only does the importance measure rank the edges, it also provides a magnitude of difference; in the case of the first 6 edges, they are more important in the original and increased demand than in the reduced demand.

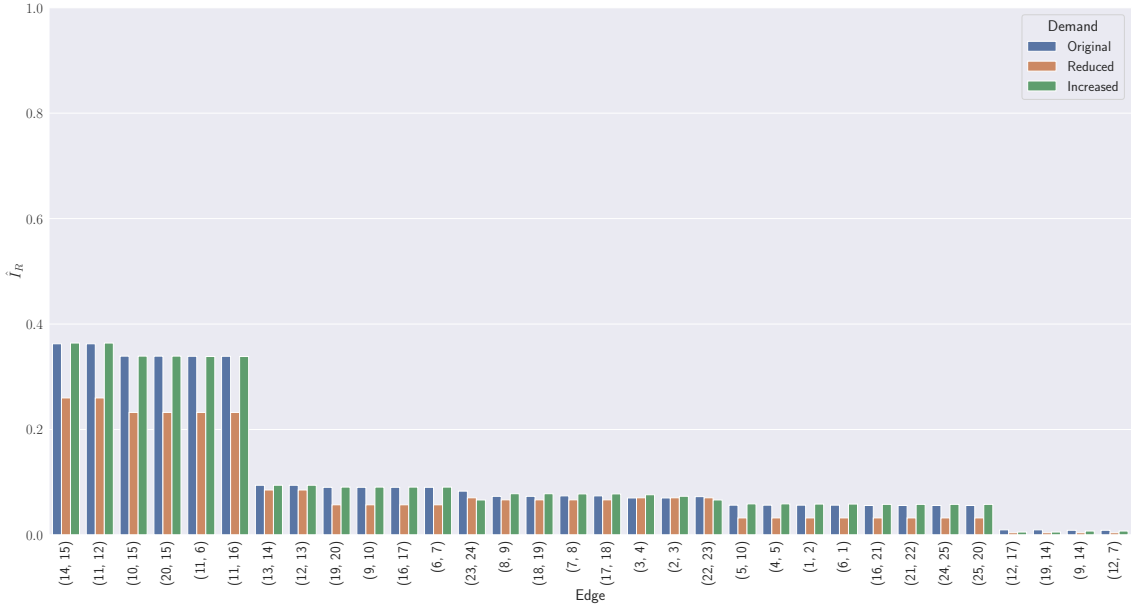


Figure 4.27: Demand changes for side-to-side network.

4.7 Chapter Summary

Answering the question of how critical components of a network are is a difficult task and is dependent on first answering the question of what matters, that is what should be measured?

In this chapter a number of existing network science and demand-based measures were analysed to assess how they correlated and performed with respect to the change in total travel time. The results generated show that the demand-based measures correlate well because of their strong connection with the change in total travel time. These measures also correlated well with the edge flow x_e, which is somewhat expected as the change in edge flow has a direct impact on the change in total travel time, i.e the two are not independent and tend to correlate highly. The high correlation with the change in total travel time also brings into question their merit, why use one of the measures? The normalised measures do, however, allow for comparison between problems and indicated the strength of one component with another, thus allowing the comparison between different topologies. The standard network science measures do not provide a strong enough correlation to be reliable in making meaningful predictions.

Issues were also identified with the demand-based measures and a new measure and revised normalised importance were presented which solve these issues and provide more context to the situation at hand. The issues identified were mainly due to a commodity or commodities being disconnected in the network, thus leading to the question of what it means to disconnect (remove) demand from the network. The answer to this question is one of interpretation; however, it was argued that it is natural to consider demand displacement (perhaps temporarily) due to some edge or vertex closure. Therefore, it is necessary that the importance measures reflect these situations.

Based on the Braess paradox, it was noted that it is impossible to determine whether a commodity had been disconnected purely from the proposed measure or the change in total travel time. Therefore, it is important to consider other measures in conjunction to this measure (and similar measures) which reflect changes in demand or path connectivity such as the Unsatisfied Demand and the OD k-connectivity.

An amended Pigou network was also used to demonstrate a paradoxical effect that disconnecting a commodity which removed demand from the network may actually increase the total travel time, that is, less users on the network, yet they experience a worse travel time. Another effect was also demonstrated on an amended

Braess network that showed that the removal of a vertex which had no associated demand can actually improve the total travel time at equilibrium.

Reciprocal and revised normalised importance allows for comparison across networks and correlates correctly with ΔT; however, it must be used in conjunction with a measure such as the unsatisfied demand to fully understand why for certain vertices, removal results in an improvement of travel time. One should be careful making comparisons across topologies and should use a robust change in demands and congestion functions to assess the importance of edges and vertices.

4.7.1 Future Work

To establish the extent to which measures can help predict both the flow and the change in total travel time, a more significant statistical analysis is required. Further analysis can be done on different topologies, e.g. in terms of their density, i.e. high average degree.

Examining the way in which the number of paths are reduced for particular topologies is also of interest as a more uniform grid structure is likely to have more similar paths (hop length) available per commodity than a less symmetric structure. In particular, examining paths which do not share edges, i.e. have paired alternative segments, is of interest and may have underlying connections with Menger's theorem [105] and other such interesting ideas related to networks.

The OD k-connectivity also represents another area of interest and the question of whether the number of paths that are shut down, or relative percentage shut down, by an edge/vertex removal is correlated with the change in total travel time.

Finally, robust optimisation techniques and supervised and unsupervised machine learning techniques could be employed on larger scale networks to ascertain whether a model can be trained that is able to predict the importance of an edge/vertex with a particular selfish routing network.

Chapter 5

Atomic Selfish Routing: An Application of Mixed-Integer Linear Programming

Chapter Preface

The standard multi-commodity flow problem (MCFP) shares a number of characteristics with the atomic selfish routing game and can be adapted to solve multi-channel routing.

This chapter presents a simplified model of a multi-channel transportation network and solves the routing/planning problem via a MILP. Such models are becoming more feasible for large scale problems, such as the MILP model developed and solved for modular 3-dimensional bin packing of plant equipment by O'Neill, Wrigley and Bagdasar [131] (Appendix A.4).

The problem presented in this chapter is to illustrate that this approach is an extension of the standard multi-commodity flow problem whereby the flows are discrete and unsplittable and considered over a multigraph.

Chapter Keywords

Multi-commodity Flow Problem, MCFP, Multi-channel Routing, Mixed-integer Linear Programming, MILP, Unsplittable Flow, Atomic Selfish Routing

5.1 Introduction

The multi-commodity flow problem is the basis of pre-planned routing of demand through a fixed cost network and is a fixed cost version of the system optimal nonatomic routing game [62, 2, 88]. In recent years there has been a huge advance in MILP solvers, such as Gurobi [73], taking advantage of parallelisation techniques and processing in the cloud. The industry leader, Gurobi, now reports only failing to solve 118 of its internal test set of over 6000 models compared with 716 and approximately a 50 times speed up in solution time in 8 years [73]. For a comprehensive benchmarking of discrete optimisation software and its history see [109, 110]. Work on large MILP models is now viable such as the MILP model developed and solved for 3-dimensional packing of plant equipment by O'Neill, Wrigley and Bagdasar [133] (Appendix A.4). Alongside traditional techniques there is much work being done into deep learning for combinatorial optimisation problems [76, 103, 71, 77, 170], such as network optimisation. The recent advances in deep learning [6], such as deep reinforcement learning and the stunning results of Deepmind [111] and Open AI [10] provide real possibilities of new intelligent routing for combinatorially large network problems.

The problem presented in this chapter is an interesting extension of the standard multi-commodity flow problem whereby the flows are discrete and unsplittable and considered over a multigraph. When the costs on the edges of the network are congestion functions, through a transformation, the presented problem is an atomic selfish routing game.

This chapter is organised as follows. The multi-channel problem is outlined in Section 5.2 and an example given. In Section 5.3 the path-based MCFP is given and the column generation technique explained. Section 5.4 presents the solution methodology along with pros and cons of the proposed method and example results. The problem is then considered with congesitible edge functions in Section 5.5. Finally, Section 5.6 concludes the chapter.

5.2 Problem Statement

Consider an undirected network $G = (V, E)$ for which each pair of vertices (i, j) may have multiple edges, each edge $e \in E$ has a fixed cost c_e and an identifier l, e.g. there may be 3 edges between vertices $(2, 5)$ represented as $(2, 5, 1), (2, 5, 2), (2, 5, 3)$ and the identifiers would be $1, 2, 3$ respectively. There are a set of commodities (vertex pairs) K and for each $k \in K$ an associated atomic (unsplittable) demand d_k, referred to as goods, to be routed. The routing of these goods will be referred to as a delivery.

Two constraints are imposed on the problem. First each edge has a capacity u_e for transporting flow (demand) and, second, the routing of a commodities demand must make use of the same identifier on its path through the network, e.g. if a commodities demand is routed on an edge with identifier 1, it must only make use of other edges with identifier 1 in forming a path to route the demand. The objective of the problem is to minimise the cost of routing all demand between commodities.

The above is summarised in the following list:

- The network is an undirected multigraph, i.e. there can be multiple edges between two vertices.
- Each edge has a fixed cost c_e.
- For each pair of vertices the edges between them have a unique identifier. $l = \{1, \ldots, n\}$, where n is the maximum number of edges between any pair of vertices.
- Each commodity $k \in K$ must route the goods which have a demand (weight) of d_k.
- There can be multiple goods on each edge, but the total weight of the goods must be less that the edge capacity u_e.
- Each delivery must use edges with the same identifier when routing.
- The objective is to minimise the total cost.

Figure 5.1 shows a very simple example network with 3 possible edges between each vertex. There is a single commodity (1,4) for which goods must be routed.

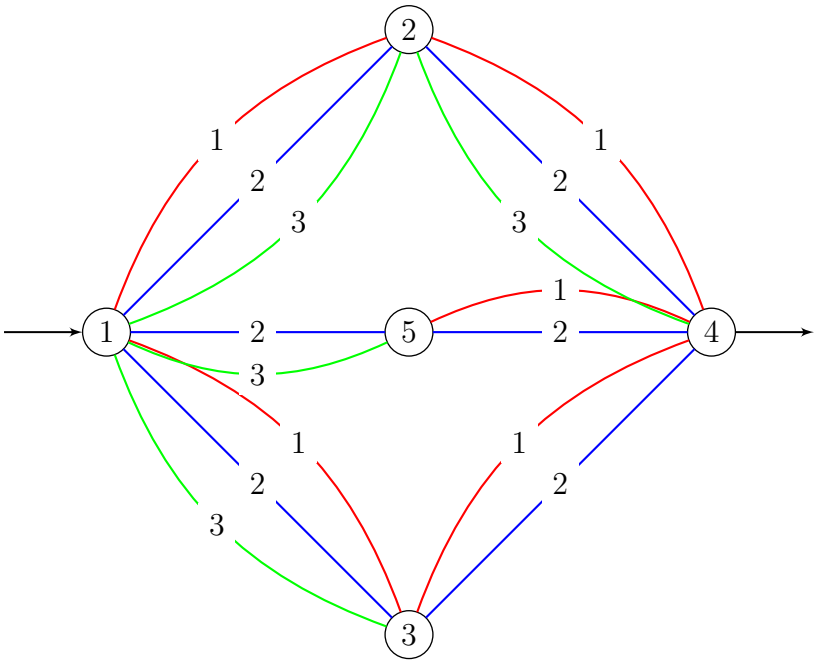


Figure 5.1: Multigraph with a single commodity $(1, 4)$.

5.2.1 Relevance to Delivery Route Planning and Telecommunications

Large scale transportation and logistics companies may have multiple existing delivery routes in place which have a given capacity. Therefore, there is a need for pre-planned delivery route optimisation to optimise the use of the existing infrastructure.

Alongside this, the continued growth of globalised communication networks and the ever present desire for faster routing has led to the advent of 5G technology and the potential for greater control and optimisation through network slicing.

The above could be seen as a very simplified high transportation network whereby the routing of demand is planned ahead of time, or is a time snapshot of demand to be routed all at a particular time. Goods can be seen as packets of data (required services) and the edges seen as a particular slice of the 5G network.

5.3 Methodology

5.3.1 Multi-commodity Flow Problem

The basic MCFP can be stated as a linear programming problem either in terms of the flow on edges or the flow on paths. Due to its combinatorial nature, the edge flow formulation quickly becomes unmanageable in terms of the number of constraints for even industry leading solvers [2]. Whilst the path-based formulation can have an exponential number of paths, column-generation techniques provide a means of finding a solution in the search space. The MCFP is given by:

$$\text{minimise} \qquad \sum_{k \in K} \sum_{\pi \in \Pi_k} c_\pi^k f_\pi^k = \sum_{e \in E} c_e x_e \tag{5.1a}$$

subject to

$$\sum_{\pi \in \Pi_k} f_\pi^k = d_k, \qquad \forall k \in K \tag{5.1b}$$

$$\sum_{k \in K} \sum_{\pi \in \Pi_k : e \in \pi} f_\pi^k \leq u_e, \qquad \forall e \in E \tag{5.1c}$$

$$f_\pi^k \geq 0, \qquad \forall \pi \in \Pi \tag{5.1d}$$

Note that the objective 5.1a can be either the path based formulation

$$\sum_{k \in K} \sum_{\pi \in \Pi_k} c_\pi^k f_\pi^k,$$

that is the cost of the path flows, or

$$\sum_{e \in E} c_e x_e,$$

the cost of the edge flows. For more detail see [2, 144].

5.3.2 Column Generation Method

Due to the exponential growth of the number of possible paths, the following column generation technique can be utilised.

1. Initialise a subset of paths $S_k \subseteq \Pi_k$ for each commodity $k \in 1, \ldots K$.

2. Solve the restricted master problem for paths $S = \bigcup_k S_k$ to obtain a solution.

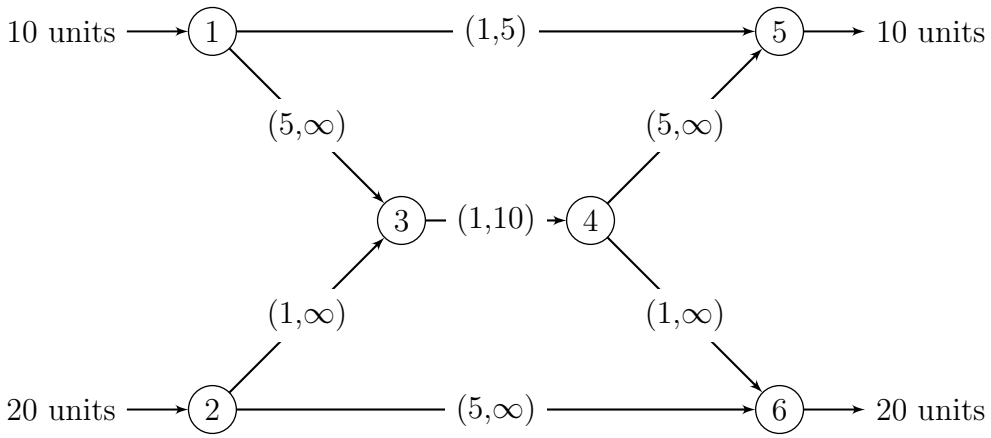


Figure 5.2: Example multi-commodity flow problem.

3. Check to see if the solution is optimal (see optimality conditions for the MCFP [2]) If not, find new paths to add to S and return to step 2.

The means by which the paths are initially obtained in step 1 and new paths selected in step 3 is a problem in of itself.

5.3.3 Illustrative Example

To demonstrate the column generation method the example given in Figure 5.2 from [2] is presented. Note, this example uses a directed graph but this causes no issues with the model given by (5.1), which can handle both undirected and directed networks. In the case that an undirected edge is represented by bidirectional directed edges, care would be needed when considering the capacity constraints of the original undirected network, that is the flow on both bidirectional edges would need to be summed and compared with the capacity of the original undirected edge.

For this example, each edge has an unordered 2-tuple containing the cost and the capacity of the edge, i.e. for a given edge e, (c_e, u_e). There are four possible paths for the two commodities, π_1, $1-5$; π_2, $1-3-4-5$; π_3, $2-6$; π_4, $2-3-4-6$ with respective costs $1, 11, 5, 3$.

To illustrate the column generation method, the paths will be generated via the k shortest paths for each commodity. For iteration 1, generate the shortest path, for iteration 2 generate the 2 shortest paths, etc. Clearly as there are a maximum of 2 paths per commodity, the column generation method will take at most 2 iterations.

For the first iteration paths $1-5$ and $2-3-4-6$ are chosen; the capacity of

$1-5$ is only 5 units and, thus, the 10 units cannot be routed for commodity (1,5), i.e. the restricted master problem is unfeasible.

In iteration 2, the 2 shortest paths for each commodity are generated for the restricted master problem. In this case, this is the entire set of paths, the restricted master problem is feasible and its optimal solution is the optimal solution of the master problem (see Figure 5.3).

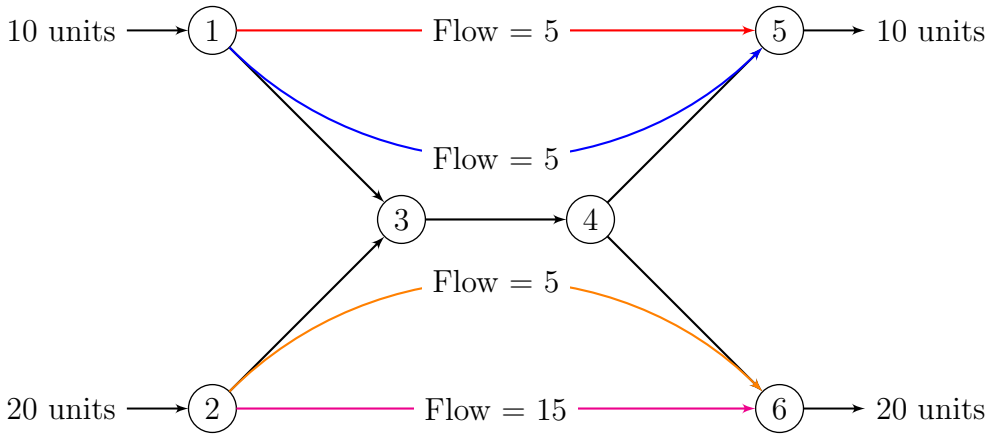


Figure 5.3: Optimal path flow solution for example given in Figure 5.2.

5.3.4 Extending MCFP with Integer Constraints

Using an additional set of binary variables to represent path use, the standard MCFP can be extended to limit the paths per commodity.

For each path $\pi \in \Pi_k$ introduce a binary variable $\delta_\pi^k \in \{0,1\}$ and the inequalities given by (5.2g) and (5.2h). This ensures that, for any path with positive flow, $f_\pi^k > 0 \implies \delta_\pi^k = 1$ and $\delta_\pi^k = 0$ otherwise. The binary variables essentially state whether a path is used or not. Equation (5.2f) sums the used paths for a commodity and requires them to be equal to N_k.

The extended MCFP with integer constraints is given by the following optimisation problem:

$$\text{minimise} \qquad \sum_{k\in K}\sum_{\pi\in\Pi_k} c_\pi^k f_\pi^k = \sum_{e\in E} c_e x_e \tag{5.2a}$$

subject to

$$\sum_{\pi\in\Pi_k} f_\pi^k = d_j, \qquad \forall k \in K \tag{5.2b}$$

$$\sum_{k\in K}\sum_{\pi\in\Pi_k : e\in\pi} f_\pi^k = x_e, \qquad \forall e \in E \tag{5.2c}$$

$$x_e \leq u_e, \qquad \forall e \in E \tag{5.2d}$$

$$f_\pi^k \geq 0, \qquad \forall \pi \in \Pi_k, \forall k \in K \tag{5.2e}$$

$$\sum_{\pi\in\Pi_k} \delta_\pi^k = N_k, \qquad \forall k \in K \tag{5.2f}$$

$$\delta_\pi^k \leq M f_\pi^k, \qquad \forall \pi \in \Pi_k, \forall k \in K \tag{5.2g}$$

$$f_\pi^k \leq M \delta_\pi^k, \qquad \forall \pi \in \Pi_k, \forall k \in K \tag{5.2h}$$

$$\delta_\pi^k \in \{0,1\}, \qquad \forall \pi \in \Pi_k, \forall k \in K \tag{5.2i}$$

$$N_k \in \mathbb{Z}, \qquad \forall k \in K. \tag{5.2j}$$

Note that the equation (5.2c) is not necessary if the objective used is the path-based formulation,

$$\sum_{k\in K}\sum_{\pi\in\Pi_k} c_\pi^k f_\pi^k,$$

and thus constraint (5.2d) can also be rewritten as,

$$\sum_{k\in K}\sum_{\pi\in\Pi_k : e\in\pi} f_\pi^k \leq u_e, \qquad \forall e \in E. \tag{5.3}$$

Figure 5.4 updates the example given by Figure 5.2 by increasing the capacity of edge $(3,4)$ from 10 to 30, this slight change helps better illustrate the solution to the problem defined by (5.2) with $N_k = 1,\ k \in \{1,\ldots,K\}$, i.e. one path per commodity.

Clearly, it is infeasible to route all 10 units between vertices 1 and 5 on path $1-5$ as the capacity is only 5 units, thus 10 units are routed on path $1-3-4-5$, leaving a residual capacity of 20 units for edge $(3,4)$. Examining the 20 units that need to be routed between vertices 2 and 6, the shortest is $1-3-4-6$ and the minimum capacity on any edge of the path is the 20 units left on edge $(3,4)$. Therefore, the 20 units

can be routed along path $1-3-4-6$. The solution is displayed in Figure 5.5 with $f_{\pi_1}=0, f_{\pi_2}=10, f_{\pi_3}=0$ and $f_{\pi_4}=20$ and respective costs $c_{\pi_1}=5, c_{\pi_2}=11, c_{\pi_3}=5$ and $c_{\pi_4}=3$. Thus, the total cost $\sum_{k\in K}\sum_{\pi\in\Pi_k} c_\pi^k f_\pi^k = 170$.

It is clear that if the capacity of edge $(3,4)$ had been less than 30, then the 20 units would have been forced to be routed along path $2-6$. This illustrates that, even in one path routing, the edge capacities play a key role in determining the optimal solution. Had all the edges had an infinite capacity, then the solution would have been the trivial one of just picking the shortest path for each commodity, namely $f_{\pi_1}=10, f_{\pi_2}=0, f_{\pi_3}=0$ and $f_{\pi_4}=20$ and a total cost of 70.

As in the example above, if the allowed paths per commodity N_k are all equal to one, then the formulation given by (5.2) can be utilised to solve the multi-channel problem by the transformation outlined in Section 5.3.5, converting the multigraph network into a graph.

Again a solution can be found by applying the column generation method presented in 5.3.2; however, the addition of the binary variables means that the optimality conditions are no longer valid. Therefore, different stopping criteria must be proposed. A simple solution is to stop after a fixed number of iterations whereby no improvement has been observed.

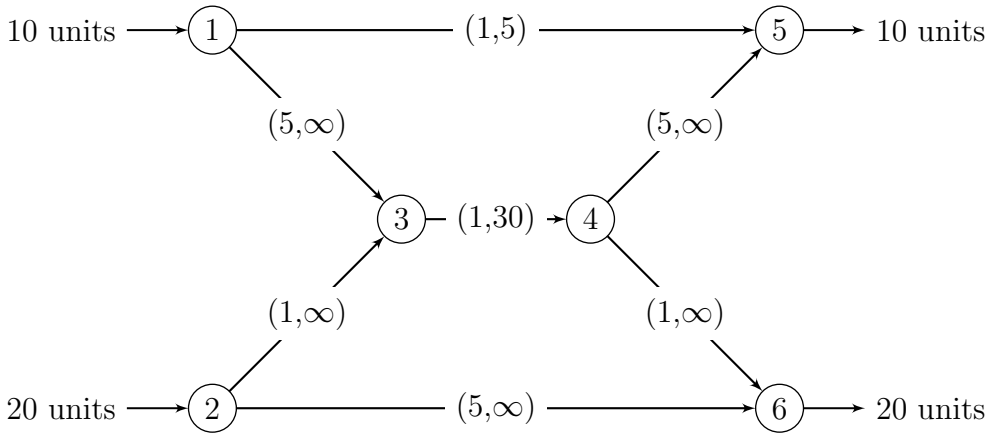


Figure 5.4: Edge (3,4) capacity increased from 10 to 30.

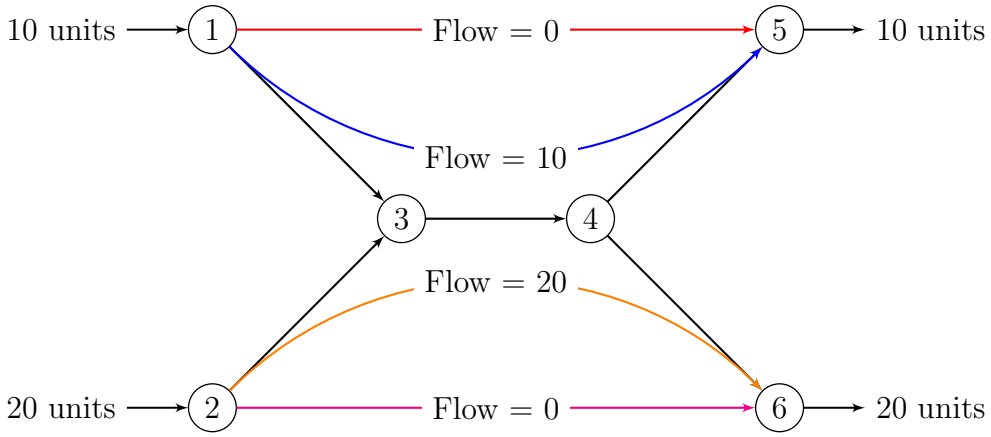


Figure 5.5: Optimal path flow solution for (5.2) with $N_k = 1,\ \forall k \in \{1, \ldots, K\}$.

5.3.5 Transformation of the Network from a Multigraph to a Graph

A simple transformation can be made to transform the multigraph into a graph that allows the for standard MCFP given by (5.1) or the amended integer constrained problem given by (5.2) to be utilised. The transformation has the following rules:

1. For each edge identifier l, create a copy of the network which contains only edges with the identifier l.

2. For each commodity (i, j), create a dummy pair of vertices (s_i, t_j) and connect s_i to i and t_j to j in every copy of the network.

Figure 5.6 transforms the multigraph example given by Figure 5.1. This illustrative example showcases the simplicity and elegance of this transformation, that is, once a delivery chooses an edge to leave the dummy source s, it is constrained to the relevant copy of the network and can only reach the dummy sink t by a path through this network.

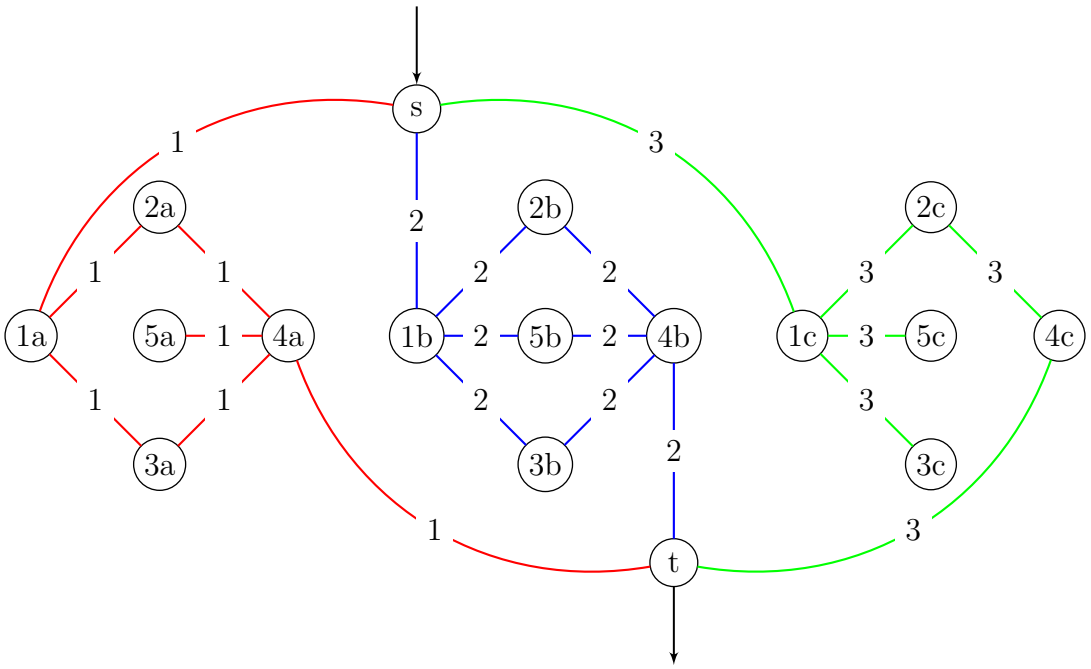


Figure 5.6: Transformation of multigraph into a graph via introduction of dummy vertices for commodities.

5.3.6 Shortest Path Technology

A key component to the column generation method given in Section 5.3.2 is that it requires the determination of a set of paths, an obvious choice in a network optimisation problem, where the objective is to minimise the total cost, is the shortest paths between commodities. In this particular problem, unlike selfish routing games, the cost of the edges are fixed and thus the shortest path per commodity will not change. Therefore, multiple shortest paths must be generated per commodity during further iterations by utilising an appropriate algorithm, for example Yen's Algorithm for the k shortest paths [40].

Owing to the size of networks it is important to consider the speed at which shortest paths can be found, as for n commodities, at least n shortest paths are needed per iteration of the column generation method. In recent years, the prevalence of graph libraries such as NetworkX, igraph, graph-tool and the continued increase in interest and usage of graph databases such as Neo4j, vast networks of millions of vertices can be processed extremely quickly. Speeds are reported of 0.06 seconds for processing a graph of ∼200K vertices and ∼1M edges [97, 156, 160].

As aforementioned in Section 2.2.5 Schnek and Nokel found that on the largest test networks, contraction hierarchies resulted in a speedup factor of 42 [138].

For urban transportation networks in the UK, the average number of vertices was ∼6.5K and the average number of edges, ∼15K, see Table 4.2. Thus, even for a large number of commodities, the time required for computing the shortest paths is likely more than manageable in most cases.

5.3.7 Pros of the Proposed Methodology

The following is a list of pros of the proposed methodology:

1. Solution to restricted master problem is optimal.
2. Column generation method allows a tractable means of exploring the search space.
3. Problem now fits into a standard fertile area of research, which allows advance techniques for solving MCFP problems to be utilised.

5.3.8 Criticisms of the Proposed Methodology

The following is a list of criticisms of the proposed methodology:

1. Whilst the restricted master problem is optimal, the extensions to the problem mean the optimality conditions of the standard MCFP are no longer usable.
2. An iteration of the column generation method is subject to solving the MILP restricted master problem which, for a large enough network, would prove costly.
3. Creates a much larger network because of the network copies required.
4. How are the paths selected in each iteration? That is, how to effectively explore the path search space.

5.3.8.1 Quantifying the Size of the Problem

Criticism 2 motivates a discussion on the size of the problem generated by this methodology and the practical implications.

Given the complexity of solving LP and MILP problems increases as the number of constraints and, in the case of MILP, integer variables increase it is worth considering the number of constraints for the formulations (5.1), (5.2) and the additional integer variables required for (5.2).

Denote the number of paths generated by $n_\Pi = \sum_{k \in K} \sum_{\pi \in \Pi_k} 1$, then the number of constraints for the MCFP given by the formulation (5.1) is $|K| + |E| + n_\Pi$, where $|K|$ and $|E|$ are the number of commodities and edges, respectively. The integer formulation given by (5.2) has $2|K| + |E| + 3n_\Pi$ constraints with an additional $n_\Pi + |K|$ integer variables.

For an average of n paths per commodity this would result in a total of $n_\Pi = n|K|$ paths and thus $|E| + (3n + 2)|K|$ constraints and $(n + 1)|K|$ integer variables. Clearly, the number of constraints grows by an order $O(|E|+|K|)$ and the number of integer variables by an order $O(|K|)$. In general (see Section 4.2.4.3) transportation networks are sparse in terms of the number of edges; however, given a number of origin/destination vertices (that either emit or receive flow) N_{od}, the number of ways to choose a commodity $k \in K$ (origin-destination pair) $\binom{N_{od}}{2} = \frac{N_{od}(N_{od}-1)}{2}$. Therefore, the number of commodities has a significant influence on the size of the MILP formulation.

If we consider the Chicago Regional network that is the standard for testing methods and algorithms (see Section 2.2.4), seen in context as a reasonably large network, we can quickly see that this would prove infeasible even for modern solvers as the number of commodities is $|K| = 1,360,427$.

For problems of a smaller size, Dai and Zhang [44] conclude that the column generation method is good for solving the MCFP on small networks reporting solutions to a Chinese airport network with 183 vertices and 2995 edges in 5186s and 5967s for GLPK (GNU Linear Programming Kit) [4] and Gurobi, respectively. Clearly through the use of distributed computing, this could be sped up.

With respect to MILP problems in general, Gurobi reports solving the MILP seymour model which has 4944 constraints and 1372 (binary) variables in 633s on 32 machines [73]. This suggests that the methodology for small/medium problems could prove feasible, although an experimental evaluation is required to conclude what a feasible problem size is for a given number of distributed machines to solve

in a reasonable time.

5.4 Preliminary Implementation and Results

The proposed column generation method given in Section 5.3.4 relies on solving the restricted master problem given by (5.2). An implementation of this MILP using Gurobi can be found at [129].

Table 5.1 displays the solutions for the problem given in Figure 5.2 with varying edge capacities for the cases $N_k = 1,\ k \in \{1, 2\}$ and $N_k = 2,\ k \in \{1, 2\}$ given by the Gurobi implementation. The solution given for the problem outlined by Figure 5.4 is displayed as case 3 of Table 5.1 and corroborates with the solution given in Figure 5.5. Even in this simple example, it is clear that not only do the edge capacities have a key role in determining the path flow solution and cost but the number of paths per commodity N_k is also instrumental and this is most obvious when comparing cases 2 and 5 in which the objective cost is almost halved from 210 to 120.

Whilst this alone does not solve the problem outlined in 5.2, it is a key component in the column generation method and a full implementation of the column generation method requires a means of transforming a multigraph (Section 5.3.5) and iteratively producing shortest paths (Section 5.3.6) for the restricted master problem to solve.

Case	u_{13}	u_{15}	u_{23}	u_{26}	u_{34}	u_{45}	u_{46}	N_k	f_{π_1}	f_{π_2}	f_{π_3}	f_{π_4}	Cost
1	∞	∞	∞	∞	∞	∞	∞	1	10	0	0	20	70
2	∞	5	∞	∞	29	∞	∞	1	0	10	20	0	210
3	∞	5	∞	∞	30	∞	∞	1	0	10	0	20	170
4	∞	∞	∞	∞	∞	∞	∞	2	10	0	0	20	70
5	∞	5	∞	∞	29	∞	∞	2	5	5	0	20	120
6	∞	5	∞	∞	30	∞	∞	2	5	5	0	20	120

Table 5.1: Comparison of solutions for (5.2) with varying edge capacities.

5.5 Extension to Atomic Selfish Routing Games

The problem stated in Section 5.2 and the MILP given by (5.2) can be seen as a fixed cost version of a system optimal selfish routing game with the additional constraint that the demand for each commodity is routed on at most N_k paths. To extend the fixed cost to a nonatomic selfish routing game, let the cost of each edge e be given by a congestible function $t_e(x_e)$.

In the case of unsplittable flow, i.e. $N_k = 1,\ \forall k \in \{1, \ldots, K\}$, this is a weighted atomic selfish routing game with fixed cost edge functions (see Section 1.4.1). For the system optimal objective, small problems can be solved using the piecewise linear approximation approached outlined in Section 2.3.3. The MINLP is given by:

$$\text{minimise} \qquad \sum_{e \in E} x_e t_e(x_e) \tag{5.4a}$$

subject to

$$\sum_{\pi \in \Pi_k} f_\pi^k = d_j, \qquad \forall k \in K \tag{5.4b}$$

$$\sum_{k \in K} \sum_{\pi \in \Pi : e \in \pi} f_\pi^k \leq u_e, \qquad \forall e \in E \tag{5.4c}$$

$$f_\pi^k \geq 0, \qquad \forall \pi \in \Pi \tag{5.4d}$$

$$\sum_{\pi \in \Pi_k} \delta_\pi^k = N_k, \qquad \forall k \in K \tag{5.4e}$$

$$\delta_\pi^k \leq M f_\pi^k, \qquad \forall \pi \in \Pi_k, \forall k \in K \tag{5.4f}$$

$$f_\pi^k \leq M \delta_\pi^k, \qquad \forall \pi \in \Pi_k, \forall k \in K \tag{5.4g}$$

$$N_k \in \mathbb{Z}, \qquad \forall k \in K. \tag{5.4h}$$

5.6 Chapter Summary

In this chapter, a multi-commodity flow problem with extensions to a multi-graph and unsplittable flow was presented. The proposed solution methodology allowed for the problem to be solved by conventional solvers without the need for a bespoke method. It was noted that for large problems, even the current fastest solvers may struggle to find a solution in a reasonable time when limited to a single or a limited number of cores.

It was also shown how the problem was a fixed cost version of a weighted atomic selfish routing game,

5.6.1 Future Work

Whilst for the fixed cost problem presented in 5.2 the proposed solution presented in this chapter is of interest, there is a need to test the solution methodology on a large dataset using industry standard commercial solvers to assess its use and the size of problem that is practicably solvable.

Consideration would need to be given to how paths are selected in the column generation method to effectively explore the search space. Owing to the fixed cost of the paths, the shortest paths would not change and therefore this is not a trivial problem and there is not an obvious way in which to select additional paths.

The atomic routing formulation presented in Section 5.5 requires development of bespoke methods as the edge cost functions are congestible. Methods such as those presented in Section 2.3 can be utilised and further developed.

Finally, given the recent advances in deep learning and, specifically, combinatorial optimisation, such methods could be developed and benchmarked against the state-of-the-art MILP solvers.

Chapter 6

Summary and Conclusions

Chapter Preface

This chapter summarises the work covered in this thesis. Each chapter is treated in turn and highlighting work, highlights and contributions made within those chapters areas of potential future research.

Chapter Keywords

Selfish Routing, Nonatomic, Atomic, Multi-criteria, Criticality, Congestion Games, Multi-commodity Flow Problem, Network Flow

6.1 Introduction

The purpose of this chapter is to collate the work of this thesis. Chapter One introduced the broader topics pertinent to selfish routing detailing preliminaries for subsequent chapters, relevant literature and aims/objectives for this thesis.

Chapter Two surveyed and detailed algorithms for nonatomic/atomic selfish routing games, including some novel algorithms. Algorithms given in this chapter were then used for chapters three and four to provide the necessary analysis. Chapter Three explored multi-criteria optimisation of selfish routing games and means of measuring the efficiency against the social optimal solutions. Chapter Four explored the importance of selfish routing games and measures by which this could

be assessed. Chapter Five presented a novel multi-channel problem associated with the multi-commodity flow problem and proposed a solution methodology based on mixed-integer linear programming.

6.2 Chapter One

Chapter One introduced the broader area of congestion games and in particular, nonatomic and atomic selfish routing games. Preliminaries were given in Section 3.2.1 and previous work and definitions of nonatomic and atomic selfish routing was presented in Sections 1.3 and 1.4. Section 1.5 surveyed the literature on multi-objective optimisation with respect to selfish routing problems and Section 1.6 surveyed literature on assessing how critical given components were in selfish routing games.

Finally Section 1.7 set out the aims and areas to which this thesis makes contributions.

6.3 Chapter Two

Chapter Two (based on work presented in [9, 132], Appendix A.1 and Appendix A.2) starts by presenting the history of algorithms used to solve the nonatomic selfish routing problem with reference to its history as a tool used in the transportation industry for planning. These algorithms were then grouped into three types; link-based, path-based and bush-based and the general method for each presented alongside some particular instances of each

Algorithms for the weighted atomic selfish routing game were then given, including those proposed in Section 2.3. Finally, online learning algorithms involving bandit machines were presented for solving the unweighted atomic selfish routing game.

6.4 Chapter Three

Chapter Three (based on the work in [131], Appendix A.3) investigated how the efficiency and suboptimality of equilibria with regards to the system optimal solution

could be extended when considering multi-criteria. This chapter began with a study into multi-criteria selfish routing which considered fuel consumption as an additional criterion to travel time. Weighted sum models were explored and presented and then results generated for a simple 3 edge parallel network. These results were used to identify the set of Pareto optimal solutions and then classify the suboptimality of the equilibrium solution with respect to the ideal point. In addition, through the use of manipulating "free" parameters, equilibrium solutions that dominate the prior solution were found and demonstrated.

The technique introduced to classify the suboptimality of the equilibrium solution with respect to the ideal point was then extended to allow for use of a general distance metric.

6.5 Chapter Four

Chapter Four assessed and analysed metrics pertaining to nonatomic congestion games and their use for providing information about the importance of network components. Such information is useful in understanding the resilience and vulnerability of networks containing flow.

The chapter began by presenting some useful network science measures that are not reliant on network flow and demonstrating them on a large case study of flow networks, namely 111 major cities and towns in England and Wales. These measures are then also correlated with the change in total travel time for the Sioux Falls network and grid-based network.

Existing demand-based measures were then outlined and analysed and a number of issues highlighted. A new measure based on the reciprocal of the travel cost was then proposed and analysed and a number of associated theorems given.

Due to the nature of disconnecting a vertex and thus disrupting the demand, it was noted that total travel cost tended to lower. A paradoxical effect was then given which acted as a counter example, demonstrating that disconnecting a commodity and it's associated demand could lead to an increase in total travel cost. An amendment to the Braess network also provided an example that demonstrated that removal of a vertex without associated demand could lead to an improvement

in travel time.

Finally, the newly proposed measure was used to assess the impact on the importance of network components by independently considering the positioning of external demand in grid network and the change in demand for a single commodity.

6.6 Chapter Five

Chapter Five introduces an extension to the classic multi-commodity flow problem whereby the routing operates on a multigraph and the the number of paths allowed per commodity is restricted. When the number of paths is restricted to one for a commodity then the flow is said to be unsplittable; if this is the case for all commodities in the network then, given a transformation to the multigraph, this is actually a weighted atomic selfish routing game.

The chapter begins by defining a network flow problem on a multigraph which requires goods be transported on a single channel. The classic multi-commodity flow problem is then given and the column generation method for the path-based formulation presented. The classic problem is then extended to allow for a limit to the number of paths that a commodity can use.

The method for solving the stated problem is then given, demonstrating how a transformation to the multigraph allows the path-limited extension of the multi-commodity flow problem to be utilised to solve the problem.

Finally, the model is extended to allow for congestible functions and, when paths are limited to one for all commodities, the model solves the weighted atomic selfish routing game.

6.7 Critical Reflection

This section highlights the contributions of this thesis and how these meet the initial aims set out in Section 1.7. For clarity these are restated and addressed in order.

1. *To investigate and implement methods for the use in further analysis of nonatomic selfish routing concepts and atomic selfish routing concepts.*

Chapter 2 presented methods for solving both nonatomic and atomic selfish routing problems. Gradient descent proved efficient enough to handle the problems presented in this thesis, but larger problems would require an efficient implementation of a bush-based method. Methods presented for solving the atomic selfish routing game present a means to solving moderate sized problems, but tractable solutions are needed to deal with larger problems such as Chicago Regional.

The novel probabilistic algorithms based on the concept of bandit machines demonstrated how independent decision making led to a reasonably stable state, mirroring how human decision making would happen and underlining the idea of user equilibrium. Applications of online learning algorithms and human behaviour offer a potential avenue for interesting applications such as A/B testing. In such situations were a large volume of data is generated over a period of time, the tractability of these algorithms ceases to be an issue as the algorithm will progressively converge towards a solution as the amount of data processed increases.

In situations were tractability matters, the algorithms/methods for atomic routing games makes larger scale problems difficult to tackle and work is required on developing solutions that can converge quickly, perhaps through evolutionary means or deep learning.

2. *To consider multi-criteria selfish routing and understand/extend the use of the Price of Anarchy/Stability in multi-criteria scenarios.*

Consideration of a bi-criteria selfish routing problem was investigated, and this provided insight into the nature of equilibrium solutions within a multi-criteria space. Results were generated that demonstrated how "free" parameters could lead to better equilibrium solutions that dominated within the multi-criteria space. The importance of this is that real-world problems could be improved through simple changes that prove to be cost effective whilst simultaneously satisfying multiple criteria.

In addition, novel work was done on the Price of Anarchy/Stability that provided a definition of how to measure suboptimality for a multi-criteria problems using a distance metric. This allows for classification of networks and their propensity to be suboptimal when at equilibrium in a multi-criteria context. This work has natural extensions to consider within the wider game theoretic context and given the number of choices one could make in the definition it requires further investigation from a theoretical and practical standpoint.

3. *To analyse the existing means of assessing the criticality of network components under equilibrium against the primary function, total travel cost.*

Chapter 4 provided two main studies, the first looked into the size, structure and properties of the road networks of 111 major cities and towns in England and Wales. The results demonstrated that although these networks were organically formed, they exhibit particular structure and sparsity. This work motivates further study into the structure of networks of a particular type (i.e. their use) and their properties which could then be applied to classification work on the size of said networks and the appropriate solution methodologies. The second study looked into measures for assessing the criticality of network components with respect to the total travel time, i.e. the effect of removing edges, vertices or a combination of these. A number of issues were identified with existing measures and a new measure was proposed that dealt with these issues. This measure was then demonstrated in classifying the importance of edges and vertices.

Paradoxical effects were also presented that extended the seminal work of Pigou and Braess. These have potential consequences for network planners in that careful consideration must be given to the removal of a vertex irrespective of the demand attached as it can have the opposite effect to the one desired. i.e. The expectation is that removal of vertex with associated demand would intuitively improve the travel time and that removal of a vertex with no associated demand would worsen the network. Neither are guaranteed.

6.8 Potential Applications of Work

The following section lists potential applications of the work presented in this thesis. For clarity this is presented in order of chapter.

6.8.1 Chapter 2

- The mixed-integer linear programming program with piece-wise linear approximation as an objective provides a possible way to extend the routing game to include constraints on edges, vertices etc through the use of integer variables
- Provides models and methods which can be solved by a mixed-integer nonlinear programming (MINLP) utilising the advancement in recent research into solvers
- Online learning algorithms could be used in path like situations. e.g. an A/B testing scenario whereby the choices are are connected via a network could be modelled by an atomic routing game.

6.8.2 Chapter 3

- The basic nonatomic selfish routing formulation could be expanded to include edge-capacity constraints and emissions modelling via pricing schemes
- When considering multi-criteria optimisation problems, the extensions to the price of anarchy allow for classification of networks and their propensity to be suboptimal when at equilibrium.
- The proof of concept that changes to small "free" parameters can have a large positive effect on the solution could be utilised in a real-world scenario for manipulation of a network to better cope with simultaneously satisfying multiple criteria.

6.8.3 Chapter 4

- Analysis of urban topologies that can be used as a dataset to explore the classification of these networks and extrapolate potential features.

- Prediction of the importance of edges through machine learning techniques utilising the measures and data.

6.8.4 Chapter 5

- Allows a complex multi-channel flow problem to be solved using MCFP. This allows existing state-of-the-art solutions in combinatorial optimisation to be utilised. Such transformations may prove practically useful in computer and communication network optimisation.

6.9 Conclusion

This thesis has outlined a number of problems related to network equilibrium flows which deal with the three objectives set out in Section 1.7. The main highlights are methods for solving atomic selfish routing games (Chapter 2) [9, 132]; extending equilibrium modelling to multi-criteria and dealing with efficiency measures in a multi-criteria context (Chapter 3) [131]; identifying the size of the network problem in urban transportation networks through the use of network measures and identifying critical components pertaining to the total (average) travel cost (Chapter 4); and extending the problem into a multi-channel (multigraph) instance via consideration of the multi-commodity flow problem and atomic selfish routing (Chapter 5).

The work presented in this thesis was the subject of a number of publications [9, 132, 131, 133, 130] and provides a solid foundation for future work with each chapter outlining potential avenues of investigation to capitalise on.

Bibliography

[1] A.C.Pigou. *The Economics of Welfare*. Palgrave Macmillan, 1920.

[2] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[3] J. Almond. Traffic assignment with flow-dependent journey times. *Vehicular Traffic Science*, pages 222–234, 1967.

[4] Andrew Makhorin. GLPK - GNU Project - Free Software Foundation (FSF), 2018.

[5] E. Anshelevich, A. Dasgupta, J. Kleinberg, E. Tardos, T. Wexler, and T. Roughgarden. The price of stability for network design with fair cost allocation. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 295–304. IEEE, 2004.

[6] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017.

[7] B. Awerbuch, Y. Azar, and A. Epstein. The price of routing unsplittable flow. *SIAM Journal on Computing*, 42(1):57–66, 2005.

[8] N. Y. Aydin, Y. Casali, H. Sebnem Duzgun, and H. R. Heinimann. Identifying changes in critical locations for transportation networks using centrality. *Lecture Notes in Geoinformation and Cartography*, pages 405–423, 2019.

[9] O. Bagdasar, S. Berry, S. O'Neill, N. Popovici, and R. Raja. Traffic assignment: Methods and simulations for an alternative formulation of the fixed

demand problem. *Mathematics and Computers in Simulation*, 155:360–373, 1 2019.

[10] B. Baker, I. Kanitscheider, T. Markov, Y. Wu, G. Powell, B. McGrew, and I. Mordatch. Emergent tool use from multi-agent autocurricula. *CoRR*, 2019.

[11] C. Balijepalli and O. Oppong. Measuring vulnerability of road network considering the extent of serviceability of critical road links in urban areas. *Journal of Transport Geography*, 39:145–155, 2014.

[12] S. Bandaru and K. Deb. Metaheuristic techniques. *Decision sciences*, pages 693–750, 2016.

[13] H. Bar-Gera. Origin-based algorithm for the traffic assignment problem. *Transportation Science*, 36(4):398–417, 11 2002.

[14] H. Bar-Gera. Primal method for determining the most likely route flows in large road networks. *Transportation Science*, 40(3):269–286, 2006.

[15] H. Bar-Gera. Traffic assignment by paired alternative segments. *Transportation Research Part B: Methodological*, 44(8-9):1022–1046, 2010.

[16] H. Baroud, K. Barker, J. E. Ramirez-Marquez, and C. M. Rocco S. Importance measures for inland waterway network resilience. *Transportation Research Part E: Logistics and Transportation Review*, 62:55–67, 2014.

[17] M. J. Beckmann. A continuous model of transportation. *Econometrica: Journal of the Econometric Society*, pages 643–660, 10 1952.

[18] M. J. Beckmann. The partial equilibrium of a continuous space market. *Weltwirtschaftliches Archiv*, 71:73–89, 1953.

[19] M. J. Beckmann, C. B. McGuire, and C. B. Winsten. *Studies in the Economics of Transportation*. 1956.

[20] E. V. Belmega, P. Mertikopoulos, R. Negrel, and L. Sanguinetti. Online convex optimization and no-regret learning: Algorithms, guarantees and applications. *CoRR*, abs/1804.04529, 4 2018.

[21] S. Berry, V. Lowndes, and M. Trovati. *Guide to Computational Modelling for Decision Processes*. 2017.

[22] J. Błażewicz, K. H. Ecker, E. Pesch, G. Schmidt, and J. Węglarz. Scheduling Computer and Manufacturing Processes. *Scheduling Computer and Manufacturing Processes*, 2001.

[23] G. Boeing. *Methods and measures for analyzing complex street networks and urban form*. PhD thesis, University of California, Berkley, 2017.

[24] G. Boeing. OSMnx: A Python package to work with graph-theoretic OpenStreetMap street networks. *Journal of Open Source Software*, 2(12), 2017.

[25] G. Boeing. A multi-scale analysis of 27,000 urban street networks. *Environment and Planning B: Urban Analytics and City Science*, 2018.

[26] S. Boslaugh. *Statistics in a Nutshell: A Desktop Quick Reference*. O'Reilly Media, Inc, 2012.

[27] P. Bothner and W. Lutter. Ein direktes Verfahren zur Verkehrsumlegung nach dem 1. *Prinzip von Wardrop. Universität Bremen, Forschungsbereich Verkehrssysteme. Arbeitsbericht*, (1), 1982.

[28] D. Bouneffouf and I. Rish. A survey on practical applications of multi-armed and contextual bandits. *arXiv preprint arXiv:1904.10040*, 2019.

[29] D. Bouneffouf, I. Rish, and C. Aggarwal. Survey on applications of multi-armed and contextual bandits. In *2020 IEEE Congress on Evolutionary Computation (CEC)*, pages 1–8, 2020.

[30] D. Boyce. Beckmann's transportation network equilibrium model: Its history and relationship to the Kuhn–Tucker conditions. *Economics of Transportation*, 2(1):47–52, 3 2013.

[31] D. Boyce and H. Williams. *Forecasting Urban Travel: Past, Present and Future*. Edward Elgar Publishing Ltd., 2015.

[32] D. E. Boyce, H. S. Mahmassani, and A. Nagurney. A retrospective on Beckmann, McGuire and Winsten's Studies in the Economics of Transportation. *Papers in Regional Science*, 84(1):85–103, 3 2005.

[33] D. Braess. Über ein Paradoxon aus der Verkehrsplanung. *Unternehmensforschung*, 12(1):258–268, 1968.

[34] M. Bruynooghe, A. Gilbert, and M. Sakarovitch. Une methode d'affectation du trafic. In *Fourth International Symposium on the Theory of Traffic Flow*, pages 198–204, 1969.

[35] G. Burtini, J. Loeppky, and R. Lawrence. A survey of online experiment design with the stochastic multi-armed bandit. *arXiv preprint arXiv:1510.00757*, 2015.

[36] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[37] K. Chen, D. Zhu, Y. Hu, and J. Liu. The bound of price of anarchy for multi-class and multi-criteria traffic equilibrium problem. *Journal of Systems Science and Systems Engineering*, 21(1):77–93, 2012.

[38] J. Cohen, A. Héliou, and P. Mertikopoulos. Learning with bandit feedback in potential games. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6372–6381, 12 2017.

[39] J. E. Cohen and P. Horowitz. Paradoxical behaviour of mechanical and electrical networks. *Nature*, 352(6337):699–701, 8 1991.

[40] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT press, 2009.

[41] J. R. Correa, A. S. Schulz, and N. E. Stier-Moses. Selfish routing in capacitated networks. *Mathematics of Operations Research*, 29(4):961–976, 2004.

[42] S. Dafermos. A multicriteria route-mode choice traffic equilibrium model. *Bulletin of the Greek Mathematical Society*, 24:13–32, 1983.

[43] S. C. Dafermos and F. T. Sparrow. The traffic assignment problem for a general network. *Journal of Research of the National Bureau of Standards, Series B*, 73(2):91–118, 1969.

[44] W. Dai, J. Zhang, and X. Sun. On solving multi-commodity flow problems: An experimental evaluation. *Chinese Journal of Aeronautics*, 30(4):1481–1492, 8 2017.

[45] G. B. Dantzig, S. F. Maier, and Z. F. Lansdown. Application of decomposition to transportation network analysis. Technical report, United States. Dept. of Transportation. Office of the Secretary, 1976.

[46] G. B. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Operations research*, 8(1):101–111, 1960.

[47] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou. The complexity of computing a Nash equilibrium. *SIAM Journal on Computing*, 39(1):195–259, 1 2009.

[48] K. Deb. *Multi-objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, 2001.

[49] M. S. Dehghani, G. Flintsch, and S. McNeil. Impact of road conditions and disruption uncertainties on network vulnerability. *Journal of Infrastructure Systems*, 20(3):04014015, 9 2014.

[50] H. Demirel, M. Kompil, and F. Nemry. A framework to analyze the vulnerability of European road networks due to Sea-Level Rise (SLR) and sea storm surges. *Transportation Research Part A: Policy and Practice*, 81:62–76, 11 2015.

[51] S. Devarajan. A note of network equilibrium and noncooperative games. *Transportation Research Part B: Methodological*, 15(6):421–426, 1981.

[52] R. Dial. Algorithm B: Accurate traffic equilibrium (and how to bobtail Frank-Wolfe. *Volpe National Transportation Systems Center, Cambridge, MA*, 1999.

[53] R. B. Dial. A probabilistic multipath traffic assignment model which obviates path enumeration. *Transportation research*, 5(2):83–111, 1971.

[54] R. B. Dial. A model and algorithm for multicriteria route-mode choice. *Transportation Research Part B: Methodological*, 13(4):311–316, 1979.

[55] R. B. Dial. Bicriterion traffic assignment: Basic theory and elementary algorithms. *Transportation Science*, 30(2):93–111, 1996.

[56] R. B. Dial. Bicriterion traffic assignment: Efficient algorithms plus examples. *Transportation Research Part B: Methodological*, 31(5):357–379, 1997.

[57] R. B. Dial. A path-based user-equilibrium traffic assignment algorithm that obviates path storage and enumeration. *Transportation Research Part B: Methodological*, 40(10):917–936, 2006.

[58] Q. Du, K. Kishi, N. Aiura, and T. Nakatsuji. Transportation network vulnerability: Vulnerability scanning methodology applied to multiple logistics transport networks. *Transportation Research Record*, 2410(1):96–104, 2014.

[59] M. Ehrgott. *Multicriteria Optimization - 2nd Edition.* Springer Berlin Heidelberg, 2005.

[60] A. Fabrikant, C. Papadimitriou, and K. Talwar. The complexity of pure Nash equilibria. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 604–612, 2004.

[61] G. Fagiolo. Clustering in complex directed networks. *Physical Review E*, 76(2):26107, 2007.

[62] M. C. Ferris and J. S. Pang. Engineering and economic applications of complementarity problems. *SIAM Review*, 39(4):669–713, 1997.

[63] M. Florian, I. Constantin, and D. Florian. A new look at projected gradient method for equilibrium assignment. *Transportation Research Record*, 2090(1):10–16, 2009.

[64] D. Fotakis and Dimitris. Congestion games with linearly independent paths: Convergence time and price of anarchy. *Theory of Computing Systems*, 47(1):113–136, 2010.

[65] M. Frank and P. Wolfe. An algorithm for quadratic programming. *Naval Research Logistics Quarterly*, 3(1-2):95–110, 3 1956.

[66] P. Gauthier, A. Furno, and N. E. El Faouzi. Road network resilience: how to identify critical links subject to day-to-day disruptions. *Transportation Research Record*, 2672(1):54–65, 2018.

[67] R. Geisberger, P. Sanders, D. Schultes, and D. Delling. Contraction hierarchies: Faster and simpler hierarchical routing in road networks. In *International Workshop on Experimental and Efficient Algorithms*, pages 319–333. Springer, 2008.

[68] G. Gentile. Local User Cost Equilibrium: a bush-based algorithm for traffic assignment. *Transportmetrica A: Transport Science*, 10(1):15–54, 2014.

[69] G. Gigerenzer and R. Selten. *Bounded Rationality: The Adaptive Toolbox*. MIT Press, 2001.

[70] A. Göpfert, H. Riahi, C. Tammer, and C. Zalinescu. *Variational Methods in Partially Ordered Spaces*. Springer-Verlag, New York, 2003.

[71] S. Gu and T. Hao. A pointer network based deep learning algorithm for 0–1 knapsack problem. In *2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)*, pages 473–477, 2018.

[72] J. Gudmundsson and M. Horton. Spatio-temporal analysis of team sports – A survey. *CoRR*, abs/1602.06994, 2016.

[73] L. Gurobi Optimization. Gurobi Optimizer Reference Manual, 2021.

[74] D. Han and H. Yang. The multi-class, multi-criterion traffic equilibrium and the efficiency of congestion pricing. *Transportation Research Part E: Logistics and Transportation Review*, 44(5):753–773, 2008.

[75] H. Hauptmann, W. Krelle, and K. C. Mosler. *Operations Research and Economic Theory: Essays in Honor of Martin J. Beckmann.* Springer Berlin Heidelberg, 1984.

[76] H. Hu, X. Zhang, X. Yan, L. Wang, and Y. Xu. Solving a new 3d bin packing problem with deep reinforcement learning method. *CoRR*, abs/1708.05930, 2017.

[77] J. Huang, M. Patwary, and G. Diamos. Coloring big graphs with alphagozero. *CoRR*, abs/1902.10162, 2019.

[78] J. D. Hunter. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[79] S. Ieong, R. Mcgrew, E. Nudelman, Y. Shoham, and Q. Sun. Fast and compact: A simple class of congestion games. *AAAI*, 5:489–494, 2005.

[80] INRO. Emme, 2021.

[81] B. A. Jafino, J. Kwakkel, and A. Verbraeck. Transport network criticality metrics: a comparative analysis and a guideline for selection. *Transport Reviews*, 40(2):241–264, 3 2020.

[82] J. Jahn. *Vector Optimization: Theory, Applications, and Extensions.* Springer Berlin Heidelberg, 2011.

[83] R. Jayakrishnan, W. Tsai, J. Prashker, and S. Rajadhyaksha. A faster path-based algorithm for traffic assignment. *Transportation Research Record*, 1443, 1994.

[84] E. Jenelius, T. Petersen, and L.-G. Mattsson. Road network vulnerability: Identifying important links and exposed regions. *Transportation Research A*, 40(7):537–560, 6 2006.

[85] D. S. Johnson, C. H. Papadimitriou, and M. Yannakakis. How easy is local search? *Journal of Computer and System Sciences*, 37(1):79–100, 8 1988.

[86] A. Kermanshah and S. Derrible. A geographical and multi-criteria vulnerability assessment of transportation networks against extreme earthquakes. *Reliability Engineering and System Safety*, 153:39–49, 9 2016.

[87] H. Khun and A. Tucker. Nonlinear Programming. In *Proceedings of the 2nd Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492, 1951.

[88] S. Kontogiannis and P. Spirakis. Atomic selfish routing in networks: A survey. In *International Workshop on Internet and Network Economics*, pages 989–1002. Springer, Berlin, Heidelberg, 2005.

[89] E. Koutsoupias and C. Papadimitriou. Worst-case equilibria. In *Annual Symposium on Theoretical Aspects of Computer Science*, volume 1563, pages 404–413. Springer Verlag, 1999.

[90] M. W. Krentel. On finding locally optimal solutions. In *Proceedings: Structure in Complexity Theory Fourth Annual Conference*, pages 132–133, 1989.

[91] H. W. Kuhn and A. W. Tucker. John von Neumann's work in the theory of games and mathematical economics. *Bulletin of the American Mathematical Society*, 64(3):100–123, 5 1958.

[92] Q. D. Lã, Y. H. Chew, and B.-H. Soong. *Potential Game Theory: Applications in Radio Resource Allocation*. Springer, 2016.

[93] T. Larsson and M. Patriksson. Simplicial decomposition with disaggregated representation for the traffic assignment problem. *Transportation Science*, 26(1):4–17, 1992.

[94] V. Latora and M. Marchiori. Efficient behavior of small-world networks. *Physical review letters*, 87(19):198701, 2001.

[95] L. LeBlanc. *Mathematical programming algorithms for large scale network equilibrium and network design problems*. PhD thesis, Northwestern University, Evanston, IL, 1973.

[96] L. J. LeBlanc, E. K. Morlok, and W. P. Pierskalla. An efficient approach to solving the road network equilibrium traffic assignment problem. *Transportation Research*, 9(5):309–318, 10 1975.

[97] J. Leskovec and R. Sosic. Snap: A general-purpose network analysis and graph-mining library. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(1):1–20, 2016.

[98] F. Leurent. Cost versus time equilibrium over a network. *European Journal of Operational Research*, 71(2):205–221, 12 1993.

[99] F. Leurent. The theory and practice of a dual criteria assignment model with a continuously distributed value-of-time. In *ISTTT*, pages 455–477. Pergamon, 1996.

[100] T. G. Lewis. *Network Science: Theory and Applications.* John Wiley & Sons, 2009.

[101] K. Leyton-Brown and Y. Shoham. Essentials of game theory: A concise multidisciplinary introduction. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 2(1):1–88, 2008.

[102] P. Luathep, A. Sumalee, H. W. Ho, and F. Kurauchi. Large-scale road network vulnerability analysis: A sensitivity analysis based approach. *Transportation*, 38(5):799–817, 2011.

[103] Q. Ma, S. Ge, D. He, D. Thaker, and I. Drori. Combinatorial optimization by graph pointer networks and hierarchical reinforcement learning. *CoRR*, abs/1911.04936, 2019.

[104] R. Ma, X. J. Ban, and W. Y. Szeto. Emission modeling and pricing on single-destination dynamic traffic networks. *Transportation Research Part B: Methodological*, 100:255–283, 6 2017.

[105] K. Menger. Zur allgemeinen kurventheorie. *Fundamenta Mathematicae*, 10(1):96–115, 1927.

[106] V. Miele, C. Matias, S. Robin, and S. Dray. Nine quick tips for analyzing network data. *PLOS Computational Biology*, 15(12):e1007434, 2019.

[107] S. Mishra, T. F. Welch, and M. K. Jha. Performance indicators for public transit connectivity in multi-modal transportation networks. *Transportation Research Part A: Policy and Practice*, 46(7):1066–1085, 2012.

[108] M. Mitradjieva and P. O. Lindberg. The stiff is moving - Conjugate direction Frank-Wolfe methods with applications to traffic assignment. *Transportation Science*, 47(2):280–293, 2013.

[109] H. Mittelmann. Latest benchmark results. In *Proceedings of the INFORMS Annual Conference*, pages 4–7, Phoenix, AZ, USA, 2018.

[110] H. D. Mittelmann. Benchmarking optimization software - a (hi)story. *SN Operations Research Forum*, 1(1):2, 2020.

[111] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[112] D. Monderer and L. S. Shapley. Potential Games. *Games and economic behavior*, 14(1):124–143, 1996.

[113] A. E. Motter. Improved network performance via antagonism: From synthetic rescues to multi-drug combinations. *BioEssays*, 32(3):236–245, 3 2010.

[114] J. Murchland. Road network traffic distribution in equilibrium. *paper for the Tagung Weber "Mathematische Methoden in den Wirtchafts Wissenschaften," Mathematisches Forschungsinstitut, Oberwolfach*, 1969.

[115] A. Nagurney. A multiclass, multicriteria traffic network equilibrium model. *Mathematical and Computer Modelling*, 32(3-4):393–411, 2000.

[116] A. Nagurney. The negation of the Braess paradox as demand increases: The wisdom of crowds in transportation networks. *EPL (Europhysics Letters)*, 91(4):48002, 2010.

[117] A. Nagurney and J. Dong. A multiclass, multicriteria traffic network equilibrium model with elastic demand. *Transportation Research Part B: Methodological*, 36(5):445–469, 2002.

[118] A. Nagurney and L. S. Nagurney. Braess paradox. In R. Vickerman, editor, *International Encyclopedia of Transportation,*, Amsterdam, Netherlands, 2020. Elsevier.

[119] A. Nagurney and Q. Qiang. A network efficiency measure with application to critical infrastructure networks. *Journal of Global Optimization*, 40(1-3):261–275, 2008.

[120] A. Nagurney and Q. Qiang. *Fragile Networks: Identifying Vulnerabilities and Synergies in an Uncertain World*. John Wiley & Sons, 2009.

[121] J. F. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences*, 36(1):48–49, 1950.

[122] M. K. Nasir, R. M. Noor, M. A. Kalam, and B. M. Masum. Reduction of fuel consumption and exhaust pollutant using intelligent transport systems. *The Scientific World Journal*, 2014.

[123] S. Nguyen. An algorithm for the traffic assignment problem. *Transportation Science*, 8(3):203–216, 8 1974.

[124] S. Nguyen. *Une approche unifée des méthodes d'équilibre pour l'affectation du traffic*. PhD thesis, Université de Montréal, Montréal, 1974.

[125] Y. Nie, H. M. Zhang, and D. H. Lee. Models and algorithms for the traffic assignment problem with link capacity constraints. *Transportation Research Part B: Methodological*, 38(4):285–312, 2004.

[126] Y. M. Nie. A class of bush-based algorithms for the traffic assignment problem. *Transportation Research Part B: Methodological*, 44(1):73–89, 1 2010.

[127] N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, Cambridge, 2007.

[128] Office for National Statistics. Major Towns and Cities – Methodological Note and User Guidance. 2015.

[129] S. O'Neill. PhD Thesis Repository - https://github.com/samtoneill/phdthesis, 2021.

[130] S. O'Neill and T. O'Neill. The traffic assignment problem. *Mathematics Today*, pages 52–56, 4 2021.

[131] S. O'Neill, O. Bagdasar, S. Berry, N. Popovici, and R. Raja. Modelling equilibrium for a multi-criteria selfish routing network equilibrium flow problem. *Mathematics and Computers in Simulation*, 2021.

[132] S. O'Neill, O. Bagdasar, and A. Liotta. An online learning approach to a multi-player n-armed functional bandit. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11974 LNCS, 2020.

[133] S. O'Neill, P. Wrigley, and O. Bagdasar. A mixed-integer linear programming formulation for the modular layout of three-dimensional connected systems. *Mathematics and Computers in Simulation*, 2021.

[134] C. H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and system Sciences*, 48(3):498–532, 1994.

[135] M. Patriksson. *The Traffic Assignment Problem: Models and Methods*. Dover Publications, 1994.

[136] O. Perederieieva, M. Ehrgott, J. Y. T. Wang, and A. Raith. A computational study of traffic assignment algorithms. In *Australasian Transport Research Forum*, Brisbane, Australia, 2013.

[137] PTV Group. PTV Vissim, 2021.

[138] P. Qiang and A. Nagurney. A bi-criteria indicator to assess supply chain network performance for critical needs under capacity and demand disruptions. *Transportation Research Part A: Policy and Practice*, 46(5):801–812, 1 2012.

[139] Q. Qiang and A. Nagurney. A unified network performance measure with importance identification and the ranking of network components. *Optimization Letters*, 2(1):127–142, 2008.

[140] A. Raith, J. Y. T. Wang, M. Ehrgott, and S. A. Mitchell. Solving multi-objective traffic assignment. *Annals of Operations Research*, 222(1):483–516, 3 2014.

[141] R. W. Rosenthal. A class of games possessing pure-strategy Nash equilibria. *International Journal of Game Theory*, 2(1):65–67, 1973.

[142] R. W. Rosenthal. The network equilibrium problem in integers. *Networks*, 3(1):53–59, 1973.

[143] A. E. Roth and A. Ockenfels. Last-minute bidding and the rules for ending second-price auctions: Evidence from eBay and Amazon auctions on the Internet. *American Economic Review*, 92(4):1093–1103, 9 2002.

[144] T. Roughgarden. *Selfish Routing and the Price of Anarchy*. MIT Press, 2005.

[145] T. Roughgarden. On the severity of Braess's paradox: Designing networks for selfish users is hard. *Journal of Computer and System Sciences*, 72(5):922–953, 2006.

[146] T. Roughgarden. Routing Games. In N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, editors, *Algorithmic Game Theory*, pages 461–486. Cambridge University Press, Cambridge, 2007.

[147] T. Roughgarden and É. Tardos. How bad is selfish routing? *Journal of the ACM*, 49(2):236–259, 3 2002.

[148] W. H. Sandholm. Potential games with continuous player sets. *Journal of Economic theory*, 97(1):81–108, 2001.

[149] A. Saxena and S. Iyengar. Centrality measures in complex networks: A survey. *CoRR*, abs/2011.07190, 2020.

[150] A. Schneck and K. Nökel. Accelerating traffic assignment with customizable contraction hierarchies. *Transportation Research Record*, 2674(1):188–196, 2020.

[151] M. Schneider. Access and land development. *Highway Research Board Special Report*, (97):164–177, 1968.

[152] Y. Sheffi. *Urban Transportation Networks: Equilibrium Analysis With Mathematical Programming Methods*. Prentice Hall, 1984.

[153] B. Skinner. The price of anarchy in basketball. *Journal of Quantitative Analysis in Sports*, 6(1), 2010.

[154] H. Slavin, J. Brandon, and A. Rabinowicz. An empirical comparison of alternative user equilibrium traffic assignment methods. In *Proceedings of the European Transport Conference*, Strasbourg, France, 2006.

[155] N. Sobrino, A. Monzón, and S. Hernandez. Reduced Carbon and Energy Footprint in Highway Operations: The Highway Energy Assessment (HERA) Methodology. *Netw Spat Econ*, 16:1–20, 2 2014.

[156] C. L. Staudt, A. Sazonovs, and H. Meyerhenke. NetworKit: A tool suite for large-scale complex network analysis. *Network Science*, 4(4):508–530, 2016.

[157] C. Sun, L. Cheng, S. Zhu, F. Han, and Z. Chu. Multi-criteria user equilibrium model considering travel time, travel time reliability and distance. *Transportation Research Part D: Transport and Environment*, 66:3–12, 2019.

[158] W. Y. Szeto, X. Jaber, and S. C. Wong. Road network equilibrium approaches to environmental sustainability. *Transport Reviews*, 32(4):491–518, 2012.

[159] The pandas development team. pandas-dev/pandas: Pandas 1.3.4, 2021.

[160] Tiago P. Peixoto. Performance comparison - graph-tool: Efficient network analysis with python, 2020.

[161] Transportation Networks for Research Core Team. Transportation Networks for Research.

[162] United States. Bureau of Public Roads. *Traffic Assignment Manual for Application with a Large, High Speed Computer*. US Department of Commerce, Bureau of Public Roads, Office of Planning, 1964.

[163] N. Vafaei, R. A. Ribeiro, and L. M. Camarinha-Matos. Selection of normalization technique for weighted average multi-criteria decision making. In *Doctoral Conference on Computing, Electrical and Industrial Systems*, volume 521, pages 43–52. Springer, 2018.

[164] J. Von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1944.

[165] G. M. Wang, Z. Y. Gao, and M. Xu. An MPEC formulation and its cutting constraint algorithm for continuous network design problem with multi-user classes. *Applied Mathematical Modelling*, 38(5-6):1846–1858, 2014.

[166] Y. Wang and K. Cullinane. Traffic consolidation in East Asian container ports: A network flow analysis. *Transportation Research Part A: Policy and Practice*, 61:152–163, 2014.

[167] J. G. Wardrop. Road paper. Some theoretical aspects of road traffic research. *Proceedings of the Institution of Civil Engineers*, 1(3):325–362, 1952.

[168] M. L. Waskom. seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60):3021, 4 2021.

[169] H. P. Williams. *Model Building in Mathematical Programming*. John Wiley & Sons, 2013.

[170] M. Xu. Understanding graph embedding methods and their applications. *CoRR*, abs/2012.08019, 2020.

[171] H. Yang and H. J. Huang. The multi-class, multi-criteria traffic network equilibrium and systems optimum problem. *Transportation Research Part B: Methodological*, 38(1):1–15, 2004.

[172] P. L. Yu. A class of solutions for group decision problems. *Management Science*, 19(8):936–946, 1973.

[173] P.-L. Yu. *Multiple-Criteria Decision Making.* Springer US, 1985.

[174] D. Zhu, Y. Hu, Y. Li, and B. Yu. A new measure for airline networks performance evaluation and critical cities identification. *Fudan University, Shanghai, China*, 2006.

Appendix A

Appendix: Publications

A.1 Traffic assignment: Methods and simulations for an alternative formulation of the fixed demand problem

Removed for copyright

A.2 An online learning approach to a multi-player n-armed functional bandit

Removed for copyright

A.3 Modelling equilibrium for a multi-criteria selfish routing network equilibrium flow problem

Removed for copyright

A.4 A mixed-integer linear programming formulation for the modular layout of three-dimensional connected systems

Removed for copyright

A.5 The traffic assignment problem - Mathematics Today

Removed for copyright

Appendix B

Appendix: Conference Papers

B.1 Traffic modelling: A network analysis of road networks of major cities and towns for England and Wales

Removed for copyright

B.2 Traffic assignment problem: An overview of link, bush and route based solution methods

Removed for copyright

B.3 An online learning approach to a multi-player n-armed functional bandit

Removed for copyright

B.4 A mixed-integer linear programming formulation for the modularisation of 3-dimensional connected systems

Removed for copyright

B.5 Modelling of equilibrium for a multi-objective network equilibrium flow problem

Removed for copyright

Appendix C

Appendix: Networks

This appendix details networks used in this thesis, comprising of the network diagram, link function table and origin-destination trip table. All data is presented as was used in the code and generation of results for this thesis. All data can be found at [129].

C.1 Braess Network

C.1.1 Network Diagram

Note that edge $(2, 3)$ is the optional edge which is added to demonstrate the Braess Paradox.

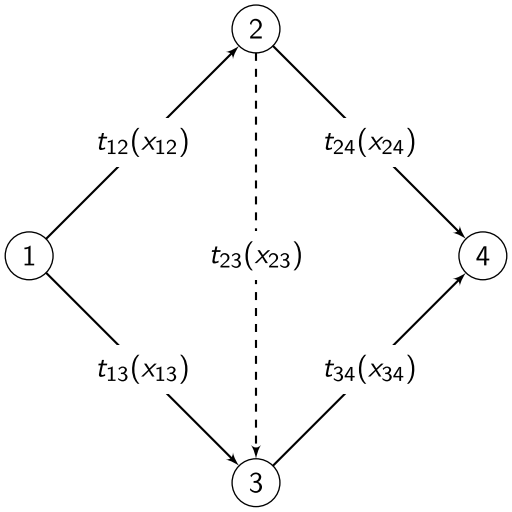


Figure C.1: Braess network diagram.

C.1.2 Link Functions

For a given edge $e = (i, j)$, link functions are of the form,

$$t_{ij}(x_{ij}) = a_{ij} + b_{ij} \left(\frac{x_{ij}}{c_{ij}} \right)^{n_{ij}}.$$

Note that edge $(2, 3)$ in table C.1 simplifies to $t_{23}(x_{23}) = 0$.

i	j	a_{ij}	b_{ij}	c_{ij}	n_{ij}
1	2	0	1	100	1
1	3	45	0	100	1
2	3	0	0	100	1
2	4	45	0	100	1
3	4	0	1	100	1

Table C.1: Link functions for Braess network.

C.1.3 Origin-destination Table

	1	**2**	**3**	**4**
1	0	0	0	4000
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0

Table C.2: Origin-destination trip table for Braess network.

C.2 Qiang and Nagurney Network

C.2.1 Network Diagram

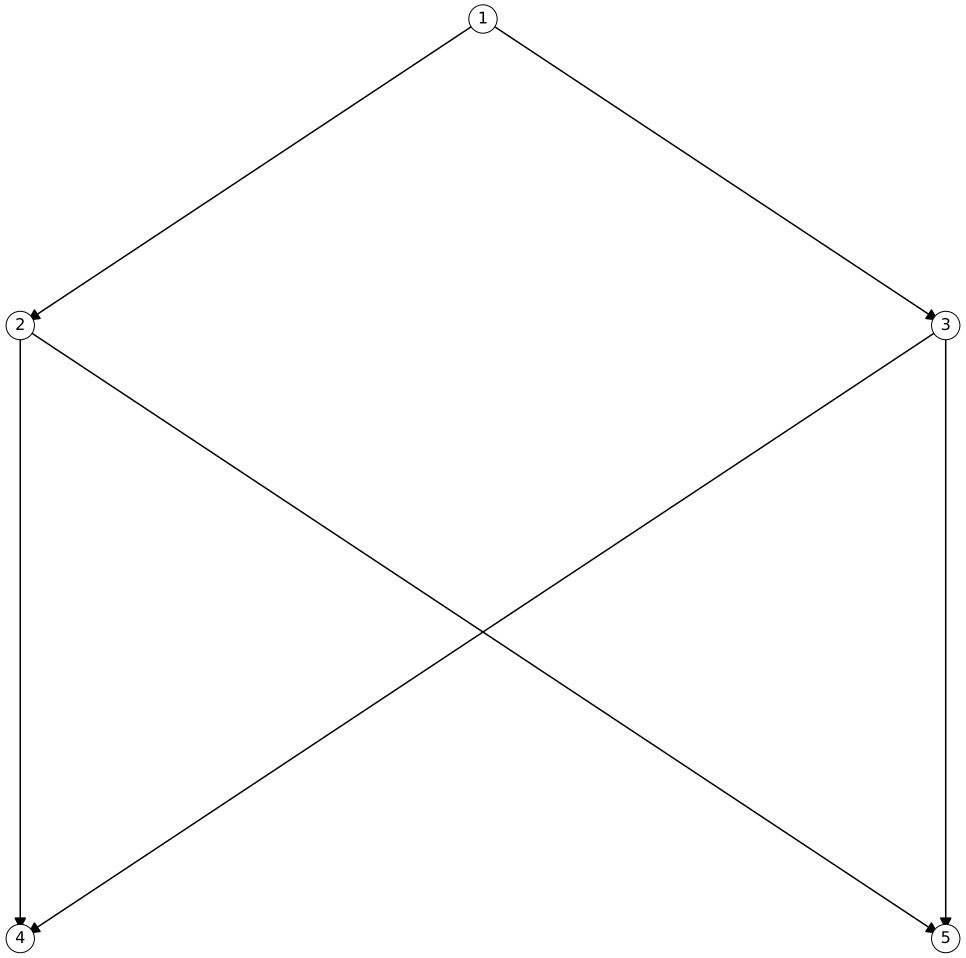


Figure C.2: Qiang and Nagurney network diagram.

C.2.2 Link Functions

For a given edge $e = (i, j)$, link functions are of the form,

$$t_{ij}(x_{ij}) = a_{ij} + b_{ij}x_{ij} + c_{ij}x_{ij}^{n_{ij}}.$$

For this network the link functions are simply,

$$t_{ij}(x_{ij}) = x_{ij}$$

i	j	a_{ij}	b_{ij}	c_{ij}	n_{ij}
1	2	0	1	0	1
1	3	0	1	0	1
2	5	0	1	0	1
3	4	0	1	0	1
2	4	0	1	0	1
3	5	0	1	0	1

Table C.3: Link functions for Qiang and Nagurney network.

C.2.3 Origin-destination Table

	1	**2**	**3**	**4**	**5**
1	0	0	0	100	20
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0

Table C.4: Origin-destination trip table for Qiang and Nagurney network.

C.3 Sioux Falls Network

C.3.1 Network Diagram

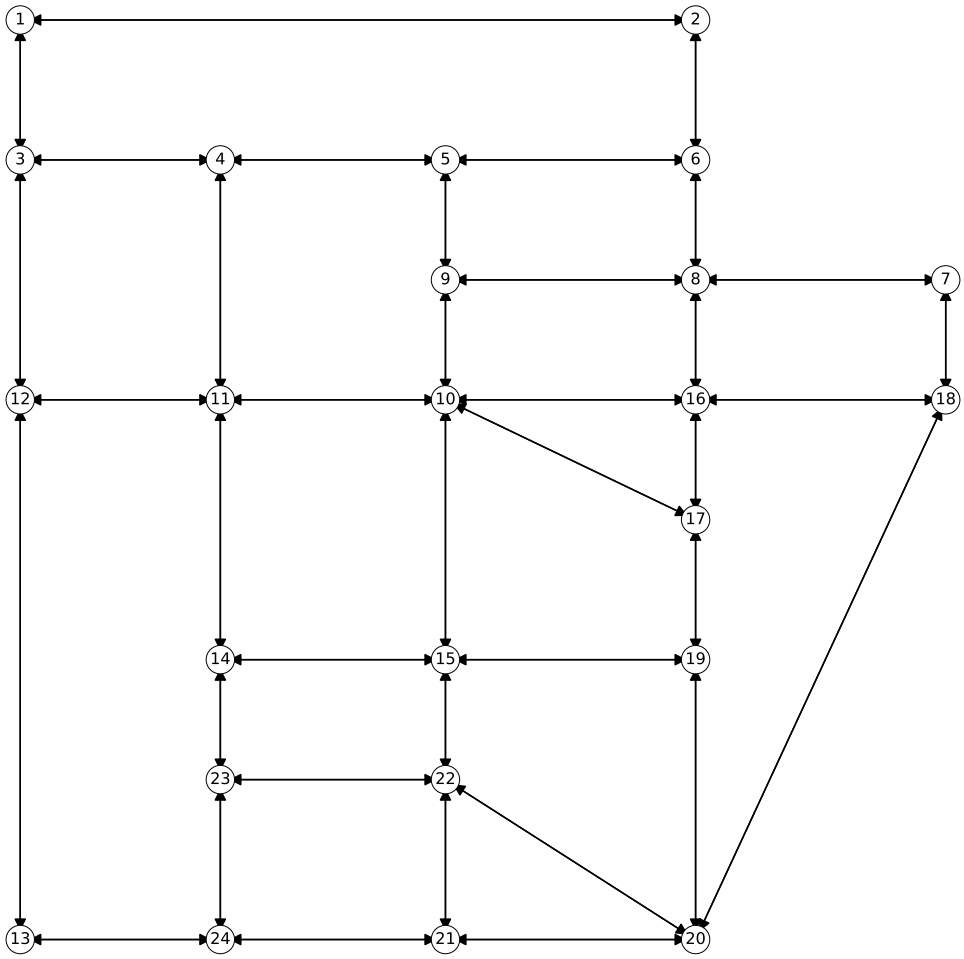


Figure C.3: Sioux Falls network diagram.

C.3.2 Link Functions

For a given edge $e = (i, j)$, link functions are of the form

$$t_{ij}(x_{ij}) = a_{ij}\left(1 + b_{ij}\left(\frac{x_{ij}}{c_{ij}}\right)^{n_{ij}}\right).$$

i	j	a_{ij}	b_{ij}	c_{ij}	n_{ij}
1	2	6	0.15	25900.20064	4
1	3	4	0.15	23403.47319	4
2	1	6	0.15	25900.20064	4
2	6	5	0.15	4958.180928	4
3	1	4	0.15	23403.47319	4
3	4	4	0.15	17110.52372	4
3	12	4	0.15	23403.47319	4
4	3	4	0.15	17110.52372	4
4	5	2	0.15	17782.7941	4
4	11	6	0.15	4908.82673	4
5	4	2	0.15	17782.7941	4
5	6	4	0.15	4947.995469	4
5	9	5	0.15	10000	4
6	2	5	0.15	4958.180928	4
6	5	4	0.15	4947.995469	4
6	8	2	0.15	4898.587646	4
7	8	3	0.15	7841.81131	4
7	18	2	0.15	23403.47319	4
8	6	2	0.15	4898.587646	4
8	7	3	0.15	7841.81131	4
8	9	10	0.15	5050.193156	4
8	16	5	0.15	5045.822583	4
9	5	5	0.15	10000	4
9	8	10	0.15	5050.193156	4
9	10	3	0.15	13915.78842	4
10	9	3	0.15	13915.78842	4
10	11	5	0.15	10000	4
10	15	6	0.15	13512.00155	4
10	16	4	0.15	4854.917717	4
10	17	8	0.15	4993.510694	4
11	4	6	0.15	4908.82673	4
11	10	5	0.15	10000	4
11	12	6	0.15	4908.82673	4
11	14	4	0.15	4876.508287	4
12	3	4	0.15	23403.47319	4
12	11	6	0.15	4908.82673	4
12	13	3	0.15	25900.20064	4
13	12	3	0.15	25900.20064	4
13	24	4	0.15	5091.256152	4
14	11	4	0.15	4876.508287	4
14	15	5	0.15	5127.526119	4
14	23	4	0.15	4924.790605	4
15	10	6	0.15	13512.00155	4

i	j	a_{ij}	b_{ij}	c_{ij}	n_{ij}
15	14	5	0.15	5127.526119	4
15	19	3	0.15	14564.75315	4
15	22	3	0.15	9599.180565	4
16	8	5	0.15	5045.822583	4
16	10	4	0.15	4854.917717	4
16	17	2	0.15	5229.910063	4
16	18	3	0.15	19679.89671	4
17	10	8	0.15	4993.510694	4
17	16	2	0.15	5229.910063	4
17	19	2	0.15	4823.950831	4
18	7	2	0.15	23403.47319	4
18	16	3	0.15	19679.89671	4
18	20	4	0.15	23403.47319	4
19	15	3	0.15	14564.75315	4
19	17	2	0.15	4823.950831	4
19	20	4	0.15	5002.607563	4
20	18	4	0.15	23403.47319	4
20	19	4	0.15	5002.607563	4
20	21	6	0.15	5059.91234	4
20	22	5	0.15	5075.697193	4
21	20	6	0.15	5059.91234	4
21	22	2	0.15	5229.910063	4
21	24	3	0.15	4885.357564	4
22	15	3	0.15	9599.180565	4
22	20	5	0.15	5075.697193	4
22	21	2	0.15	5229.910063	4
22	23	4	0.15	5000	4
23	14	4	0.15	4924.790605	4
23	22	4	0.15	5000	4
23	24	2	0.15	5078.508436	4
24	13	4	0.15	5091.256152	4
24	21	3	0.15	4885.357564	4
24	23	2	0.15	5078.508436	4

Table C.5: Link functions for Sioux Falls network.

C.3.3 Origin-destination Table

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	0	100	100	500	200	300	500	800	500	1300	500	200	500	300	500	500	400	100	300	300	100	400	300	100
2	100	0	100	200	100	400	200	400	200	600	200	100	300	100	100	400	200	0	100	100	0	100	0	0
3	100	100	0	200	100	300	100	200	100	300	300	200	100	100	100	200	100	0	0	0	0	100	100	0
4	500	200	200	0	500	400	400	700	700	1200	1400	600	600	500	500	800	500	100	200	300	200	400	500	200
5	200	100	100	500	0	200	200	500	800	1000	500	200	200	100	200	500	200	0	100	100	100	200	100	0
6	300	400	300	400	200	0	400	800	400	800	400	200	200	100	200	900	500	100	200	300	100	200	100	100
7	500	200	100	400	200	400	0	1000	600	1900	500	700	400	200	500	1400	1000	200	400	500	200	500	200	100
8	800	400	200	700	500	800	1000	0	800	1600	800	600	600	400	600	2200	1400	300	700	900	400	500	300	200
9	500	200	100	700	800	400	600	800	0	2800	1400	600	600	600	900	1400	900	200	400	600	300	700	500	200
10	1300	600	300	1200	1000	800	1900	1600	2800	0	4000	2000	1900	2100	4000	4400	3900	700	1800	2500	1200	2600	1800	800
11	500	200	300	1500	500	400	500	800	1400	3900	0	1400	1000	1600	1400	1400	1000	100	400	600	400	1100	1300	600
12	200	100	200	600	200	200	700	600	600	2000	1400	0	1300	700	700	700	600	200	300	400	300	700	700	500
13	500	300	100	600	200	200	400	600	600	1900	1000	1300	0	600	700	600	500	100	300	600	600	1300	800	800
14	300	100	100	500	100	100	200	400	600	2100	1600	700	600	0	1300	700	700	100	300	500	400	1200	1100	400
15	500	100	100	500	200	200	500	600	1000	4000	1400	700	700	1300	0	1200	1500	200	800	1100	800	2600	1000	400
16	500	400	200	800	500	900	1400	2200	1400	4400	1400	700	600	700	1200	0	2800	500	1300	1600	600	1200	500	300
17	400	200	100	500	200	500	1000	1400	900	3900	1000	600	500	700	1500	2800	0	600	1700	1700	600	1700	600	300
18	100	0	0	100	0	100	200	300	200	700	200	200	100	100	200	500	600	0	300	400	100	300	100	0
19	300	100	0	200	100	200	400	700	400	1800	400	300	300	300	800	1300	1700	300	0	1200	400	1200	300	100
20	300	100	0	300	100	300	500	900	600	2500	600	500	600	500	1100	1600	1700	400	1200	0	1200	2400	700	400
21	100	0	0	200	100	100	200	400	300	1200	400	300	600	400	800	600	600	100	400	1200	0	1800	700	500
22	400	100	100	400	200	200	500	500	700	2600	1100	700	1300	1200	2600	1200	1700	300	1200	2400	1800	0	2100	1100
23	300	0	100	500	100	100	200	300	500	1800	1300	700	800	1100	1000	500	600	100	300	700	700	2100	0	700
24	100	0	0	200	0	100	100	200	200	800	600	500	700	400	400	300	300	0	100	400	500	1100	700	0

Table C.6: Origin-destination trip table for Sioux Falls network.

C.4 Dial's Network

C.4.1 Network Diagram

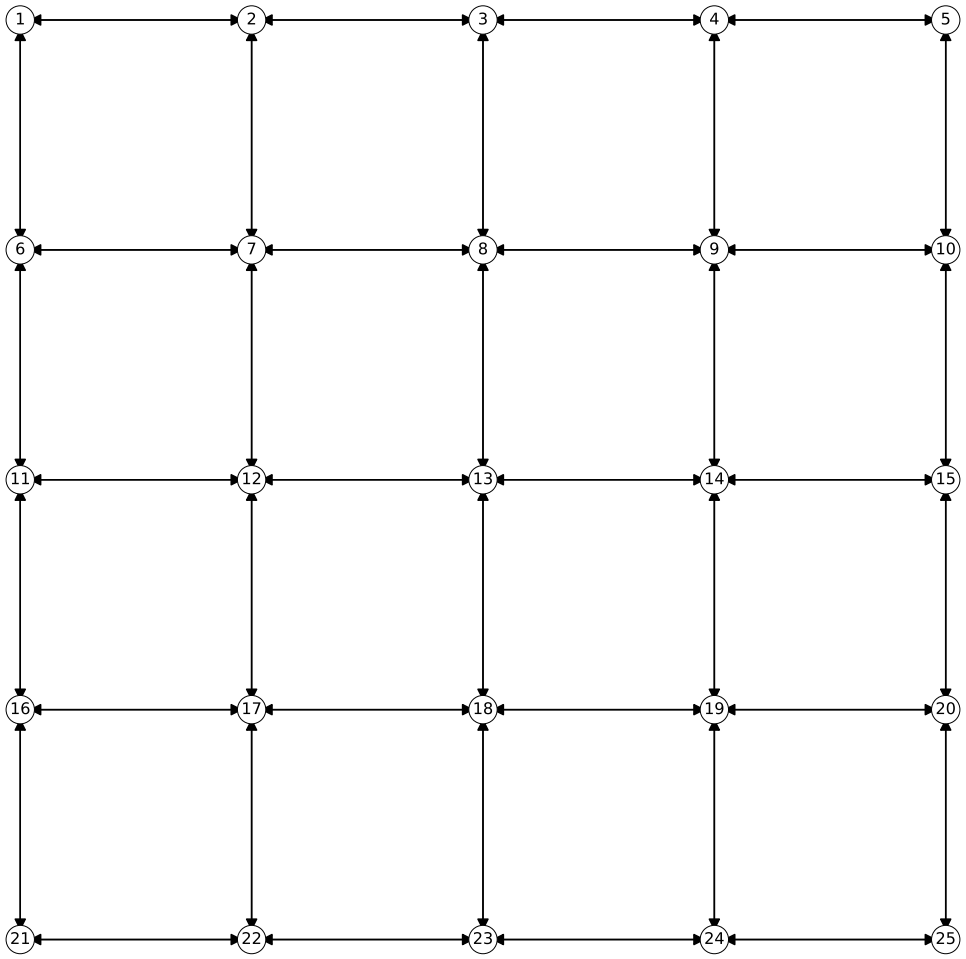


Figure C.4: Dial's network diagram.

C.4.2 Link functions

For a given edge $e = (i, j)$, link functions are of the form

$$t_{ij}(x_{ij}) = a_{ij}\left(1 + b_{ij}\left(\frac{x_{ij}}{c_{ij}}\right)^{n_{ij}}\right).$$

i	j	a_{ij}	b_{ij}	c_{ij}	n_{ij}
1	2	3.72	0.15	300	4
2	1	3.72	0.15	300	4
2	3	5.4	0.15	300	4
3	2	5.4	0.15	300	4
3	4	3.3	0.15	300	4
4	3	3.3	0.15	300	4
4	5	4.98	0.15	300	4
5	4	4.98	0.15	300	4
6	7	4.17	0.15	300	4
7	6	4.17	0.15	300	4
7	8	4.02	0.15	300	4
8	7	4.02	0.15	300	4
8	9	5.55	0.15	300	4
9	8	5.55	0.15	300	4
9	10	4.53	0.15	300	4
10	9	4.53	0.15	300	4
11	12	1.23	0.15	200	4
12	11	1.23	0.15	200	4
12	13	1.85	0.15	200	4
13	12	1.85	0.15	200	4
13	14	1.89	0.15	200	4
14	13	1.89	0.15	200	4
14	15	1.03	0.15	200	4
15	14	1.03	0.15	200	4
16	17	3.09	0.15	300	4
17	16	3.09	0.15	300	4
17	18	4.5	0.15	300	4
18	17	4.5	0.15	300	4
18	19	5.94	0.15	300	4
19	18	5.94	0.15	300	4
19	20	2.97	0.15	300	4
20	19	2.97	0.15	300	4
21	22	5.76	0.15	300	4
22	21	5.76	0.15	300	4
22	23	4.44	0.15	300	4
23	22	4.44	0.15	300	4
23	24	5.1	0.15	300	4
24	23	5.1	0.15	300	4
24	25	3	0.15	300	4
25	24	3	0.15	300	4
1	6	3.72	0.15	300	4
6	1	3.72	0.15	300	4
2	7	4.02	0.15	300	4
7	2	4.02	0.15	300	4
3	8	1.37	0.15	200	4
8	3	1.37	0.15	200	4
4	9	3.09	0.15	300	4

i	j	a_{ij}	b_{ij}	c_{ij}	n_{ij}
4	9	3.09	0.15	300	4
9	4	3.09	0.15	300	4
5	10	4.59	0.15	300	4
10	5	4.59	0.15	300	4
6	11	4.32	0.15	300	4
11	6	4.32	0.15	300	4
7	12	4.11	0.15	300	4
12	7	4.11	0.15	300	4
8	13	1.41	0.15	200	4
13	8	1.41	0.15	200	4
9	14	4.11	0.15	300	4
14	9	4.11	0.15	300	4
10	15	3.42	0.15	300	4
15	10	3.42	0.15	300	4
11	16	4.35	0.15	300	4
16	11	4.35	0.15	300	4
12	17	5.49	0.15	300	4
17	12	5.49	0.15	300	4
13	18	2.15	0.15	200	4
18	13	2.15	0.15	200	4
14	19	5.46	0.15	300	4
19	14	5.46	0.15	300	4
15	20	6.09	0.15	300	4
20	15	6.09	0.15	300	4
16	21	5.58	0.15	300	4
21	16	5.58	0.15	300	4
17	22	4.02	0.15	300	4
22	17	4.02	0.15	300	4
18	23	1.79	0.15	200	4
23	18	1.79	0.15	200	4
19	24	5.85	0.15	300	4
24	19	5.85	0.15	300	4
20	25	4.05	0.15	300	4
25	20	4.05	0.15	300	4

Table C.7: Link functions for Dial's network.

C.4.3 Origin-destination Table

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	500	0	0	0	0	0	0	0	500	0	500	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	500	0	0	0	0	0	0	0	0	0	500	0	500	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	500	0	500	0	0	0	0	0	0	0	0	0	500	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	500	0	500	0	0	0	0	0	0	0	500	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table C.8: Origin-destination trip table for Dial's network.

C.5 Grid 5x5 Network

C.5.1 Network Diagram

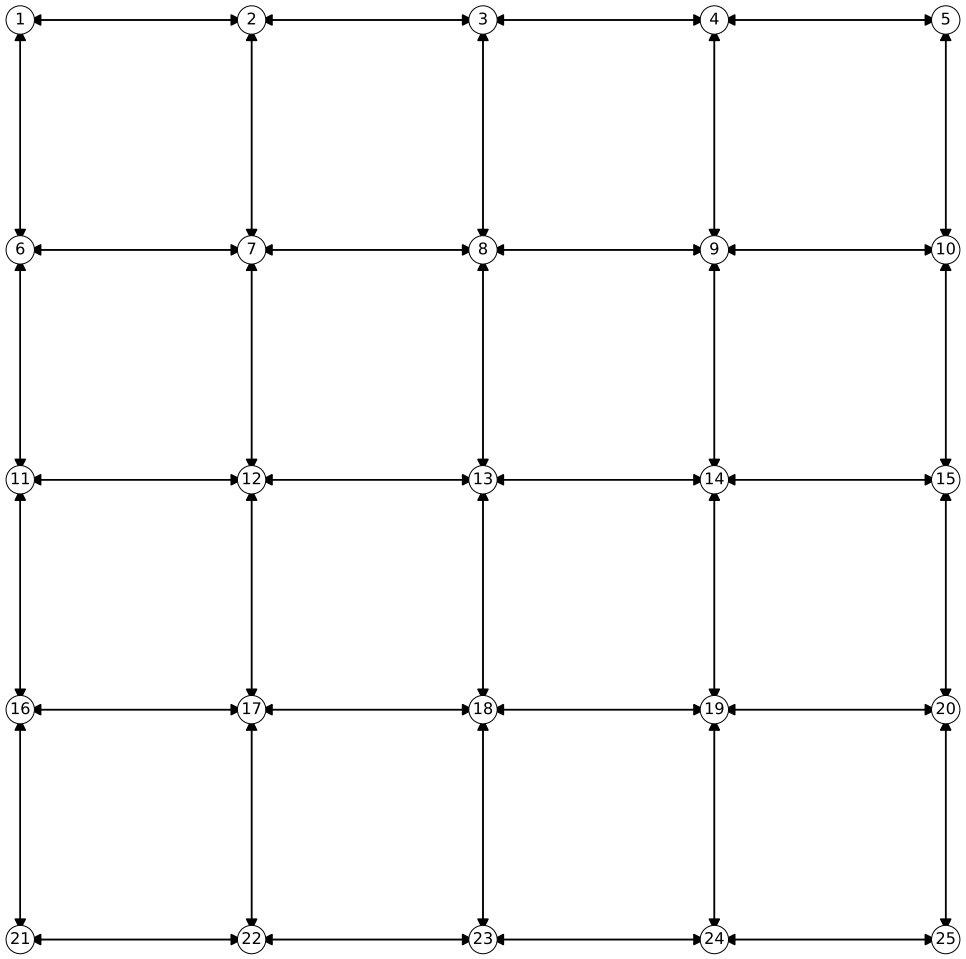


Figure C.5: Grid 5x5 network diagram.

C.5.2 Link functions

For a given edge $e = (i, j)$, link functions are of the form

$$t_{ij}(x_{ij}) = a_{ij} + b_{ij} \left(\frac{x_{ij}}{c_{ij}} \right)^{n_{ij}}.$$

i	j	a_{ij}	b_{ij}	c_{ij}	n_{ij}
1	2	2	0.5	1	2
2	1	2	0.5	1	2
2	3	2	0.5	1	2
3	2	2	0.5	1	2
3	4	2	0.5	1	2
4	3	2	0.5	1	2
4	5	2	0.5	1	2
5	4	2	0.5	1	2
6	7	2	0.5	1	2
7	6	2	0.5	1	2
7	8	2	0.5	1	2
8	7	2	0.5	1	2
8	9	2	0.5	1	2
9	8	2	0.5	1	2
9	10	2	0.5	1	2
10	9	2	0.5	1	2
11	12	2	0.5	1	2
12	11	2	0.5	1	2
12	13	2	0.5	1	2
13	12	2	0.5	1	2
13	14	2	0.5	1	2
14	13	2	0.5	1	2
14	15	2	0.5	1	2
15	14	2	0.5	1	2
16	17	2	0.5	1	2
17	16	2	0.5	1	2
17	18	2	0.5	1	2
18	17	2	0.5	1	2
18	19	2	0.5	1	2
19	18	2	0.5	1	2
19	20	2	0.5	1	2
20	19	2	0.5	1	2
21	22	2	0.5	1	2
22	21	2	0.5	1	2
22	23	2	0.5	1	2
23	22	2	0.5	1	2
23	24	2	0.5	1	2
24	23	2	0.5	1	2
24	25	2	0.5	1	2
25	24	2	0.5	1	2
1	6	2	0.5	1	2
6	1	2	0.5	1	2
2	7	2	0.5	1	2
7	2	2	0.5	1	2
3	8	2	0.5	1	2
8	3	2	0.5	1	2
4	9	2	0.5	1	2

i	j	a_{ij}	b_{ij}	c_{ij}	n_{ij}
9	4	2	0.5	1	2
5	10	2	0.5	1	2
10	5	2	0.5	1	2
6	11	2	0.5	1	2
11	6	2	0.5	1	2
7	12	2	0.5	1	2
12	7	2	0.5	1	2
8	13	2	0.5	1	2
13	8	2	0.5	1	2
9	14	2	0.5	1	2
14	9	2	0.5	1	2
10	15	2	0.5	1	2
15	10	2	0.5	1	2
11	16	2	0.5	1	2
16	11	2	0.5	1	2
12	17	2	0.5	1	2
17	12	2	0.5	1	2
13	18	2	0.5	1	2
18	13	2	0.5	1	2
14	19	2	0.5	1	2
19	14	2	0.5	1	2
15	20	2	0.5	1	2
20	15	2	0.5	1	2
16	21	2	0.5	1	2
21	16	2	0.5	1	2
17	22	2	0.5	1	2
22	17	2	0.5	1	2
18	23	2	0.5	1	2
23	18	2	0.5	1	2
19	24	2	0.5	1	2
24	19	2	0.5	1	2
20	25	2	0.5	1	2
25	20	2	0.5	1	2

Table C.9: Link functions for grid 5x5 network.

C.5.3 Origin-destination Table

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table C.10: Origin-destination trip table for grid 5x5 network.

C.5.4 Origin-destination Reduced Demand Table

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table C.11: Origin-destination reduced trip table for grid 5x5 network.

C.5.5 Origin-destination Increased Demand Table

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1000	0	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
23	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table C.12: Origin-destination increased trip table for grid 5x5 network.