

# Customer mobility and congestion in supermarkets



Fabian Ying  
Pembroke College  
University of Oxford

A thesis submitted for the degree of  
*Doctor of Philosophy*

Trinity 2019

## Acknowledgements

This work would not be possible without the invaluable support and guidance of my academic supervisors Mariano Beguerisse-Díaz, Sam D. Howison, and Mason A. Porter. I am immensely grateful for their insight, guidance, and teaching, which helped me learn how to conduct research and ask the right questions. I greatly benefited from their complementary perspectives and wisdom. I want to particularly thank Mason who inspired me to pursue a DPhil through the summer research project back in 2013.

This work is supported by the EPSRC Centre For Doctoral Training in Industrially Focused Mathematical Modelling (EP/L015803/1) in collaboration with Tesco. I am grateful to my industrial supervisors Alisdair Wallis and (previously) Jeremy Bradley at Tesco as well as the Tesco Data Science team (particularly Trevor Sidery and Robert Armstrong) for their support and for so warmly hosting me during my visits at the Tesco offices.

I am thankful to Chris Breward and Colin Please, as well as to Laura O'Mahony, Amanda Guthland Elmore, Sarah Howle, and Jonathan Mason for their hard work in running the InFoMM CDT, which has enriched me in many ways.

I am grateful to my peers and friends at the Mathematical Institute, who have made it an absolute joy to work there and helped and encouraged me in this journey, particularly my cohort at InFoMM (including Lindon Roberts, Bogdan Toader, Thomas Roy, Michael McPhail), the networks research group (including Florian Klimm, Andrew Mellor, Alice Schwarze, David O'Sullivan, Sewook Oh), the Maths & Stats Christian prayer group (including Ben Sloman and Sam Cohen), and my office mates (Roxana Pamfil, Rodrigo Leal Cerventes, and Karel Devriendt). I also want to thank Naoki Masuda, Chico Camargo, Scott Hale, and my examiners Renaud Lambiotte and Albert Díaz-Guilera for fruitful discussions that helped improve my work.

I want to thank the Oxford church ministry, particularly Mark and Rachel Abril for their friendship and encouragement (both in terms of food and lessons).

I am grateful to my parents and my brother, Alex: to my mom for always looking out for me; to my dad for instilling the love in maths in me; and to my brother, for being a great friend and source of encouragement in my life.

I am immensely grateful to my wife, Gene, for believing in me so much more than I do and for being a constant source of encouragement, support, and love throughout my DPhil.

Finally, I want to give thanks to the almighty and loving God; only through his grace this work was possible.

# Abstract

We model customer mobility and congestion in supermarkets and investigate how these two processes depend on the layout of a store. Our motivation is twofold: we seek to understand how customers move in supermarkets and how to arrange the layout of a supermarket to reduce congestion in it. Congestion affects both the shopping experience and the bottom line of supermarkets, so models that can estimate customer mobility and congestion in new store layouts are of great interest to retailers.

We use random-walk models and population-level mobility models to model customer mobility, and we use queues and queueing networks to model congestion. We represent a store as a network in which the nodes are zones in a supermarket and edges connect adjacent zones. Customers traverse the network from node to node and queue at each node. We measure congestion by the total mean queue size  $Q$  (which is equivalent to the total number of customers in a store). We are interested in how network topology (representing store layout) affects  $Q$ .

We first examine a simple model of single-source, single-sink queueing networks with unbiased random walkers. We analyse the relationship between congestion (specifically,  $Q$ ) and network topology, and we describe network topologies that minimize  $Q$ . We also present efficient greedy algorithms for edge addition and deletion to reduce  $Q$ . We then consider human-mobility models and use them to estimate mobility flows between zones in a supermarket. These models have, to our knowledge, not been applied previously to problems with such small spatial scales. We applied the human-mobility models to 17 supermarkets, and we find that they can successfully estimate 65–70% of the mobility flow in a supermarket. We also find that the parameters in our models do not change markedly between different stores and different time periods of the same store. One can therefore calibrate the parameters of our models on one store and then use the models to estimate mobility flow in all other stores using only purchase data and the store layouts. Finally, we integrate the human-mobility models into a queueing-network framework to estimate congestion in supermarkets. We present a simple optimization algorithm that swaps the locations of aisles and finds store layouts with significantly smaller values of  $Q$ . In these store layouts, popular nodes are often moved from the centre of a store to the perimeter.



Our research helps improve understanding of customer mobility and congestion in supermarkets, especially the relationship between congestion and store layout. Our models and ideas are also relevant for complex systems on similar spatial scales (e.g., mobility of visitors in a museum).

## Papers

At the time of submission of this thesis, one paper (based on the work from Chapters 5 and 6) has been submitted for publication and two papers are in preparation (one based on the work from Chapter 4 and one short review based on our framework in Section 2.4 for applying human-mobility models):

- F. Ying, A. O. G. Wallis, M. Beguerisse-Díaz, M. A. Porter, S. D. Howison. “Customer mobility and congestion in supermarkets.” (2019), [arXiv:1905.13098](#). (Submitted to *Physical Review E*)
- F. Ying, A. O. G. Wallis, M. A. Porter, S. D. Howison, M. Beguerisse-Díaz. “Minimizing total mean queue size in single-source, single-sink queueing networks” (2019), in preparation. (To be submitted to *SIAM Journal on Applied Mathematics*)
- F. Ying, M. A. Porter, S. D. Howison, M. Beguerisse-Díaz. “Framework for applying population-level human-mobility models” (2019), in preparation. (To be submitted to *Transactions of Mathematics and its Applications: A Journal of the IMA*)

## **Statement of Originality**

The research in this thesis is a result of collaboration between myself and my coauthors on the listed papers. My collaborators have helped develop the ideas described in this thesis, but I have performed all of the analysis leading to the results that I present.

# Contents

<b>List of Symbols</b>	<b>xii</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 Networks, queues, mobility, and congestion modelling</b>	<b>5</b>
2.1 Networks . . . . .	5
2.1.1 Regular, complete, spatial, and planar graphs . . . . .	10
2.1.2 Random walks on graphs . . . . .	11
2.1.3 Random-graph models . . . . .	11
2.2 Traffic on networks . . . . .	16
2.2.1 Traffic capacity . . . . .	17
2.2.2 Strategies to increase the traffic capacity . . . . .	20
2.3 Queueing networks . . . . .	20
2.3.1 Open queueing networks . . . . .	22
2.3.2 Little’s Law . . . . .	26
2.3.3 Multi-class open queueing networks . . . . .	27
2.3.4 Comparison between the traffic-dynamics model and the open-queueing-network model . . . . .	27
2.4 Human-mobility models . . . . .	28
2.4.1 Gravity models . . . . .	30
2.4.2 Intervening-opportunities models . . . . .	30
2.4.3 Constrained versions of mobility models . . . . .	34
2.4.4 Goodness-of-fit measures . . . . .	37
2.4.5 Parameter calibration . . . . .	41
2.4.6 Framework for applying mobility models . . . . .	42
2.5 Customer mobility in supermarkets . . . . .	42
2.5.1 Clustering of customer shopping journeys . . . . .	42
2.5.2 Empirical studies . . . . .	43

2.5.3	Models of customer behaviour . . . . .	44
2.5.4	Summary . . . . .	46
<b>3</b>	<b>Representing stores as networks</b>	<b>47</b>
3.1	Data . . . . .	47
3.2	Creating the store network . . . . .	48
3.2.1	Creating the nodes . . . . .	50
3.2.2	Creating the edges . . . . .	52
3.3	Assigning shelves to zones . . . . .	53
3.4	Properties of the store networks . . . . .	53
<b>4</b>	<b>Minimizing congestion in queueing networks</b>	<b>56</b>
4.1	Single-source, single-sink open queueing network with unbiased random walkers . . . . .	57
4.1.1	Traffic equations . . . . .	58
4.1.2	Total mean queue size $Q$ . . . . .	60
4.1.3	Summary of our model . . . . .	60
4.1.4	Undirected single-source, single-sink open queueing network . . . . .	61
4.2	Effect of network topology on total mean queue size $Q$ . . . . .	62
4.3	Network topologies that minimize total mean queue size $Q$ . . . . .	64
4.3.1	Proof of Theorem 1 . . . . .	66
4.3.2	Properties of $Q$ -optimal networks . . . . .	69
4.3.3	Extensions . . . . .	72
4.3.4	Numerical evidence for $Q$ -optimality of $G_{\mathcal{C}_n}$ over $\mathcal{C}_n$ . . . . .	74
4.3.5	$Q$ -optimality of undirected networks . . . . .	75
4.4	Reducing $Q$ by adding or deleting edges in undirected networks . . . . .	77
4.4.1	Mathematical set-up . . . . .	78
4.4.2	Greedy algorithms . . . . .	79
4.4.3	Random network topologies for queueing networks . . . . .	85
4.4.4	Scenario 1: Edge deletions only . . . . .	87
4.4.5	Scenario 2: Edge additions only . . . . .	90
4.4.6	Scenario 3: Edge deletions and edge additions . . . . .	95
4.4.7	Applying greedy algorithms to store networks . . . . .	98
4.5	Summary and Discussion . . . . .	102

<b>5</b>	<b>Estimating mobility flow in supermarkets</b>	<b>107</b>
5.1	Mathematical set-up . . . . .	108
5.2	Data . . . . .	109
5.3	Mobility models . . . . .	112
5.3.1	Gravity models . . . . .	116
5.3.2	Intervening-opportunities models . . . . .	116
5.3.3	Benchmark model . . . . .	117
5.3.4	Goodness-of-fit measures . . . . .	118
5.3.5	Parameter calibration . . . . .	119
5.4	Results . . . . .	119
5.4.1	Fit to data . . . . .	119
5.4.2	Sensitivity analysis: Size of the data set . . . . .	126
5.4.3	Sensitivity analysis: Parameter dependence of models . . . . .	128
5.4.4	Evaluation: Model performance on estimating trips in unseen data . . . . .	129
5.5	Incorporating zone-zone correlations . . . . .	131
5.6	Summary and Discussion . . . . .	132
<b>6</b>	<b>Integrating human-mobility models with queueing networks</b>	<b>137</b>
6.1	Congestion model . . . . .	137
6.2	Sales model . . . . .	141
6.3	Mobility model . . . . .	141
6.4	Optimization algorithm . . . . .	142
6.5	Results . . . . .	142
6.6	Summary and Discussion . . . . .	150
<b>7</b>	<b>Conclusions and future work</b>	<b>154</b>
	<b>Appendices</b>	<b>159</b>
<b>A</b>	<b>Store zoning, store networks, and store network properties</b>	<b>160</b>
<b>B</b>	<b>Appendix to Chapter 4</b>	<b>179</b>
B.1	Simulated-annealing algorithm for finding optimal network topologies	179
B.1.1	Description of simulated-annealing algorithm . . . . .	179
B.1.2	Results . . . . .	180
B.2	Maximum arrival rates of $G_{\mathcal{U}_n}^{\lambda_{\max}}$ and $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$ . . . . .	183
B.3	$Q$ -optimal networks over $\mathcal{U}_n$ for $n = 3, 4$ . . . . .	184

B.4	Computation time of greedy algorithms . . . . .	189
B.5	Performance of HARF and HDF with recalculation of edge rankings .	191
B.6	Reducing $\lambda_{\max}$ or $\lambda_{\text{total}}$ by adding or deleting edges . . . . .	193
B.6.1	Reducing $\lambda_{\text{total}}$ . . . . .	193
B.6.2	Reducing $\lambda_{\max}$ . . . . .	199
<b>C</b>	<b>Appendix to Chapter 5</b>	<b>205</b>
C.1	Performance of the doubly-constrained gravity model with an exponential deterrence function . . . . .	205
C.2	Estimating $\{O_k^{\text{data}}\}_{k=1}^n$ and $D_1^{\text{data}}$ from purchase data . . . . .	205
<b>D</b>	<b>Appendix to Chapter 6</b>	<b>208</b>
D.1	Transition matrix of shortest-path walker . . . . .	208
D.2	Arrival rates $\lambda_k$ . . . . .	209
	<b>Bibliography</b>	<b>212</b>

# List of Symbols

$G = (V, E)$	Graph with node set $V$ and edge set $E$
$n$	Number of nodes in a graph $G$
$m$	Number of edges in a graph $G$
$\mathbf{A}$	Adjacency matrix of a graph
$d_i$	Degree of node $i$ (in an undirected graph)
$d_i^{\text{out}}$	Out-degree of node $i$ (in a directed graph)
$d_i^{\text{in}}$	In-degree of node $i$ (in a directed graph)
$\mathbf{L}$	Laplacian matrix
$\mathbf{D}$	Degree matrix
$l_{ij}$	Length of edge $(i, j)$ in $G$
$G[S]$	Induced subgraph of $G = (V, E)$ on $S \subset V$
$C_{\text{in}}(i)$	In-component of node $i$
$C_{\text{out}}(i)$	Out-component of node $i$
$\mathbb{Z}_+$	Set of non-negative integers
$\mathbf{P}$	Transition matrix of a Markov random walk
$\rho$	Packet generation rate in traffic-dynamics model
$\rho_c$	Traffic capacity in traffic-dynamics model
$\mathbf{T}$	Origin–destination matrix with entries $T_{ij}$
$T_{ij}$	Number of origin–destination (OD) trips from zone $i$ to zone $j$
$S_{ij}$	Number of intervening opportunities of origin–destination pair $(i, j)$
$\mathbf{f}$	Matrix with attraction values $f_{ij}$
$f_{ij}$	Attraction factor of zone $j$ to zone $i$
$\lambda_{\text{max}}$	Maximum arrival rate in a queueing network (equal to $\max_i \lambda_i$ )
$\mathcal{C}_n$	Set of directed networks with $n$ nodes that satisfy the reachability conditions
$\mu_i$	Service rate of single-server queue at node $i$
$\mu$	Homogeneous service rate of single-server queues
$\mathcal{U}_n$	Set of undirected networks with $n$ nodes that satisfy the reachability conditions
$\lambda_{\text{total}}$	Total arrival rate in a queueing network (equal to $\sum_i \lambda_i$ )



$C_p$	Permissible-edge-perturbation function that maps a network to the set of permissible edge perturbations
$\Lambda$	Distance matrix (with entries $d_{ij}$ ) associated with a graph $G$
$\theta$	Fraction of baskets in the data set (i.e., number of them relative to the total number of baskets during the same time period)
$O_k$	Number of OD trips that start at zone $k$ (it equals the sum of the entries in the $k^{\text{th}}$ row of $\mathbf{T}$ )
$D_k$	Number of OD trips that terminate at zone $k$ (it equals the sum of the entries in the $k^{\text{th}}$ column of $\mathbf{T}$ )
$C_s$	Number of shopping journeys in the data set (it equals $O_1 - D_1$ )
$N$	Total number of origin–destination trips (it equals $\sum_{i,j} T_{ij}$ )
$l$	Mean zone length (of a store network)
$v_k$	Estimated number of visits to zone $k$
$\omega_{ikj}$	Fraction of shortest paths from $i$ to $j$ that traverse node $k$
$r_{ij}$	Zone–zone correlation between zones $i$ and $j$

# Chapter 1

## Introduction

Understanding how customers move inside stores is of considerable interest to retailers [57], as it can help them improve store layouts with less congestion, more item sales, or other desirable features. However, the mobility of customers in a store is a complicated process that depends on many factors, including customers' item preferences, shopping lists, familiarity with a store's layout, the availability and price of items, and the level of congestion in a store. Some of these factors also vary over time (e.g., availability of items or congestion), and the flow of people or the way in which people navigate therefore varies over time. A mathematical model that explains the main features of customer mobility and that gives good estimates of the movements (even if on an aggregate level) is of great interest to retailers. Such a model can be used to improve store layouts, so that there is less congestion or greater exposure of promotional items.

This thesis is motivated by the problem of understanding how customers move in a supermarket and the problem of measuring and mitigating congestion inside a store. There are three primary objectives:

- model how customers move in a supermarket,
- model congestion based on customer movements, and
- identify store layouts that minimize congestion.

Supermarket companies seek to reduce congestion inside their stores, because it has a negative effect both on customer shopping experience and on the fulfilment time for online orders. In many supermarkets, staff members go around a store (at the same time as customers shop in the store) and pick up items that were ordered online. Congestion may delay such orders and thereby incur additional costs and inconvenience customers in the stores.

Until recently, there were only qualitative studies on customer mobility [104]. These studies were based largely on surveys and observations (e.g., by following customers around a store) [27, 85]. Recent technological developments now allow large-scale, anonymous, and non-invasive tracking of customer shopping journeys. The technologies include overhead cameras that can recognize movement (without recording video footage), radio-frequency identification (RFID) chips on trolleys whose positions are triangulated using ceiling sensors, and WiFi/Bluetooth sensors that can measure the journey of a mobile device. With the introduction of hand-held scanners that customers can use to scan the items that they pick up during a shopping journey, it is possible to infer the order in which customers pick up items and therefore the order of visited locations. Together with data about a store’s layout, one can use these data sources to analyse how store layout affects the shopping and mobility patterns of customers. Existing work that studied such customer-location data have been mainly of a descriptive nature; they have included clustering of customers by their trajectories [44, 47, 57, 105] and measuring how much customers deviate from a shortest route that visits the locations of their item purchases (i.e., an optimal travelling-salesman-problem (TSP) route) [42]. Existing models for customer mobility include both Markovian random-walk models [27], agent-based models [41, 106], and TSP models [14]. However, the first two types of models have so far yielded poor fits with empirical mobility data and the TSP models are prohibitively expensive to simulate.

In this thesis, we follow a similar approach to [41] for modelling a supermarket, which we represent as a network (also called a graph). Each node in the network is a zone (i.e., an area) in a store, and adjacent zones are connected by an edge. We use different models (including random-walk models and population-level mobility models) of how customers move and navigate around a supermarket. We compare some of the models to anonymized customer mobility data from stores of a major United Kingdom supermarket chain (Tesco). We use a queueing-network framework to model congestion in stores and examine queueing networks at stationarity. In this framework, each node is a queue, at which customers queue before traversing to the next node. The rate at which customers arrive at each queue is determined by the choice of mobility model. Our work therefore integrates concepts from queueing theory, network science, and human mobility. We summarize our approach in Figure 1.1.

The thesis is organized as follows. In Chapter 2, we define the relevant concepts that we use in this thesis and survey the literature on modelling human mobility, congestion, and customer mobility in supermarkets. In Chapter 3, we describe our

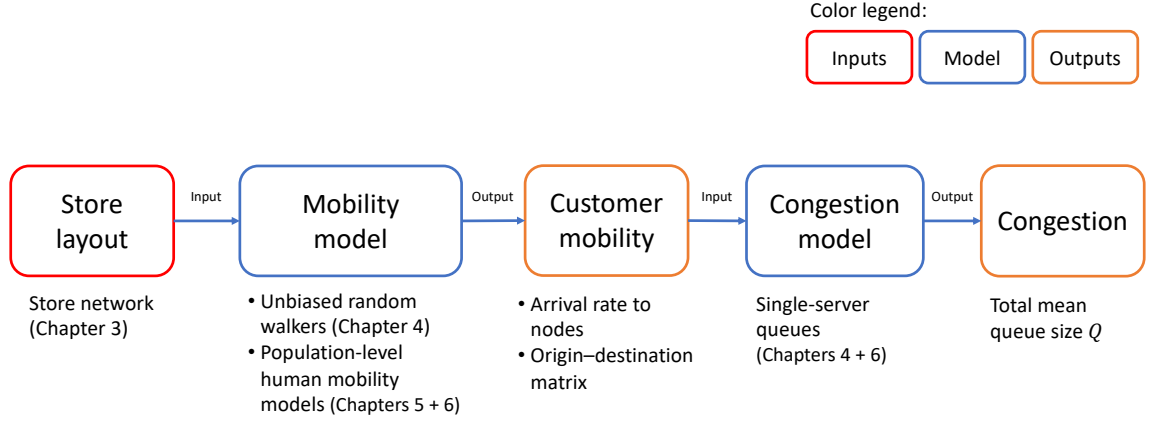


Figure 1.1. Summary of the approach that we take in this thesis. We represent a store as a network (see Chapter 3). In Chapters 4 and 5, we use mobility models that estimate customer mobility from a store’s layout (represented by a network). In Chapters 4 and 6, we use congestion models based on queueing networks to estimate congestion in a store from customer mobility.

procedure for representing stores as networks and present some properties of these networks. Chapters 4–6 are research chapters. In Chapter 4, we study queueing networks in which the entrance is a single source and the tills are a single sink. We assume that customers traverse the network according to an unbiased random walk and we analyse how network topology affects the total mean queue size  $Q$  (our measure for congestion). We examine network topologies that minimize  $Q$  and provide proofs for some cases and numerical evidence for others. We present greedy algorithms that add and delete edges from a network to reduce  $Q$ , and we apply these algorithms to an actual store. This study extends the work of Arenas *et al.* [6, 35], who analysed how congestion depends on network topology using a discrete-time congestion model that is related to queueing networks. In our work, we use queueing networks which a continuous-time congestion model and we use a different congestion measure (namely,  $Q$ ) from Arenas *et al.* However, we show that the congestion measure that was used by Arena *et al* is related to  $Q$ . In Chapter 5, we apply more realistic models for customer mobility by using population-level human-mobility models to characterize the flow of customers between zones. To the best of our knowledge, this work is the first application of these models to building-level spatial scales; previous applications have been on larger spatial scales, ranging from inter-country level on a scale of thousands of kilometres (e.g., estimating trade flow [4, 11]) to city and region levels at the scale of tens of kilometres (e.g., estimating commuting patterns [70]). We also propose an extension to these mobility models that gives a bias to flows between zones in

which items are frequently bought together. In Chapter 6, we integrate our human-mobility models from Chapter 5 with queueing networks to estimate congestion in supermarkets and optimize a store layout to reduce congestion. Finally, in Chapter 7, we give concluding remarks and outline avenues of research that our work has opened up. In appendices, we give more details on the numerical evidence for the conjectured network topologies, describe further results when using our greedy algorithms for edge addition and deletion to reduce  $Q$ , and present results of another human-mobility model.

## Chapter 2

# Networks, queues, mobility, and congestion modelling

In this chapter, we introduce key concepts and notation that we use throughout this thesis. We begin by introducing graphs in Section 2.1. In Section 2.2, we describe a traffic-dynamics model that has been used to model congestion on networks and examine the relationship between underlying network topology and congestion. In Section 2.3, we introduce queueing networks, the framework that we use for modelling congestion in supermarkets. We give a brief survey of the human-mobility models that we use to model customer mobility in Section 2.4 and present our framework for applying these mobility models. Finally, we survey the literature on modelling customer mobility in supermarkets in Section 2.5.

### 2.1 Networks

A *graph* (i.e., a *network*)  $G = (V, E)$  consists of a set  $V$  of *nodes* (i.e., *vertices*) and a set  $E$  of ordered pairs of nodes in  $V$  that constitute the *edges* (i.e., *arcs*) of  $G$  and encode the connections between nodes [78]. See Figure 2.1(a) for an example. Graphs are useful models of systems for many applications; for example, in road networks, one can model the intersections of roads as nodes and roads that connect the intersections as edges. Other examples include social networks, ecological networks, metabolic networks, internet networks, flight networks, and many more [78].

Formally, an edge  $(i, j)$  is a directed connection from  $i$  (the *initial node* of the edge) to  $j$  (the *terminal node* of the edge). We assume that  $i \neq j$  for any edge  $(i, j)$ ; that is, we do not consider *self-edges*. We also do not allow multiple edges (called *multi-edges*) from  $i$  to  $j$ . A graph is *undirected* if every edge  $(i, j) \in E$  implies that the reciprocal

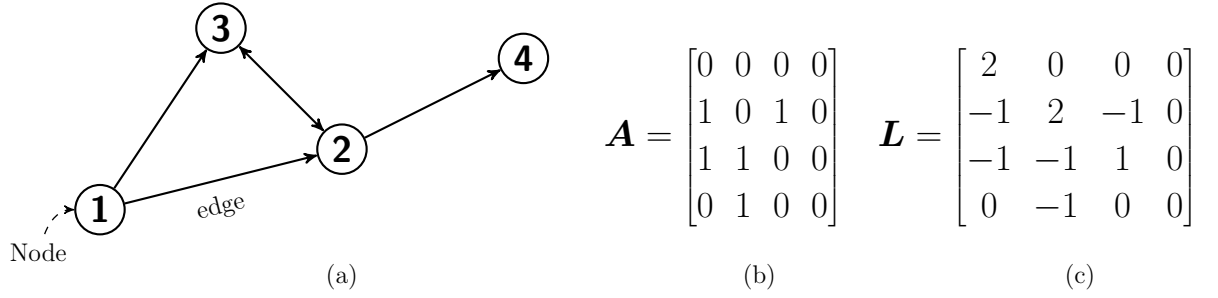


Figure 2.1. Example of (a) a directed graph with  $n = 4$  nodes and  $m = 5$  edges with (b) its adjacency matrix  $\mathbf{A}$  and (c) its combinatorial Laplacian matrix  $\mathbf{L}$ .

edge  $(j, i)$  is also in  $E$ . Otherwise (i.e., if  $(i, j) \in E$  does not imply that  $(j, i) \in E$ ), the graph is *directed*. In this thesis, we consider both undirected and directed graphs.

In an undirected graph, one usually replaces each reciprocal pair of edges  $(i, j)$  and  $(j, i)$  by  $\{i, j\}$ , such that  $E \subset \binom{V}{2}$  (where  $\binom{V}{2}$  is the set of unordered pairs of nodes). In this thesis, we use  $\{i, j\}$  to represent an undirected edge. However, in definitions that apply to both directed and undirected networks, we use the notation  $(i, j)$  to denote the undirected edge  $\{i, j\}$  if the network is undirected and the directed edge  $(i, j)$  otherwise. In a directed graph, a directed edge  $(i, j)$  is an *out-edge* of node  $i$  and an *in-edge* of node  $j$ . When  $(i, j) \in E$ , we say that  $i$  is *adjacent* to  $j$  in  $G$  and that  $e$  is *incident* to nodes  $i$  and  $j$ . If the graph is undirected, we say that  $i$  and  $j$  are *neighbours*; if the graph is directed, we say  $j$  is an *out-neighbour* of  $i$  and  $i$  is an *in-neighbour* of  $j$ .

Throughout this thesis, we denote the number of nodes by  $n$  and the number of edges by  $m$ .<sup>1</sup> The node set  $V$  is always  $V = \{1, \dots, n\}$ . The *topology* of a network is the pattern of the underlying connections of the network (i.e., the presence or absence of edges between nodes, along with the structure that they form). We do not consider *weighted networks*, in which each edge is associated with a weight.

## Adjacency matrix

A useful way to represent a graph  $G = (V, E)$  is via an *adjacency matrix*  $\mathbf{A}$ . The adjacency matrix  $\mathbf{A}$  of  $G$  is a  $n \times n$  matrix with entries

$$A_{ij} = \begin{cases} 1, & \text{if } (j, i) \in E, \\ 0, & \text{otherwise.} \end{cases} \quad (2.1)$$

<sup>1</sup>When we consider undirected graphs,  $m$  is the number of *undirected* edges.

See Figure 2.1(b) for an example. Note that we use the convention that  $A_{ij} = 1$  implies that there is an edge from  $j$  to  $i$  (instead of from  $i$  to  $j$ ). The adjacency matrix  $\mathbf{A}$  is symmetric in an undirected graph and asymmetric in general in a directed graph.

## Degree

The *degree*  $d_i$  of a node  $i$  in an undirected graph  $G$  is the number of neighbours of  $i$ . We can calculate it from the adjacency matrix  $\mathbf{A}$  by

$$d_i = \sum_{j \in V} A_{ij} = \sum_{j \in V} A_{ji}. \quad (2.2)$$

In a directed graph, we distinguish between the *out-degree*  $d_i^{\text{out}}$  and the *in-degree*  $d_i^{\text{in}}$  of node  $i$ . The out-degree and in-degree of node  $i$  are the numbers of its out-edges and in-edges, respectively. Again, we can calculate  $d_i^{\text{out}}$  and  $d_i^{\text{in}}$  from  $\mathbf{A}$ :

$$d_i^{\text{out}} = \sum_{j \in V} A_{ji}, \quad (2.3)$$

$$d_i^{\text{in}} = \sum_{j \in V} A_{ij}. \quad (2.4)$$

In matrix form,

$$\mathbf{d}^{\text{out}} = \mathbf{A}^T \mathbf{1}, \quad (2.5)$$

$$\mathbf{d}^{\text{in}} = \mathbf{A} \mathbf{1}, \quad (2.6)$$

where  $\mathbf{d}^{\text{out}}$  and  $\mathbf{d}^{\text{in}}$  are the vectors of in-degrees and out-degrees, respectively, and  $\mathbf{1}$  is an  $n$ -dimensional vector of ones. Additionally,

$$\sum_{i \in V} d_i^{\text{out}} = \sum_{i \in V} d_i^{\text{in}}. \quad (2.7)$$

A *degree sequence*  $(d_1, \dots, d_n)$  is a sequence of degrees of nodes in an undirected graph. A *degree distribution* is a probability distribution for the degrees of the nodes of an undirected graph. In a directed graph, there are both *in-degree* and *out-degree sequences* and *distributions*.

## Laplacian

Another matrix that we use to encode the connectivity of a network  $G$  is the *combinatorial Laplacian*  $\mathbf{L}$  [78] (often simply called the *Laplacian*<sup>2</sup>), which has entries

$$L_{ij} = \begin{cases} d_i \delta_{ij} - A_{ij}, & \text{if } G \text{ is undirected,} \\ d_i^{\text{out}} \delta_{ij} - A_{ij}, & \text{otherwise,} \end{cases} \quad (2.8)$$

---

<sup>2</sup>There are several variants of the Laplacian. In this thesis, we use the combinatorial Laplacian.



where  $\delta_{ij}$  is the *Kronecker delta*. In matrix form, the Laplacian is

$$\mathbf{L} = \mathbf{D} - \mathbf{A}, \quad (2.9)$$

where the diagonal *degree matrix*  $\mathbf{D}$  has entries

$$D_{ij} = \begin{cases} d_i \delta_{ij}, & \text{if } G \text{ is undirected,} \\ d_i^{\text{out}} \delta_{ij}, & \text{otherwise.} \end{cases} \quad (2.10)$$

See Figure 2.1(c) for an example.

One can interpret the Laplacian matrix as a discrete diffusion operator, which describes the linear diffusion on a graph  $G$ . Suppose  $\phi = (\phi_1, \dots, \phi_n)$  is the heat distribution on  $G$ , where  $\phi_i$  is the heat of node  $i \in V$ . Suppose further that heat is transferred from  $j$  to  $i$  in proportion to  $\phi_j - \phi_i$  (according to Newton's law of cooling) if  $(j, i) \in E$ . In an undirected<sup>3</sup> graph,  $\phi$  then satisfies [78]

$$\frac{d\phi}{dt} = -\kappa \mathbf{L}\phi, \quad (2.11)$$

where  $\kappa$  is the heat capacity.

In an undirected graph  $G = (V, E)$ , the Laplacian  $\mathbf{L}$  can be expressed as a sum of rank-1 matrices as follows. For each undirected edge  $e = \{i, j\} \in E$ , the *incidence vector*  $\mathbf{b}_e$  is an  $n \times 1$  vector with exactly two non-zero entries, which occur at the  $i^{\text{th}}$  and  $j^{\text{th}}$  position. We arbitrarily<sup>4</sup> set one of the entries to 1 and the other to  $-1$ . We can then write  $\mathbf{L}$  as a sum of  $m$  rank-1 matrices [78]:

$$\mathbf{L} = \sum_{e \in E} \mathbf{b}_e \mathbf{b}_e^T. \quad (2.12)$$

In matrix form,

$$\mathbf{L} = \mathbf{B}\mathbf{B}^T, \quad (2.13)$$

where  $\mathbf{B}$  is the  $n \times m$  *node-incidence matrix*, which is formed by vertically stacking the  $m$  incidence vectors (i.e.,  $\mathbf{B} = (\mathbf{b}_{e_1}, \dots, \mathbf{b}_{e_m})$ , where  $e_1, \dots, e_m$  are the edges of  $G$ ).

In directed graphs, we write:

$$\mathbf{L} = \sum_{e \in E} \mathbf{a}_e \mathbf{c}_e^T, \quad (2.14)$$

---

<sup>3</sup>For directed graphs,  $\phi$  satisfies Equation (2.11), but  $\mathbf{L}$  is now a variant of the Laplacian matrix with entries  $L_{ij} = d_i^{\text{in}} \delta_{ij} - A_{ij}$ . Our present definition of the Laplacian for directed graphs instead describes, for example, consensus dynamics (rather than diffusion), where the state of node  $i$  is influenced by its out-neighbours (rather than its in-neighbours) [71].

<sup>4</sup>For example, we can use the convention that the entry corresponding to the smaller index (i.e.,  $\min(i, j)$ ) is 1 and the other entry is  $-1$ .

where  $\mathbf{a}_e$  and  $\mathbf{c}_e$  are  $n \times 1$  vectors for each directed edge  $e = (i, j)$ . These vectors have entries

$$[a_e]_k = \begin{cases} 1, & \text{if } k = i, \\ 0, & \text{otherwise,} \end{cases} \quad (2.15)$$

and

$$[c_e]_k = \begin{cases} -1, & \text{if } k = j, \\ 1, & \text{if } k = i, \\ 0, & \text{otherwise.} \end{cases} \quad (2.16)$$

In matrix form,

$$\mathbf{L} = \mathbf{B}_1 \mathbf{B}_2^T, \quad (2.17)$$

where  $\mathbf{B}_1 = (\mathbf{a}_{e_1}, \dots, \mathbf{a}_{e_m})$  and  $\mathbf{B}_2 = (\mathbf{c}_{e_1}, \dots, \mathbf{c}_{e_m})$ .

### Walks, paths, and reachability

A *walk* in a (directed or undirected) graph  $G = (V, E)$  starting at node  $s$  and ending at node  $t$  is a sequence  $(v_0, \dots, v_l)$  of nodes, such that  $(v_i, v_{i+1}) \in E$  for  $i \in \{0, \dots, l-1\}$ , with  $v_0 = s$  and  $v_l = t$ . A *path* in  $G = (V, E)$  is a walk in which the nodes do not repeat (except for a possible duplication of the starting and ending nodes). The *length* of a walk is the number of edges that the walk traverses (i.e., the length of the sequence minus 1). A node  $t$  is *reachable* from  $s$  if there is path starting from  $s$  that ends at  $t$ .

If there is a distance (i.e., length)  $l_{ij}$  associated with each edge  $e = (i, j)$ , we define the length of a walk as the sum of the distances of the edges that it traverses (instead of the number of edges that the walk traverses). Note that we consider a graph to be unweighted, even if there is a distance associated with each edge.

A *shortest path* [78] (also called a *geodesic path*) from node  $s$  to node  $t$  in  $G = (V, E)$  is a path from  $s$  to  $t$  of minimal length. That is, there does not exist another path from  $s$  to  $t$  with strictly shorter length. For example,  $(1, 2, 4)$  is a shortest path from 1 to 4 in the graph in Figure 2.1(a). (It is also the unique shortest path in this graph, but, in general, there may be multiple shortest paths from  $s$  to  $t$ .)

### Induced subgraph

The induced subgraph  $G[S]$  of  $G$  on  $S \subset V$  is the graph  $G[S] = (S, E_S)$ , where  $E_S \subset E$  consists of the edges whose incident nodes are both in  $S$ . In other words, we obtain the induced subgraph  $G[S]$  by removing all nodes and associated incident edges that are not in  $S$  from  $G$ .

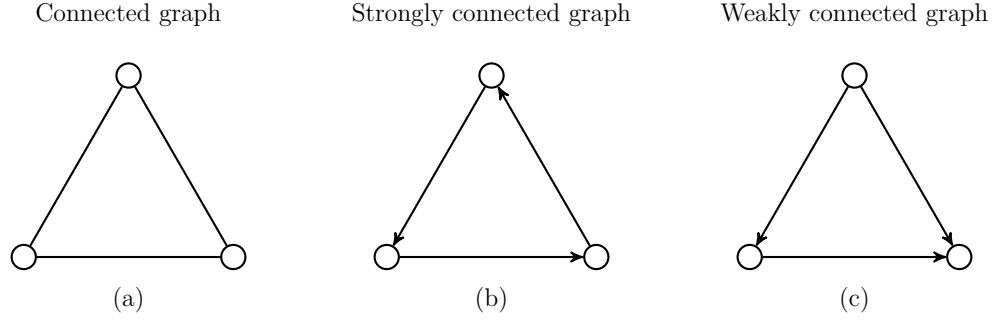


Figure 2.2. Examples of (a) a connected, (b) a strongly connected, and (c) a weakly connected graph.

## Connectedness and components

An undirected graph  $G = (V, E)$  is *connected* if there is a walk from any node  $i$  to any other node  $j$  in  $G$ . Otherwise, the graph is *disconnected*. We say a directed graph  $G = (V, E)$  is *strongly connected* if there is a walk from any node  $i$  to any other node  $j$  in  $G$ . A directed graph  $G = (V, E)$  is *weakly connected* if the undirected graph obtained from  $G$  by replacing every directed edge by an undirected edge (or bidirectional edge) is connected. See Figure 2.2 for examples of connected, strongly, and weakly connected graphs. Note that a strongly connected graph is automatically weakly connected.

A *connected component* of an undirected graph  $G = (V, E)$  is a subset  $S$  of nodes of  $G$  such that the induced graph  $G[S]$  is connected and there is no other node  $v \in V \setminus S$  that is reachable from a node in  $S$ . Therefore, one can partition the node set  $V$  into the connected components of  $G$ .

In a directed graph  $G = (V, E)$ , the *in-component*  $C_{\text{in}}(i)$  of node  $i$  is the set of nodes  $j$  for which there exists a path from  $j$  to  $i$ , plus  $i$  itself. The *out-component*  $C_{\text{out}}(i)$  of node  $i$  is the set of nodes  $j$  for which there exists a path from  $i$  to  $j$ , plus  $i$  itself.

### 2.1.1 Regular, complete, spatial, and planar graphs

An undirected graph is a *d-regular graph* if every node has the same degree  $d \in \mathbb{Z}_+$ . (We define  $\mathbb{Z}_+$  to be the set of non-negative integers.)

An undirected graph is a *complete graph* (i.e., a *fully connected graph*) if there is an edge between any pair of distinct nodes.

We say that a (directed or undirected) graph is a *spatial graph* (i.e., a *spatial network*) if each node is associated with a point in a metric space [8]. In many

cases, the metric space is  $\mathbb{R}^d$ , where  $d$  is a positive integer (usually  $d = 2$  or  $d = 3$ ). Throughout the thesis, the spatial networks that we consider are embedded in  $\mathbb{R}^2$ .

A graph is a *planar graph* if there exists a mapping from every node to a point on  $\mathbb{R}^2$  and from every edge to a curve on  $\mathbb{R}^2$  that satisfies the following two conditions.

1. The extreme points of each curve are the points mapped to its end nodes.
2. All curves are disjoint except on their extreme points.

Informally speaking, a graph is planar if one can draw the graph on the plane such that no edges cross each other. Therefore, to show that a graph is planar, it suffices to show a representation of the graph on  $\mathbb{R}^2$  such that no two edges cross each other. Planar networks are often good approximations of spatial networks (such as rail or road networks), even when the real networks are not themselves planar (e.g., due to bridges in transportation networks) [8].

### 2.1.2 Random walks on graphs

Random walks are the simplest model for diffusion on a network. A discrete-time *unbiased random walk* on a network  $G$  with adjacency matrix  $\mathbf{A}$  is a Markov chain on the nodes of  $G$  with transition matrix  $\mathbf{P}$ , where the probability of moving to node  $j$  when starting from node  $i$  is  $P_{ij} = A_{ji} / \sum_k A_{ki}$  if  $\sum_k A_{ki} > 0$  and is equal to 0 otherwise. For a review on random walks on networks, see [71].

### 2.1.3 Random-graph models

A *random-graph model* is a probability distribution over a set of graphs [78]. Many random-graph models aim to generate some aspects of real-world networks. We present six models below.

#### Erdős–Rényi graph

An Erdős–Rényi (ER) graph  $G(n, p)$  [25] (also called a *Bernoulli random graph*) with  $p \in [0, 1]$  is an undirected graph with  $n$  nodes, in which any two nodes  $i$  and  $j$  are connected by an edge with probability  $p$ , independently for each pair of nodes.<sup>5</sup> The

<sup>5</sup>The original formulation of the ER graph is the random graph  $G(n, m)$ , which is an undirected graph with  $n$  nodes and exactly  $m$  edges, chosen uniformly at random among all graphs with  $n$  nodes and  $m$  edges. The two models  $G(n, p)$  and  $G(n, m)$  are closely related and have largely the same behaviour in the asymptotic limit as  $n \rightarrow \infty$ . For example,  $G(n, p)$  has a monotone property  $P$  almost surely as  $n \rightarrow \infty$  with  $np^2 \rightarrow \infty$  if and only if  $G(n, m)$  with  $m = \binom{n}{2}p$  has the property  $P$  almost surely as  $n \rightarrow \infty$  [13]. (A property  $P$  is *monotone* with respect to subgraph ordering if for

number of edges follows a binomial distribution with mean

$$\mathbb{E}[m] = \frac{n(n-1)p}{2}. \quad (2.18)$$

See Figure 2.3 for an example of an ER graph.

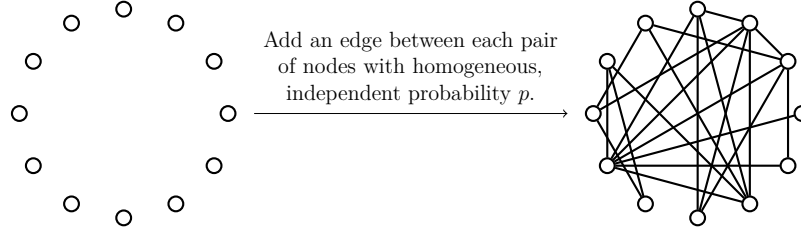


Figure 2.3. Schematic for generating an Erdős-Rényi graph  $G(n, p)$ , where  $n = 12$  and  $p = 0.2$ .

### Barabási-Albert graph

A *Barabási-Albert* (BA) graph [3] is an undirected network that is created as follows. Given the parameters  $n \in \mathbb{Z}_+$  and  $m_{BA} \in \mathbb{Z}_+$  and an initial undirected network with  $m_0 \geq m_{BA}$  nodes, we add one node to the network at a time until the network has  $n$  nodes. Each new node  $j$  attaches to  $m_{BA}$  distinct, existing nodes, where each node  $i$  is chosen with probability proportional to its degree  $d_i$ .

The BA model is one of the most famous preferential-attachment models. One of the key features of the model is that it exhibits a power-law degree distribution as  $n \rightarrow \infty$ . Many real-world networks exhibit heavy-tailed degree distributions (such as a power-law distribution). The power-law degree distribution arises due to a form of the preferential-attachment mechanism in the BA model, in which high-degree nodes tend to receive more edges than low-degree nodes during the generation process of the BA model.

In this thesis, we set the initial network to be the graph with  $m_0 = m_{BA}$  nodes and no edges. Therefore, every BA graph with the same parameters has the same number  $m$  of edges, with

$$m = (n - m_{BA})m_{BA}. \quad (2.19)$$

See Figure 2.4 for an example of a BA network. There are also many other preferential-attachment models [78].

---

any graph  $H$  that has the property  $P$ , then any graph  $G$  for which  $H$  is a subgraph of  $G$  also has the property  $P$ . For example, having a path of length 3 is a monotone property.)

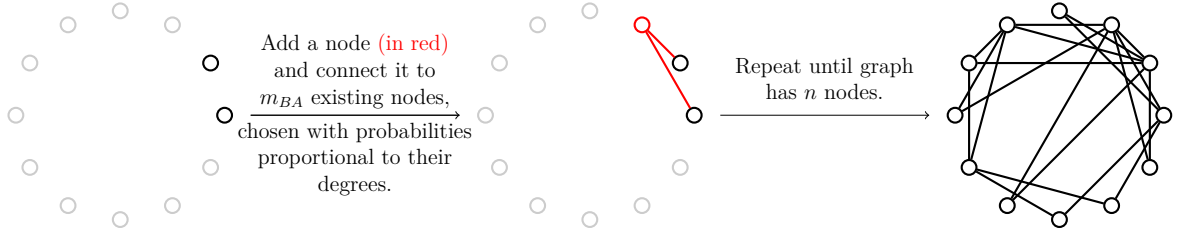


Figure 2.4. Schematic for generating a Barabási–Albert graph with  $n = 12$  and  $m_{BA} = 2$ . The initial network has  $m_0 = 2$  nodes and no edges.

### Watts–Strogatz graph

The *Watts–Strogatz* (WS) model [99] is a model that exhibits *small-world* behaviour for many parameter values. That is, this model often produces undirected graphs that tend to have low mean shortest-path length<sup>6</sup>. A further feature is that the graphs generated by the WS model tend to have large clustering coefficients (that is, neighbours of a node tend to be neighbours as well). Low mean shortest-path lengths and large clustering coefficient are two features that are often observed in real-world networks [78].

Given parameters  $n \in \mathbb{Z}_+$ ,  $k_{WS} \in \mathbb{Z}_+$ , and  $p \in [0, 1]$ , we generate a *Watts–Strogatz* graph as follows. Starting with a graph with  $n$  nodes arranged in an regular  $n$ -gon, we add undirected edges such that each node is adjacent to its closest  $2k_{WS}$  nodes (see Figure 2.5(a) and (b)). For each node  $i$ , we then consider the edges that connect  $i$  with its  $k_{WS}$  rightmost neighbours. We rewire each of the  $k_{WS}$  edges with independent probability  $p$  as follows. For each edge  $\{i, j_{old}\}$  to be rewired, we choose a node  $j_{new}$  uniformly at random from all nodes that are not neighbours of  $i$  and replace  $\{i, j_{old}\}$  with  $\{i, j_{new}\}$  (see Figure 2.5(c)).

In this thesis, we use a variant of the original Watts–Strogatz model (which we described above) that produces connected WS networks by repeatedly sampling WS networks using the procedure above until it produces a connected graph.

The number  $m$  of edges in each WS graph is

$$m = nk_{WS}. \quad (2.20)$$

### Random geometric graph

Given parameters  $n \in \mathbb{Z}_+$  and  $r \in [0, \sqrt{2}]$ , we define a *random geometric graph* (RGG) [8] to be a spatial, undirected graph in  $\mathbb{R}^2$  in which  $n$  nodes are placed uniformly at

<sup>6</sup>The mean shortest-path length is the mean of the shortest-path lengths between two nodes, averaged over all pairs of distinct nodes.

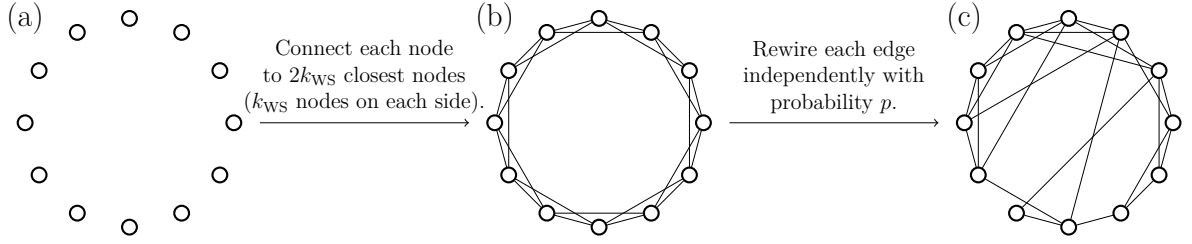


Figure 2.5. Schematic for generating a Watts–Strogatz graph, where  $n = 12$ ,  $k_{\text{WS}} = 2$ , and  $p = 0.1$ .

random in the unit square in  $\mathbb{R}^2$  and each node is adjacent to all nodes within a radius  $r$ . In this thesis, we use the Euclidean distance as our distance.<sup>7</sup>

For small values of  $r$ , the mean number of edges of a RGG is

$$\mathbb{E}[m] \approx \frac{n(n-1)\pi r^2}{2}. \quad (2.21)$$

Due to boundary effects, the precise value of  $\mathbb{E}[m]$  is smaller than the right-hand side of Equation (2.21). Equation (2.21) becomes exact as  $n \rightarrow \infty$  and  $r \rightarrow 0$ , with  $nr$  fixed, or if we introduce periodic boundary conditions (such that the unit square becomes a torus).

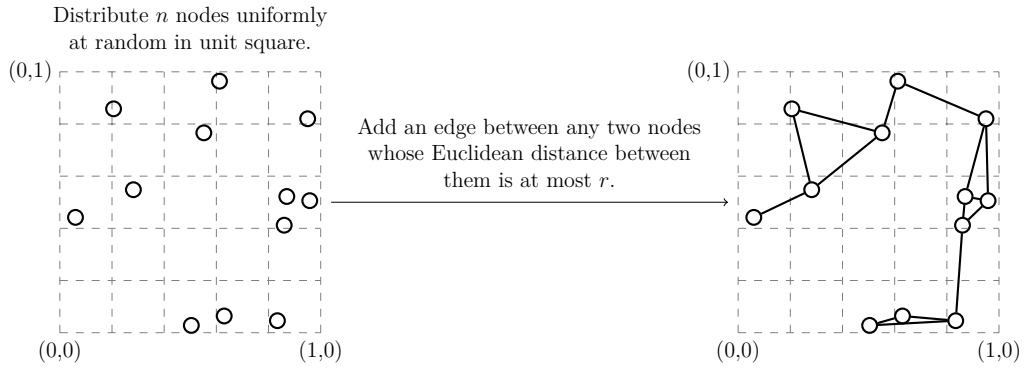


Figure 2.6. Schematic for generating a random geometric graph with  $n = 12$  and  $r = 0.4$ .

## Random regular graph

Given parameters  $n \in \mathbb{Z}_+$  and  $d \in \mathbb{Z}_+$  such that  $nd$  is even, a *random regular graph* is a  $d$ -regular graph on  $n$  nodes chosen uniformly at random from all  $d$ -regular graphs with  $n$  nodes. See Figure 2.7 for an example. We use the algorithm by Kim and Vu [52]

<sup>7</sup>Our definition is the simplest, traditional version of a RGG. See [81] for more general versions of RGGs.

that samples a  $d$ -regular graph in an asymptotically uniform way when  $d = \mathcal{O}(n^{1/3-\epsilon})$  for any  $\epsilon < 1/3$  as  $n \rightarrow \infty$ .

The number of edges of a random regular graph is

$$m = \frac{nd}{2}. \quad (2.22)$$

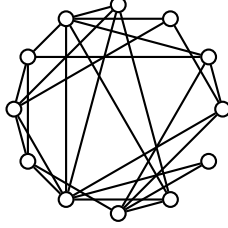


Figure 2.7. A random regular graph with  $n = 12$  and  $d = 4$ .

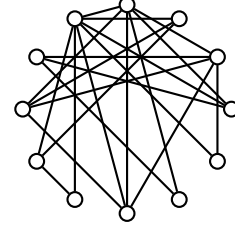


Figure 2.8. A Chung-Lu graph with  $n = 12$  and the degree sequence generated from a BA graph with  $n = 12$  and  $m_{BA} = 2$ .

## Configuration model and Chung-Lu model

A *configuration model* [28] is a uniform distribution of undirected<sup>8</sup> graphs with a specific degree sequence<sup>9</sup>  $(d_1, \dots, d_n)$ . The model has been used to sample a ‘typical graph’ with a given degree distribution. This is often done to assess whether there are any structural features in a network that cannot be explained only by the degree distribution. A widely used way of generating a configuration-model graph is the *stub-matching process* [28], in which each node  $i$  has  $d_i$  stubs (i.e., half-edges). At each step, we choose two stubs uniformly at random and replace them by an edge that connects the end nodes of the stubs. This is done until no more stubs are left. This process can lead to *multi-edges* (multiple edges between two pairs of nodes) and self-edges, though they are typically rare for large networks.

A variant of the configuration model is the *Chung-Lu model* [18], which does not produce multi-edges or self-edges.<sup>10</sup> Given  $n \in \mathbb{Z}_+$  and a sequence  $(w_1, \dots, w_n)$  of positive weights (representing the expected degree sequence), we place an edge  $\{i, j\}$  between two nodes  $i$  and  $j$  (with  $i \neq j$ ) with independent probability  $w_i w_j / \sum_k w_k$ . See Figure 2.8 for an example. The expected degree  $\mathbb{E}[d_i]$  of node  $i$  is then

$$\mathbb{E}[d_i] = w_i \left( 1 - \frac{w_i}{\sum_k w_k} \right), \quad (2.23)$$

<sup>8</sup>Configuration models for directed graphs exists as well [28], but we only consider versions for undirected graphs in this thesis.

<sup>9</sup>One can also start with a degree distribution and sample a degree sequence from the degree distribution.

<sup>10</sup>The original Chung-Lu model includes self-edges, but we choose a variant that excludes them.



which tends to  $w_i$  as  $n \rightarrow \infty$ .

The expected number of edges is

$$\mathbb{E}[m] = \frac{\sum_i \mathbb{E}[d_i]}{2}. \quad (2.24)$$

## 2.2 Traffic on networks

We seek to examine how the topology of a network affects traffic dynamics and congestion on it. For an introduction to dynamics on networks, see [83]. In this section, we focus on one particular discrete-time traffic model [76, 79, 112] that has been studied widely in the context of congestion on networks, with applications to communication networks. We refer to this model as the *traffic-dynamics model*, and we present the main features of the model. See [17] for more details on the model.

In the traffic-dynamics model, each node  $i$  of an unweighted, strongly connected network generates a random number  $N_i$  of walkers (or “packets”, in the context of a communication network) at each time step, where  $N_i$  is Poisson distributed at rate  $\rho$ , independently of all other nodes. The rate  $\rho$  is also called the *packet-generation rate*. The mean number of generated walkers in the entire network is therefore  $n\rho$  per time step. (For simplicity, we assume each time step is of length 1.)

Each walker is assigned a destination, which is chosen uniformly at random from all nodes except the node from which it was generated. The walker moves according to specific routing strategies, which can be stochastic (e.g., a random walk, in which the next node is chosen randomly from all neighbouring nodes) or deterministic (e.g., shortest-path routing, in which the walker always follows a shortest path from its current node to the destination node).

Each node  $i$  is assigned a constant *delivering capacity*  $C_i$ , which specifies the maximum number of walkers that can leave the node per time step. If the number of walkers at a node  $i$  exceeds the capacity  $C_i$  of the node, a queue forms and only  $C_i$  walkers are delivered to their next node in the current time step. The queue is a *first-in-first-out* (FIFO) queue, so the walkers that arrive first also leave first. (Walkers that arrive at the same time are ordered randomly.) We assume that the capacity is homogeneous, so  $C_i = C$  for some constant  $C$ . Finally, whenever walkers reach their destination, they exit the system.

The traffic-dynamics model is a special case of a queueing-network model (which we define formally in Section 2.3). In this special case, each node is a FIFO service station with  $C$  servers with deterministic service time of exactly 1 time step. On average,  $\rho$

walkers arrive from outside to each node at each time step. The advantage of this model (in comparison to more complicated models or using an agent-based model) is that it is simple, tractable, and applicable to a wide range of routing strategies [17]. In particular, under any Markovian routing strategy, one can derive a simple formula for congestion measures, as we show in Section 2.2.1. The model can also be used to test the effectiveness of routing strategies [22, 23, 103, 111]. In a communication-network application, walkers represent packets that are sent through a network via routers, which constitute the nodes of the network.

### 2.2.1 Traffic capacity

The traffic-dynamics model is typically studied at *stationarity* (if such state exists); at stationarity, the distribution of the number of people at each node does not change over time. One can show that the model undergoes a continuous phase transition [35, 76, 79] at  $\rho = \rho_c$  for some critical packet generation rate  $\rho_c$  (called the *traffic capacity*). The value of  $\rho_c$  depends on the routing strategy and the network topology. When  $\rho < \rho_c$ , a stationary state exists and the network is in a *free-flow state*, in which the mean number of walkers in the network is constant over time (as  $t \rightarrow \infty$ ). When  $\rho > \rho_c$ , no stationarity state exists and the number of walkers increases linearly with time. The number of walkers in the system tends to infinity as  $t \rightarrow \infty$ , leading to congestion (and so-called a *congested state*). To characterize the transition point  $\rho_c$ , one can use an *order parameter* [35] to characterize the relative mean rate of system-size increase per time step as  $t \rightarrow \infty$ . It is

$$\eta(\rho) = \frac{1}{\rho n} \lim_{t \rightarrow \infty} \frac{\sum_{k=0}^t \Delta N(k)}{t}, \quad (2.25)$$

$N(t)$  denotes the system size (i.e., the number of walkers in the system) at time  $t$ , and the quantity  $\Delta N(t) = N(t+1) - N(t)$  denotes the change in the system size over one time step. The term inside the limit in Equation (2.25) is the mean system-size increase per time step for the time period up to time  $t$ . The order parameter  $\eta(\rho)$  is the limit of this term as  $t \rightarrow \infty$  divided by the total number  $\rho n$  of packets that is generated per time step.

In a free-flow state (i.e., when  $\rho < \rho_c$ ), the system size stays constant, so  $\eta(\rho) = 0$ . In the congested state (i.e., when  $\rho > \rho_c$ ), the system size grows linearly with  $t$ , so  $\eta(\rho) > 0$ , indicating that the system becomes congested. Therefore,  $\rho_c$  is

$$\rho_c = \sup\{\rho > 0 : \eta(\rho) = 0\}. \quad (2.26)$$

Equation (2.26) is not a practical<sup>11</sup> characterisation for  $\rho_c$ , but one can express  $\rho_c$  in closed form using the effective betweenness centrality of the network [35]. The *effective betweenness centrality* (or simply *effective betweenness*)  $B_j^{\text{eff}}$  of a node  $j$  is defined as

$$B_j^{\text{eff}} = \sum_{\substack{i,k \in V \\ i \neq k}} b_{ij}^{(k)}, \quad (2.27)$$

where  $b_{ij}^{(k)}$  is the mean number of times that a walker who enters the system at  $i$  with designated destination  $k$  visits  $j$ . Effective betweenness centrality depends both on network topology and on the routing strategy. It is a generalized form of node betweenness centrality (and, in particular, of *geodesic node betweenness centrality* [30]), and we recover a version of the original betweenness centrality when walkers always take a shortest path to their destination. At stationarity in the free-flow regime, we calculate the mean number of walkers that arrive at a given node  $j$  as follows. Each node  $i$  generates walkers with destination  $k$  at a rate of  $\rho/(n-1)$ . Each such walker visits  $j$  on average  $b_{ij}^{(k)}$  times, so the total mean number of walkers who arrive at  $j$  per time step is  $\sum_{\substack{i,k \in V \\ i \neq k}} b_{ij}^{(k)} \rho/(n-1) = \rho B_j^{\text{eff}}/(n-1)$ . If  $\rho B_j^{\text{eff}}/(n-1) > C$ , more walkers arrive than the node can deliver, and walkers accumulate at the node, creating congestion. To ensure a free-flowing state, we therefore need  $\rho B_j^{\text{eff}}/(n-1) \leq C$  for all  $j$ . Therefore, the traffic capacity  $\rho_c$  is

$$\rho_c = \frac{C(n-1)}{B_{\max}^{\text{eff}}}, \quad (2.28)$$

where  $B_{\max}^{\text{eff}}$  is the maximum effective betweenness centrality of the network.

For Markovian routing — i.e., when the next step for each packet depends only on its current position — effective betweenness can be written in terms of transition matrices [35]. For each possible destination  $k \in \{1, \dots, n\}$ , let  $\mathbf{P}^{(k)}$  be the  $n \times n$  transition matrix that dictates how a walker with destination  $k$  navigates the network. The  $(i, j)$  entry  $P_{ij}^{(k)}$  of  $\mathbf{P}^{(k)}$  is the probability that a packet at  $i$  moves to  $j$  in the next move. The probability  $\tilde{P}_{ij}^{(k)}(M)$  of going from  $i$  to  $j$  in  $M \geq 1$  steps is then

$$\tilde{P}_{ij}^{(k)}(M) = \sum_{l_1, \dots, l_{M-1} \in V} P_{il_1}^{(k)} P_{l_1 l_2}^{(k)} \cdots P_{l_{M-1} j}^{(k)}, \quad (2.29)$$

with  $\tilde{P}_{ij}^{(k)}(0) = \delta_{ij}$ . In matrix form, this is

$$\tilde{\mathbf{P}}^{(k)}(M) = \left( \mathbf{P}^{(k)} \right)^M. \quad (2.30)$$

---

<sup>11</sup>In a simple method of approximating  $\rho_c$ , one calculates  $\eta(\rho)$  for many values of  $\rho$  and finds the largest  $\rho$  for which  $\eta(\rho) \approx 0$ . Each  $\eta(\rho)$  calculation requires performing many simulations, so the full calculation is expensive.

Let  $(V_0, V_1, \dots)$  be the random path of a walker that starts at  $i$  with destination  $k$ , where  $V_M$  is a random variable that denotes the position of the walker after  $M$  steps. (If a walker reaches  $k$  after  $T$  steps, we define  $V_M = k$  for all  $M \geq T$ .) The total number  $v_j$  of visits to  $j$  ( $j \neq i$ ) of such a walker is then equal

$$v_j = \sum_{M=0}^{\infty} \mathbf{1}_{\{V_M=j\}}, \quad (2.31)$$

where  $\mathbf{1}_{\{V_M=j\}}$  is the indicator variable that is equal to 1 if  $V_M = j$  and 0 otherwise. Note that

$$\mathbb{E}[\mathbf{1}_{\{V_M=j\}}] = \tilde{P}_{ij}^{(k)}(M). \quad (2.32)$$

Therefore the mean number  $b_{ij}^{(k)}$  of visits to  $j$  of a walker who enters the system at  $i$  with designated destination  $k$  is

$$b_{ij}^{(k)} = \sum_{M=0}^{\infty} \mathbb{E}[\mathbf{1}_{\{V_M=j\}}] = \sum_{M=0}^{\infty} \tilde{P}_{ij}^{(k)}(M). \quad (2.33)$$

In matrix form,

$$\mathbf{b}^{(k)} = \sum_{M=0}^{\infty} \tilde{\mathbf{P}}^{(k)}(M) = \sum_{M=0}^{\infty} (\mathbf{P}^{(k)})^M = (\mathbf{I} - \mathbf{P}^{(k)})^{-1} \mathbf{I}, \quad (2.34)$$

where  $\mathbf{b}^{(k)}$  is the  $n \times n$  matrix whose  $(i, j)$  entry is  $b_{ij}^{(k)}$  and  $\mathbf{I}$  is the identity matrix. Note that  $\rho(\mathbf{P}^{(k)}) < 1$  [35], where  $\rho(\mathbf{P}^{(k)})$  is the magnitude (called the *spectral radius*) of the leading eigenvalue of  $\mathbf{P}^{(k)}$ , so  $\mathbf{I} - \mathbf{P}^{(k)}$  has a well-defined inverse. Therefore, if we know the transition matrices  $\mathbf{P}^{(k)}$ , we can compute the effective betweenness  $B_j^{\text{eff}}$  by calculating all matrices  $\mathbf{b}^{(k)}$  using Equation (2.34).

*Local search* is one example of Markovian routing. In it, a walker moves to its destination  $k$  if it is a neighbour node of its present position  $i$ ; otherwise, it goes to a neighbour node  $j$  with transition probability  $d_j^\alpha / \sum_l d_l^\alpha$ , where  $d_l$  is the degree<sup>12</sup> of node  $l$  and  $\alpha > 0$  is a parameter. The full transition probability is therefore

$$P_{ij}^{(k)} = A_{ik} \delta_{jk} + (1 - A_{ik} - \delta_{ik}) \frac{d_j^\alpha}{\sum_l d_l^\alpha}. \quad (2.35)$$

Local search is a type of degree-biased random walk, where the parameter  $\alpha$  is a measure of the bias towards high-degree nodes. When  $\alpha = 0$ , we recover an unbiased random walk.

*Shortest-path routing* is another type of Markovian routing. In it, a walker that enters the network at  $i$  chooses a destination  $j$  uniformly at random from all nodes

---

<sup>12</sup>If the network is directed, we replace  $d_k$  by the out-degree  $d_k^{\text{out}}$ .

other than  $i$  and traverses a shortest-path, chosen uniformly at random from all shortest paths. In a network in which there is a unique shortest path between any two nodes, the transition matrices are binary matrices, where  $P_{ij}^{(k)} = 1$  if  $(i, j) \in E$  and  $(i, j)$  belongs to the shortest path from  $i$  to  $k$ . When multiple shortest paths exist from  $i$  to  $k$ , the  $(i, j)$  entry is  $P_{ij}^{(k)} = s_{ij}^{(k)} / \sum_{j'} s_{ij'}^{(k)}$ , where  $s_{ij'}^{(k)}$  is the number of shortest paths from  $i$  to  $k$  that traverse the directed edge  $(i, j')$  if  $(i, j') \in E$  and 0 otherwise. The effectiveness of betweenness centrality coincides with the geodesic betweenness centrality for shortest-path routing.

One can introduce heterogeneity to the traffic-dynamics model to make it more realistic. For example, one can consider heterogeneous packet generation [64] (with packets generated only at certain nodes) or heterogeneous node capacities (for example, with some nodes having much higher capacities than other nodes) [103, 111].

## 2.2.2 Strategies to increase the traffic capacity

In the traffic-dynamics model, the traffic capacity  $\rho_c$  is an important measure of congestion of a network, and a vast literature is dedicated to methods to increase  $\rho_c$  [22, 23, 67, 75, 103, 110, 111]. These methods fall mainly into two categories: ‘soft’ strategies, in which one aims to find better routing strategies; and ‘hard’ strategies, in which one aims to change the underlying network topology (by adding or removing edges or nodes). See Chen *et al.* in [17] for a survey of both strategies. For example, one type of ‘soft’ strategy is traffic-aware routing [22, 23, 103, 111], in which the routing mechanism takes the number of walkers at neighbouring nodes into account, so walkers avoid congested nodes. ‘Hard’ strategies include removing the edges that are incident to high-degree nodes [67, 75, 110], which often become congested, or adding edges between low-degree nodes [39], which act as shortcuts. In shortest-path routing, these shortcuts reduce the maximum effective betweenness  $\max_j B_j^{\text{eff}}$ , and they therefore reduce  $\rho_c$ .

Some researchers also attempted to identify networks with a fixed number of edges that maximize  $\rho_c$  [21, 35]. The networks that were found have been called *entangled network* and tend to have a very homogeneous structure, as their degree, betweenness, node distance, and loop distributions are very narrow [21].

## 2.3 Queueing networks

A natural approach for modelling congestion on networks is using queueing networks, which is a continuous-time model for congestion on networks. In this approach, each

node represents a service station. Walkers (e.g., representing customers) go from node to node and queue at each node to be served. Queueing networks have been used in numerous applications [12, 32], including waiting rooms in medical offices [97], check-ins at airports [97], manufacturing systems [51], communication networks [12], traffic flow [73], pedestrian modelling [54, 56, 68, 74, 109], and others.

Formally, a *queueing network* [50] (also called a *migration network*) is a network  $G = (V, E)$  in which each node contains a service station<sup>13</sup>, with one or multiple servers, where customers form a queue until being served. At each node, customers arrive either from outside the system or via an incoming edge from a neighbouring node. Customers traverse the network according to a random walk with transition matrix  $\mathbf{P}$ . We assume that  $P_{ij} > 0$  only if  $(i, j) \in E$ . That is, customers can only directly go to another node via an edge. Note that  $(i, j) \in E$  does not imply that  $P_{ij} > 0$ ; it is possible that some edges have zero transition probability. After completing service at node  $i$ , a customer traverses an edge  $(i, j) \in E$  to a random neighbouring node  $j$  with probability  $P_{ij}$ , and it leaves the system with probability  $1 - \sum_k P_{ik}$ . (We assume that each row sum<sup>14</sup> is at most 1.) When customers are allowed to enter and leave the system, it is called an *open* queueing network. When no customers can enter or leave the system, it is a *closed* queueing network, and the number of customers in the system is constant.<sup>15</sup> Queueing networks can have various probability distributions of service times (i.e., *service distributions*), probability distributions of the pattern of arrivals of customers in time (i.e., *arrival processes*), and *service disciplines*, which specify the number of servers and the order of customer service at each node.

We only consider open queueing networks, as the number of customers in a store varies over time. We consider the system at *stationarity* (if such a state exists), as it allows our analysis to be tractable; at stationarity, the distribution of the queue sizes  $(X_1(t), \dots, X_n(t))$  does not change in time. The queue size  $X_i(t)$  of node  $i$  denotes the number of customers who are waiting at node  $i$  (including the one being served) at time  $t$ . We call the queue size distribution at stationarity the *equilibrium distribution* of  $X_i(t)$ . We consider open queueing networks at stationarity in which the equilibrium distribution of queue sizes  $X_1(t), \dots, X_n(t)$  can be written in product form. That is,

$$\mathbb{P}(X_1(t) = k_1, \dots, X_n(t) = k_n) = \prod_{j=1}^n \mathbb{P}(X_j(t) = k_j) \quad (2.36)$$

---

<sup>13</sup>We use the words *node* and *service station* interchangeably when referring to nodes of a queueing network.

<sup>14</sup>Unlike the classic random walk, the row sum is not always equal to 1, because customers leave the system at some nodes (called *sink nodes*, as we define in Section 2.3.1).

<sup>15</sup>We do not consider the case in which customers can enter but not leave, or one in which customers can leave but not enter.

for any non-negative integers  $k_j$ . Equation (2.36) says that the queue sizes  $X_1(t), \dots, X_n(t)$  are independent from one another. We say that open queueing networks that satisfy Equation (2.36) have a *product-form equilibrium distribution*.

### 2.3.1 Open queueing networks

We consider open queueing networks with an underlying  $n$ -node network  $G = (V, E)$  and we suppose that it satisfies the following conditions:

1. **First-in-first-out queue:** Each node is a first-in-first-out (FIFO) service station; in other words, the first customer who arrives at the node gets served first.
2. **Exponential service time:** At each node  $i$ , when there are  $k \geq 1$  customers at  $i$ , the service time (i.e., the time it takes for the first customer in the queue to be finished serving) is an exponential random variable with parameter  $\mu_{ik} > 0$  (called the *service rates*). In other words,

$$\begin{aligned} \mathbb{P}(\text{Server } i \text{ completes a service in } (t, t + \Delta t) \mid X_i(t) = k) \\ = \mu_{ik}\Delta t + o(\Delta t), \end{aligned} \quad (2.37)$$

for all  $i \in \{1, \dots, n\}$  and  $k \geq 1$ . Furthermore, the service time is independent of the service times of all other nodes.

3. **Poisson arrival processes:** Arrivals from outside the system to distinct nodes are independent Poisson processes. The rate of arrival at node  $i$  is a constant  $\lambda_{0i} \geq 0$ . We call any node  $i$  with  $\lambda_{0i} > 0$  a *source node* (or simply a *source*).
4. **Markovian random walk with transition matrix  $P$ :** Customers who depart from nodes travel instantly to other nodes (or they leave the system) with fixed probabilities. The probability that a customer who departs from node  $i$  travels to node  $j$  is  $P_{ij}$ . The probability of leaving the system from node  $i$  is  $1 - \sum_j P_{ij}$ . We call node  $j$  a *sink node* (or simply a *sink*) if  $\sum_k P_{jk} < 1$ ; that is, a node  $j$  is a sink if a non-zero proportion of walkers leaves the system from it.
5. **Sources and sinks:** We assume that there is at least one source and at least one sink.

6. **Reachability:** For each node  $k$ , there exists a source node  $s$  and a sink node  $t$  such that there is a path in  $G$  from  $s$  to  $k$  and a path from  $k$  to  $t$  in  $G$ . Furthermore, both paths must be along edges with positive transition probabilities (i.e.,  $P_{ij} > 0$  for all edges  $(i, j)$  in the paths). We call these conditions the *reachability conditions*. They are conditions on both  $G$  and  $\mathbf{P}$ ; they ensure that each node can receive walkers and that these walkers leave the system eventually through some sink node.
7. **Sufficiently large service rates:** We assume that the service rates  $\mu_{ik}$  (with  $i \in \{1, \dots, n\}$  and  $k \geq 1$ ) are sufficiently large such that nodes can serve customers faster than they arrive. This ensures that a stationary state exists. More precisely, the service rates  $\mu_{ik}$  need to satisfy Equation (2.43), which we discuss later in this section.

With these conditions, the queue sizes  $\mathbf{X}(t) = (X_1(t), \dots, X_n(t))$  form an irreducible, time-homogeneous, continuous-time Markov chain in  $\mathbb{Z}_+^n$ . That is, the state of  $\mathbf{X}(t)$  depends only on the previous state (i.e., the last state that was different from  $\mathbf{X}(t)$ ) and not the prior history (Markov property) or the time  $t$  (time homogeneity). Furthermore,  $\mathbf{X}(t)$  can reach any state in  $\mathbb{Z}_+^n$  from any other state in  $\mathbb{Z}_+^n$  (irreducibility). The networks that satisfy the above properties are also called open *Jackson networks* [45]. See Figure 2.9 for an example of an open queueing network.

Because we consider the system at stationarity, the distributions of  $(X_1(t), \dots, X_n(t))$  do not change in time. We denote the equilibrium distribution of  $X_i(t)$  by  $X_i$  and call it a *stationary state* of the queueing network.<sup>16</sup> If a stationary state exists, it is unique and an irreducible, time-homogeneous Markov chain reaches this state as  $t \rightarrow \infty$  from any initial state [50]. The above conditions ensure that a stationary state exists and that the queueing network has a product-form equilibrium distribution [50, 51].

The parameters of the distribution for the queue sizes  $X_i$  of node  $i$  at stationarity depend on the arrival rate  $\lambda_i$  of customers to  $i$ . We obtain the arrival rates  $\lambda_1, \dots, \lambda_n$  by solving the *traffic equations*

$$\lambda_i = \lambda_{0i} + \sum_{j=1}^n \lambda_j P_{ji}, \quad \text{for } i = 1, \dots, n. \quad (2.38)$$

The traffic equations (2.38) are flow-balance equations and state that the mean departure rate equals the mean arrival rate at any node  $i$  at stationarity. This is a necessary condition at stationary. In matrix form, Equation (2.38) becomes

$$(\mathbf{I} - \mathbf{P}^T) \boldsymbol{\lambda} = \mathbf{b}, \quad (2.39)$$

---

<sup>16</sup>More precisely, it is a stationary state of the queue sizes  $(X_1(t), \dots, X_n(t))$ .



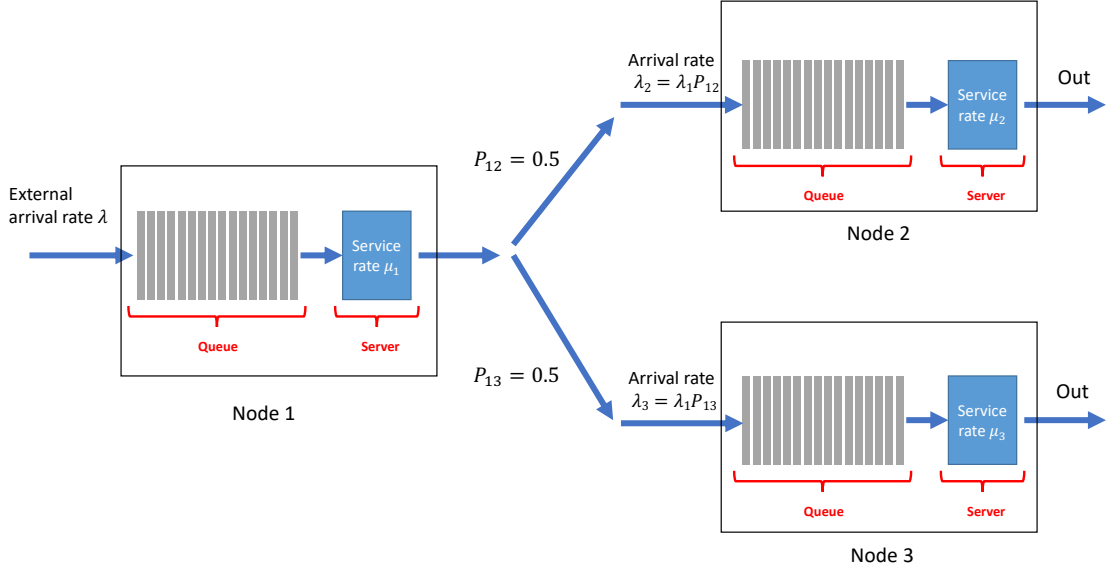


Figure 2.9. Example of an open queueing network with 3 nodes. Node 1 is a source node, and nodes 2 and 3 are sink nodes.

where  $\mathbf{I}$  is the  $n \times n$  identity matrix,  $\mathbf{l} = (\lambda_1, \dots, \lambda_n)^T$ , and  $\mathbf{b} = (\lambda_{01}, \dots, \lambda_{0n})^T$ . The matrix  $\mathbf{I} - \mathbf{P}^T$  is the transpose of a version of the *random-walk normalized Laplacian*. There is a unique, positive solution to the traffic equations (2.38) [50]. Therefore,  $\mathbf{I} - \mathbf{P}^T$  is invertible and

$$\mathbf{l} = (\mathbf{I} - \mathbf{P}^T)^{-1} \mathbf{b}. \quad (2.40)$$

The queue sizes  $X_i$  satisfy Equation (2.36), and the marginal probability  $\mathbb{P}(X_i = k)$  is [50]

$$\mathbb{P}(X_i = k) = \frac{\pi_{i0} \lambda_i^k}{\mu_{i1} \mu_{i2} \cdots \mu_{ik}}, \quad \text{for } k = 1, 2, \dots, \quad (2.41)$$

with

$$\pi_{i0} = \mathbb{P}(X_i = 0) = \left( 1 + \sum_{k=1}^{\infty} \frac{\lambda_i^k}{\mu_{i1} \cdots \mu_{ik}} \right)^{-1}, \quad (2.42)$$

where we assume that

$$\sum_{k=1}^{\infty} \frac{\lambda_i^k}{\mu_{i1} \cdots \mu_{ik}} < \infty, \quad \text{for } i = 1, \dots, n. \quad (2.43)$$

Equation (2.43) is satisfied when  $\mu_{ik}$  is sufficiently large. (If Equation (2.43) is not satisfied, an equilibrium distribution does not exist [51], as there is a queue where walkers enter more quickly than they leave.) Equation (2.43) implies that the arrival

rate  $\lambda_i$  cannot exceed some value  $U_i$ , where

$$U_i = \sup \left\{ \lambda_i \geq 0 : \sum_{k=1}^{\infty} \frac{\lambda_i^k}{\mu_{i1} \cdots \mu_{ik}} < \infty \right\}. \quad (2.44)$$

Therefore, if  $\lambda_i < U_i$  for all  $i$ , we are guaranteed to have a stationary state.<sup>17</sup>

The service discipline does not need to be FIFO for Equations (2.36), (2.41), and (2.42) to hold. It is sufficient to have any service discipline that ensures that Equation (2.37) is satisfied. Equation (2.37) states that the time until the next customer departs from the node (with no selection from among the waiting customers) is distributed exponentially. For example, both *last-in-first-out* (LIFO) queueing, in which the last customer to arrive gets served first, and *processor sharing* queueing, in which all customers at node  $i$  are served at the same rate  $\mu_{ik}/k$  (where  $k$  is the number of customers at  $i$ ), satisfy Equation (2.37).

### Single-server queues

If each node  $i$  is a queue with a single server with service rate  $\mu_i$ , then  $\mu_{ik} = \mu_i$ . In this case,  $U_i = \mu_i$ . If  $\lambda_i < \mu_i = U_i$  (so Equation (2.43) is satisfied),  $X_1, \dots, X_n$  are independent geometric random variables, with

$$\mathbb{P}(X_i = k) = \left(1 - \frac{\lambda_i}{\mu_i}\right) \left(\frac{\lambda_i}{\mu_i}\right)^k, \quad \text{for } k = 0, 1, \dots \quad (2.45)$$

The mean queue lengths are

$$\mathbb{E}[X_i] = \frac{\lambda_i}{\mu_i - \lambda_i}, \quad \text{for all } i. \quad (2.46)$$

For most of the thesis, we use single-server queues.

### Multi-server queues

If each node  $i$  has  $s \geq 1$  servers, each with service rate  $\mu_i$ , then  $\mu_{ik} = \min(k, s)\mu_i$ . In this case,  $X_1, \dots, X_n$  are independent random variables whose marginal distributions are given by Equation (2.41).

For example, when  $s = 2$ , the distribution of  $X_i$  is

$$\mathbb{P}(X_i = k) = \begin{cases} \frac{1-\rho_i}{1+\rho_i}, & \text{for } k = 0, \\ \left(\frac{1-\rho_i}{1+\rho_i}\right) 2\rho_i^k, & \text{for } k = 1, 2, \dots, \end{cases} \quad (2.47)$$

---

<sup>17</sup>When  $\lambda_i = U_i$ , it may or may not have a stationary state, depending on whether Equation (2.43) is satisfied. For example, if  $\mu_{ik} = \mu$  for some  $\mu > 0$  (i.e., single-server queue), there does not exist an equilibrium distribution when  $\lambda_i = U_i = \mu$ . If  $\mu_i k$  satisfies  $\prod_{l=1}^k \mu_{il} = k^2/2^{2k}$  for every  $k \geq 1$ , one can show that Equation (2.43) is satisfied for  $\lambda_i \leq U_i = 1/4$ .

where  $\rho_i = \lambda_i / (2\mu_i)$  and we assume that  $\rho_i < 1$  [97]. The quantity  $\rho_i$  is also called the *utilization* of node  $i$ . The mean queue size for a two-server queue is

$$\mathbb{E}[X_i] = \frac{2\rho_i}{1 - \rho_i^2}, \quad \text{for all } i. \quad (2.48)$$

### 2.3.2 Little's Law

Little's Law, first formulated by John Little in a 1961 publication [65], is one of the most general theorems in queueing theory. It states a relationship between three quantities in a queueing system at stationarity: the mean number  $L$  of customers, the arrival rate  $\lambda$  of customers to the system, and the mean time  $W$  spent by customers in the system. Little's Law states that

$$L = \lambda W. \quad (2.49)$$

For example, in a single-server queue with exponential service rate  $\mu$  and constant Poisson arrival at rate  $\lambda$ , the queue length at equilibrium is distributed geometrically with a mean queue length  $L$  [97] of

$$L = \frac{\lambda}{\mu - \lambda}. \quad (2.50)$$

The quantity  $W$  represents the customer *waiting time* (also known as a *sojourn time*), which is the time between the customer joining and leaving a queueing system. In other words, the waiting time is the total time that a customer spends queueing and being served. Using Little's Law (2.49) and Equation (2.50), we find that the mean customer waiting time  $W$  in a single-server queue is

$$W = \frac{1}{\mu - \lambda}. \quad (2.51)$$

Little's Law holds under very general assumptions and holds for any arrival-process distributions, any service distributions, and any service disciplines. The result also applies to a system of systems, such as a queueing network [50]. Little's Law then indicates that the mean number of customers in a system is directly proportional to the mean time spent in it. If we wish to minimize the mean time that a customer spends in a system, we can do so by minimizing the mean number of customers (which is the total mean queue size  $Q$  in queueing networks) in the system. Therefore, we use  $Q$  as our congestion measure in this thesis.

### 2.3.3 Multi-class open queueing networks

In a supermarket, there are different types of customers and shopping visits, depending on the purpose of their visit (also called a *customer mission* [34]), their demographics, and other factors. Each type of customer can have a different transition matrix. We can model this situation using a *multi-class open queueing network*, which have  $M \geq 2$  customer classes. A customer from a class  $c \in \{1, \dots, M\}$  enters a network according to the arrival rates  $\lambda_{0i}^{(c)}$  (with  $i \in \{1, \dots, n\}$ ) and traverses the network with transition matrix  $\mathbf{P}^{(c)}$ .

We assume that the arrival process for customers of each class  $c$  to a node  $i$  is a Poisson process with rate  $\lambda_{0i}^{(c)}$ , independent of all other arrival processes (from either the same class or from other classes). We further assume that each node contains a single server with FIFO service discipline. The server does not distinguish between customers of different classes and serves the first customer at an exponential rate  $\mu_{ik}$  when there are  $k$  customers at the node.

Such a multi-class queueing network falls into the class of *BCMP networks* [10, 32] (named after the four authors F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacios, who first described this network), which have a product-form equilibrium distribution. For each class  $c$ , we solve a separate set of traffic equations

$$\lambda_i^{(c)} = \lambda_{0i}^{(c)} + \sum_{j=1}^n \lambda_j^{(c)} P_{ji}^{(c)}, \quad \text{for all } i = 1, \dots, n \quad (2.52)$$

to determine the arrival rate  $\lambda_i^{(c)}$  of customers of class  $c$  to node  $i$  at equilibrium. The total arrival rate  $\lambda_i$  to node  $i$  is then

$$\lambda_i = \sum_{c=1}^M \lambda_i^{(c)}, \quad \text{for } i = 1, \dots, n, \quad (2.53)$$

assuming that the service rates  $\mu_{ik}$  satisfy Equation (2.43) [32]. In this setting, the queue sizes  $X_1, \dots, X_n$  are independent and satisfy the same equations as in the single-class case (see Equations (2.36), (2.41), and (2.42)) [32].

These results can be generalized further to allow different types of service discipline, different service distributions (which may depend on the type of customer), and other variations. See Chapter 3 of [51] and Chapter 4 of [32] for more details.

### 2.3.4 Comparison between the traffic-dynamics model and the open-queueing-network model

We compare the traffic-dynamics model with Markovian routing (see Section 2.2) with the open-queueing-network model that we discussed earlier in this section. Below we

establish the relationship between the traffic capacity  $\rho_c$  (our performance measure in the traffic-dynamics model) and the maximum arrival rate  $\lambda_{\max}$  to a node.

One can view the traffic-dynamics model with Markovian routing as a discrete approximation of an open queueing network with  $n$  customer classes; we group customers by their designated destination node, so there are  $n$  customer classes in total. The main difference is in the distribution of the service time, which is deterministic in the traffic-dynamics model but stochastic in the queueing-network model. The queueing behaviour in the traffic-dynamics model is a discrete approximation of a FIFO single-server queue with service rate  $\mu = C$ , where  $C$  is the delivering capacity in the traffic-dynamics model. The corresponding open-queueing-network model has  $n$  customer classes, with one class for each destination node  $c$ . Walkers of class  $c$  traverse the network according to the Markovian transition matrix  $\mathbf{P}^{(c)}$  (see Section 2.2) until they reach node  $c$ . For each customer class  $c \in \{1, \dots, n\}$ , the associated external arrival rate is  $\lambda_{0i}^{(c)} = \rho/(n-1)$  for all  $i \neq c$  and 0 otherwise.

The arrival rate  $\lambda_i$  to each node, which we calculate using the multi-class traffic equations (see Equations (2.52) and (2.53)), is related to the effective betweenness  $B_i^{\text{eff}}$  (see Equation (2.27)) in the traffic-dynamics model by

$$\lambda_i = \frac{\rho B_i^{\text{eff}}}{n-1}. \quad (2.54)$$

For a fixed  $\rho$ , we have  $B_{\max}^{\text{eff}} = \lambda_{\max}(n-1)/\rho$ , where  $\lambda_{\max}$  is the maximum arrival rate. By Equation (2.28), increasing the traffic capacity  $\rho_c$  is equivalent to decreasing the maximum effective betweenness  $B_{\max}^{\text{eff}}$ . Therefore, increasing  $\rho_c$  is equivalent to decreasing  $\lambda_{\max}$  when the (homogeneous) external arrival rate is fixed. One can therefore construe the maximum arrival rate  $\lambda_{\max}$  as a measure of congestion that is derived from the traffic-dynamics model. Furthermore, as we show in Section 4.3.2, when we minimize  $Q$  for sufficiently small homogeneous service rate  $\mu$ , we also minimize  $\lambda_{\max}$ . We use some of the ‘hard’ methods (i.e., adding and deleting edges) for increasing  $\rho_c$  (or, equivalently, decreasing  $\lambda_{\max}$ ) in Section 4.4 and compare their performance in decreasing  $\lambda_{\max}$  with that of our proposed algorithms. We also use  $\lambda_{\max}$  as an objective function to minimize in Chapter 6.

## 2.4 Human-mobility models

Understanding how and why humans move is important for numerous applications, such as traffic forecasting [82], urban planning [38], epidemic modelling [96, 100], and more [7]. With the growing abundance of location data for pedestrians, cars,

airlines, and other entities, scientists are increasingly able to quantitatively study both individual and collective mobility patterns. We focus on collective mobility patterns (i.e., looking at aggregate mobility flow instead of individual trajectories), as congestion is a phenomenon that arises from collective mobility flow. To do this, we study and apply population-level human-mobility models, which try to describe the mobility flow  $T_{ij}$  between locations  $i$  and  $j$  (e.g., the number of individuals who travel from  $i$  to  $j$  in one time unit), where  $i \neq j$ . We can write the flow as a matrix  $\mathbf{T}$ , whose off-diagonal entry  $T_{ij}$  denote the flow of each *origin-destination* (OD) pair  $(i, j)$ . We set its diagonal entries to  $T_{ii} = 0$ . The matrix  $\mathbf{T}$  is also called the *origin-destination (OD) matrix*. In our discussion in this section, we treat  $\mathbf{T}$  as a matrix that records the mobility flow of people and we say  $T_{ij}$  is the number of *origin-destination trips* (or simply *trips*) from  $i$  to  $j$ . In general, the mobility flow is not restricted to the flow of people, but it can also be flow of other entities such as goods, vehicles, or animals.

We are interested primarily in simple mobility models that yield an OD matrix  $\mathbf{T}^{\text{model}}$ , such that  $\mathbf{T}^{\text{model}}$  is as “close” as possible to an empirical OD matrix  $\mathbf{T}^{\text{data}}$ . (We discuss diagnostics for comparing  $\mathbf{T}^{\text{model}}$  and  $\mathbf{T}^{\text{data}}$  in Section 2.4.4.) The models should be simple to avoid overfitting; models should not only fit the data well, but they should also yield good estimates on unseen data. One can then use simple models to estimate or forecast mobility flow in different scenarios.

One can classify most population-level human-mobility models as falling into one of two paradigms [7]: *gravity models* and *intervening-opportunities models*. In a gravity model [16, 26, 102, 113], one assumes that the mobility flow depends on the populations at the origin and destination and that it decreases with the distance between two locations. In an intervening-opportunities (IO) model [95], distance only plays a surrogate role, in that it increases the number of intervening opportunities, which are opportunities at intermediate locations that are more desirable than the destination. It is the number of intervening opportunities between two locations (as well as the populations at the origin and destination) that determines mobility flow.

In this section, we present models that belong to these two paradigms. We also discuss several existing goodness-of-fit measures and parameter-calibration methods. Finally, we present a modelling framework for using and fitting mobility models. We apply this framework when modelling mobility flow in a supermarket in Chapter 5. For more details on human-mobility models, see the review by Barbosa *et al.* [7].

### 2.4.1 Gravity models

Gravity models take their inspiration from Newton’s law of gravity. In the simplest form of such a model, the mobility flow is

$$T_{ij}^{\text{model}} = m_i m_j f(d_{ij}) \quad (2.55)$$

for each OD pair  $(i, j)$ , where  $m_i$  is the origin population,  $m_j$  is the destination population,  $d_{ij}$  is the distance between  $i$  and  $j$ , and  $f(d_{ij})$  is a decreasing function of  $d_{ij}$ . Often called a “deterrence function” [92],  $f(d_{ij})$  is usually chosen as either an exponential function  $f(d_{ij}) = e^{-\gamma d_{ij}}$  or a power-law function  $f(d_{ij}) = d_{ij}^{-\gamma}$ , where  $\gamma > 0$  is a fitting parameter [7].

There are several variants of gravity models, including ones in which the  $m_i$  and  $m_j$  are replaced by tunable functions of other input data, such as socio-economic characteristics (e.g., when considering commuting flow between regions) or economic strength (e.g., when considering trade flow between countries). In our application (see Chapter 5), we use a variant in which  $m_i$  and  $m_j$  are replaced by  $O_i^{\text{data}}$  and  $D_j^{\text{data}}$ , respectively, where  $O_i^{\text{data}} = \sum_j T_{ij}^{\text{data}}$  is the empirical number of individuals departing from  $i$  and  $D_j^{\text{data}} = \sum_i T_{ij}^{\text{data}}$  is the empirical number of individuals arriving at  $j$ .

Gravity models have been applied in investigations of human migration [26, 48], cargo-ship movement [49], and inter-city telecommunication flow [55]. They have been criticized for lacking theoretical foundations and for requiring a large number of parameters to fit data in some cases [92]. Furthermore, the conventional gravity model fails to capture mobility flow well in some applications, such as in job seeking [92] or bus and train flows [70], and on some applications on smaller spatial scales [70].

### 2.4.2 Intervening-opportunities models

In intervening-opportunities (IO) models, first proposed by Stouffer in 1940 [95], each location  $i$  has  $m_i$  opportunities, which correspond roughly to its ‘popularity’ or ‘populations’. The key concept of IO models is the notion of *intervening opportunities*. Intervening opportunities are opportunities at intermediate locations that are more desirable than the destination. In IO models, the mobility flow from  $i$  to  $j$  depends on the number  $S_{ij}$  of intervening opportunities (instead of the distance between them) of the OD pair  $(i, j)$  and on the populations of the two zones. A larger number of intervening opportunities of an OD pair  $(i, j)$  entails a smaller mobility flow from  $i$  to  $j$ , because people are more likely to find what they are looking for (or to be diverted) before they reach  $j$ .

In our discussion, we refer to the desirability of a location  $k$  to some origin location  $i$  by the accessibility of  $k$  to  $i$ . The most common definition of accessibility is in terms of a distance to the origin location. More specifically, a location  $k$  is more accessible than location  $j$  to origin  $i$  if  $d_{ik} < d_{ij}$  (i.e.,  $i$  is closer to  $k$  than  $j$ ). In this case, the number of intervening opportunities is

$$S_{ij} = \sum_{\substack{k \neq i \\ d_{ik} < d_{ij}}} m_k. \quad (2.56)$$

For example, in an commuting application, each opportunity may corresponds to a job and the number  $S_{ij}$  of intervening opportunities of the OD pair  $(i, j)$  is equal to the number of jobs at locations that are closer to  $i$  than  $j$ . An IO model states that the more jobs there are at these intermediate locations, the less likely a job seeker will accept a job at location  $j$ . Note that  $S_{ij} \neq S_{ji}$  in general.

IO models and their variants have been used in many applications, including to model intra-city mobility [87], interstate migration [1, 2, 5], riots [19], and the creation of social ties in a city [90].

In all IO models, the OD matrix is given by

$$T_{ij}^{\text{model}} = O_i^{\text{data}} \frac{f(m_i, m_j, S_{ij})}{\sum_k f(m_i, m_k, S_{ik})}, \quad (2.57)$$

where  $O_i^{\text{data}} = \sum_j T_{ij}^{\text{data}}$  is the (typically empirical) number of departures from  $i$  and  $f(m_i, m_j, S_{ij})$  is a model-specific function. In the IO models, one assumes that  $O_i^{\text{data}}$  is known or that one has an estimate for  $O_i^{\text{data}}$  (e.g., a fixed fraction of the population at  $i$ ). IO models are *production-constrained*, in that they satisfy

$$O_i^{\text{model}} := \sum_j T_{ij}^{\text{model}} = O_i^{\text{data}} \quad (2.58)$$

for every location  $i$ . In other words, the number of departing trips at each location  $i$  in the model is the same as in the empirical data. (We describe other constrained versions of mobility models in Section 2.4.3.) In IO models, the term  $f(m_i, m_j, S_{ij})$  can be interpreted as the attraction value of location  $j$  to location  $i$ . Each person that departs from location  $i$  chooses location  $j$  with probability proportional to  $f(m_i, m_j, S_{ij})$ . The quantity  $T_{ij}$  is then the mean number of people who take a trip from  $i$  to  $j$ .

### Stouffer's IO model

In Stouffer's original formulation, the number of individuals who move a given distance is directly proportional to the number of opportunities at that distance and inversely



proportional to the number of intervening opportunities:

$$f(m_i, m_j, S_{ij}) = \frac{m_j}{S_{ij} + c}, \quad (2.59)$$

for some fixed constant  $c$  (to avoid dividing by 0). Note that  $f(m_i, m_j, S_{ij})$  does not depend on  $m_i$  in Stouffer's IO model.

### Schneider's variant of an IO model

An especially popular reformulation of the IO model was proposed by Schneider [89]. In it,

$$f(m_i, m_j, S_{ij}) = e^{-LS_{ij}} - e^{-L(S_{ij}+m_j)}, \quad (2.60)$$

where  $L \in (0, 1)$  is a tunable parameter. We can derive Equation (2.60) as follows. In Schneider's IO model, each person who leaves location  $i$  considers each opportunity in non-decreasing order of distance from  $i$  (i.e., from the nearest location to the furthest one). For simplicity, we assume that there are no equidistant locations. A person accepts the current opportunity with probability  $L$ . Upon accepting an opportunity at  $j$ , a person takes a trip (i.e., moves) from location  $i$  to location  $j$  and thus does not consider any further opportunities. If a person has not accepted any opportunity, the process restarts and the person considers all opportunities again in non-decreasing distance from  $i$ . This process continues until the person accepts an opportunity. We calculate the probability of a person accepting an opportunity at location  $j$  as follows. Let  $R$  be the number of rejected opportunities in the final iteration when an opportunity is accepted. The random variable  $R$  has a truncated geometric distribution. A person who accepts an opportunity at location  $j$  must have rejected all  $S_{ij}$  opportunities at closer locations and accepted one of the  $m_j$  opportunities at  $j$ . In other words, a person in location  $i$  takes a trip to location  $j$  if  $S_{ij} + m_j > R \geq S_{ij}$ . The probability of rejecting at least  $k$  opportunities is

$$\mathbb{P}(R \geq k) \propto (1 - L)^k \approx \exp(-kL), \quad k < N, \quad (2.61)$$

where  $N$  is the total number of opportunities. The approximation becomes exact as  $N$  and  $k$  tend to infinity with  $k/N$  fixed. The probability that a person accepts an opportunity at location  $j$  is thus

$$\begin{aligned} \mathbb{P}(S_{ij} + m_j > R \geq S_{ij}) &= \mathbb{P}(R \geq S_{ij}) - \mathbb{P}(R \geq S_{ij} + m_j) \\ &\approx e^{-LS_{ij}} - e^{-L(S_{ij}+m_j)} \\ &= f(m_i, m_j, S_{ij}). \end{aligned} \quad (2.62)$$

Therefore,  $f(m_i, m_j, S_{ij})$  equals the number of customers who make a trip from  $i$  to  $j$  divided by the number of customers who leave  $i$ . Similar to Stouffer's IO model,  $f(m_i, m_j, S_{ij})$  does not depend on  $m_i$  in Schneider's IO model.

### Radiation model

The radiation model [92], first proposed by Simini *et al.* as an alternative to the gravity model, is a parameter-free variant of the IO model, with

$$f(m_i, m_j, S_{ij}) = \frac{m_i m_j}{(m_i + S_{ij})(m_i + m_j + S_{ij})}. \quad (2.63)$$

The model is derived by assuming that each person who leaves  $i$  considers every opportunity in all locations, including the opportunities in  $i$ . The walker assigns a benefit score  $z$ , which is drawn from a continuous distribution  $p(z)$ , to each opportunity. Different walkers assign different benefit scores to the same opportunity, but the scores are drawn from a common distribution  $p(z)$ . While not explicitly mentioned in [92], Simini *et al.* assumed that  $p(z) < \infty$  for all  $z$ , so with probability 1 (i.e., *almost surely*), no two opportunities have the same benefit score. A person travels to the closest destination  $j$  that has an opportunity with a benefit score that is larger than the largest benefit score among the opportunities at  $i$ . We assume for simplicity that there are no equidistant destinations. Simini *et al.* showed that the probability of a walker at  $i$  travelling to  $j$  is independent of the distribution  $p(z)$  and is given by the right-hand side of Equation (2.63).

The radiation model has been used to study commuting flows [91, 92], human migration [58], mobile-phone calling [92], and freight transport [92]. The radiation model has no parameters, so no parameter calibration is required. This is advantageous in applications in which no mobility flow data is available, as it is typically not possible to calibrate parameters in such a setting. (One notable exception is the extended radiation model [107] that we will present shortly.) However, recent work suggests that the radiation model is unable to describe human mobility on small spatial scales [60, 63, 70].

### Extended radiation model

Yang *et al.* [107] proposed an extension of the radiation model that has an additional dimensionless parameter  $\alpha$ , which one can calibrate to adapt the model for different spatial scales.

In this model,

$$f(m_i, m_j, S_{ij}) = \frac{[(m_i + S_{ij} + m_j)^\alpha - (m_i + S_{ij})^\alpha] (m_i^\alpha + 1)}{((m_i + S_{ij})^\alpha + 1) [(m_i + S_{ij} + m_j)^\alpha + 1]}. \quad (2.64)$$

Yang *et al.* claimed that this model fits an empirical OD matrix  $\mathbf{T}^{\text{data}}$  better for intra-city commuting flow. (They tested the extended radiation model on commuting-flow data for Western United States, the San Francisco bay area, and San Francisco.) Furthermore, they reported an empirical relationship between  $\alpha$  and the spatial scale<sup>18</sup>  $l$  (in kilometres):

$$\alpha = \left( \frac{l}{36} \right)^{1.33}. \quad (2.65)$$

Using this relationship, one can calibrate  $\alpha$  without mobility-flow data. When  $\alpha = 1$ , we recover a slightly altered version of Equation (2.63), with every occurrence of  $m_i$  replaced by  $m_i + 1$ .

### 2.4.3 Constrained versions of mobility models

Given data on  $O_k^{\text{data}}$  and/or  $D_k^{\text{data}}$  (the total number of departing trips and/or arriving trips at each location), we can cast each mobility model into a form such that

$$\sum_j T_{ij}^{\text{model}} = O_i^{\text{data}}, \quad \text{for all } i, \quad (2.66)$$

and/or

$$\sum_i T_{ij}^{\text{model}} = D_j^{\text{data}}, \quad \text{for all } j. \quad (2.67)$$

In other words, the row and/or columns sum of  $\mathbf{T}^{\text{model}}$  and  $\mathbf{T}^{\text{data}}$  coincide. There are three different constrained versions: *production-constrained*, *attraction-constrained*, and *doubly-constrained*. They were introduced originally for the gravity models [101], but we can also analogously derive constrained variants of IO models. A constrained model takes a matrix of attraction values  $\mathbf{f}$  (called the *attraction matrix*) with entries  $f_{ij}$  as input, together with the vector  $O_k^{\text{data}}$  and/or the vector  $D_k^{\text{data}}$ . If only  $O_k^{\text{data}}$  was supplied, it is a production-constrained model, which outputs an OD matrix  $\mathbf{T}^{\text{model}}$  that satisfies Equation (2.66). If only  $D_k^{\text{data}}$  was supplied, it is a attraction-constrained model, which outputs an OD matrix  $\mathbf{T}^{\text{model}}$  that satisfies Equation (2.67). If both  $O_k^{\text{data}}$  and  $D_k^{\text{data}}$  were supplied, it is a doubly-constrained model, which outputs an  $\mathbf{T}^{\text{model}}$  that satisfies both Equation (2.66) and Equation (2.67). In the gravity models, we take  $f_{ij} = f(d_{ij})$  (where  $f$  is the deterrence function); in IO models,  $f_{ij} = f(m_i, m_j, S_{ij})$ .

<sup>18</sup>In [107], the authors divided the space into squares with side length  $l$ .

**Production-constrained models.** In a *production-constrained* model (also called a *singly-constrained model*), we constrain our model such that the number of people starting a trip at  $i$  is fixed to a known quantity  $O_i^{\text{data}}$  (e.g., from census data):

$$T_{ij}^{\text{model}} = \begin{cases} O_i^{\text{data}} \frac{f_{ij}}{\sum_k f_{ik}}, & \text{if } \sum_k f_{ik} > 0, \\ 0, & \text{otherwise.} \end{cases} \quad (2.68)$$

Production-constrained models are useful when we know the number of trips that start at each location. For example, we can use such a model to estimate which local supermarket(s) customers go for their regular supermarket shopping and hence estimate the total number of visits at each supermarket, as the number of people at each location is known (or can be approximated by the population of the location). Versions of the production-constrained gravity model (e.g., Huff Gravity Model [40] or Reilly's Law of Retail Gravitation [86]) have been used for this application. Another example is modelling commuter flow when the number of commuters at each location is known. The radiation model (which is a production-constrained model) was introduced for this application [92].

We can define

$$P_{ij} = \begin{cases} \frac{f_{ij}}{\sum_k f_{ik}}, & \text{if } \sum_k f_{ik} > 0, \\ 0, & \text{otherwise,} \end{cases} \quad (2.69)$$

and view it as an independent probability of moving to  $j$  for each person at  $i$ . In such a model, there are  $O_i^{\text{data}}$  people who are taking a trip starting at  $i$ ; each of them chooses a destination  $j$  with probability  $P_{ij}$ . Then  $T_{ij} = O_i^{\text{data}} P_{ij}$  and is equal to the mean number of trips from  $i$  to  $j$ .

The OD matrix  $\mathbf{T}^{\text{model}}$  in production-constrained models is invariant under the scaling  $f_{ij} \mapsto c_i f_{ij}$  for  $c_i > 0$  (i.e., multiplying each row  $i$  of  $\mathbf{f}$  by  $c_i$ ), as the  $c_i$  factors cancel each other in Equation (2.68).

**Attraction-constrained models.** In an *attraction-constrained* model, we constrain our model such that the number of people ending a trip at  $i$  is fixed to a known quantity  $D_j^{\text{data}}$ :

$$T_{ij}^{\text{model}} = \begin{cases} D_j^{\text{data}} \frac{f_{ij}}{\sum_l f_{lj}}, & \text{if } \sum_l f_{lj} > 0, \\ 0, & \text{otherwise.} \end{cases} \quad (2.70)$$

This type of model is useful for applications in which we know the number of arrivals at each destination, e.g., because there is a fixed, known capacity at each destination. Examples include sold-out sports events or music events (under the assumption that everybody who buys a ticket actually attends the event). We can use

an attraction-constrained model to estimate the origin locations of visitors to such events.

Similar to production-constrained models, we can define

$$P_{ij} = \begin{cases} \frac{f_{ij}}{\sum_l f_{lj}}, & \text{if } \sum_l f_{lj} > 0, \\ 0, & \text{otherwise.} \end{cases} \quad (2.71)$$

and view  $P_{ij}$  as an independent probability of one arrival at  $j$  to come from  $i$ , where there are  $D_j^{\text{data}}$  total arrivals at  $j$ . Then  $T_{ij} = D_j^{\text{data}} P_{ij}$  and is the mean number of trips from  $i$  to  $j$ .

Note that the IO models are originally derived as production-constrained models, i.e., under the assumption that  $\mathbf{T}^{\text{model}}$  satisfies Equation (2.66) and that  $f_{ij}$  is the proportion of customers at  $i$  that take a trip to  $j$ . However, an attraction-constrained IO model no longer satisfies Equation (2.66) (in general) and the quantity  $f_{ij}$  no longer corresponds to the proportion of customers at  $i$  that take a trip to  $j$ . We therefore lose the interpretability of the model. For example, we derived  $f_{ij}$  in Schneider's IO model as the proportion of customers at  $i$  that go to  $j$  by assuming that each person at  $i$  goes through the process of considering opportunities in non-decreasing distance and goes to the closest opportunity that they accept (see Section 2.4.2). In an attraction-constrained Schneider IO model,  $f_{ij}$  is not equal to the proportion of customers at  $i$  that go to  $j$ . Therefore, when using the attraction constrained Schneider IO model, we can no longer say that each person at  $i$  goes through the process of considering opportunities in non-decreasing distance. (Instead, we are only using the functional form of  $f_{ij}$  and we ignore how  $f_{ij}$  was derived.)

The OD matrix  $\mathbf{T}^{\text{model}}$  in attraction-constrained models is invariant under the scaling  $f_{ij} \mapsto c_j f_{ij}$  for  $c_j > 0$  (i.e., multiplying each column  $j$  of  $\mathbf{f}$  by  $c_j$ ).

**Doubly-constrained models.** In a *doubly-constrained* model, we incorporate both production constraints and attraction constraints, so both Equation (2.66) and Equation (2.67) are satisfied. For each OD pair, the OD matrix has entries

$$T_{ij}^{\text{model}} = (A_i O_i^{\text{data}}) \times (B_j D_j^{\text{data}}) \times f_{ij}, \quad (2.72)$$

where  $A_i$  and  $B_j$  are scaling factors (also called “balancing factors”) such that Equation (2.66) and Equation (2.67) are satisfied. That is,

$$O_i^{\text{data}} = \sum_j T_{ij}^{\text{model}} = \sum_j A_i O_i^{\text{data}} B_j D_j^{\text{data}} f_{ij}, \quad (2.73)$$

$$D_j^{\text{data}} = \sum_i T_{ij}^{\text{model}} = \sum_i A_i O_i^{\text{data}} B_j D_j^{\text{data}} f_{ij}. \quad (2.74)$$

Rearranging Equations (2.73) and (2.74) yields

$$A_i = \left( \sum_j B_j D_j^{\text{data}} f_{ij} \right)^{-1}, \quad (2.75)$$

$$B_j = \left( \sum_i A_i O_i^{\text{data}} f_{ij} \right)^{-1}, \quad (2.76)$$

where we assume that  $\sum_j B_j D_j^{\text{data}} f_{ij} > 0$  and  $\sum_i A_i O_i^{\text{data}} f_{ij} > 0$ . The parameters  $A_i$  and  $B_j$  are usually calibrated with an *iterative-proportional-fitting* (IPF) procedure [20], which performs the following steps:

Step 1: Initialize  $A_i$  to some arbitrary positive values (e.g.,  $A_i = 1$  for all  $i$ ).

Step 2: Calculate  $B_j$  using Equation (2.76) from  $A_i$ , followed by an update of  $A_i$  using Equation (2.75).

Step 3: Repeat Step 2 until the values on the right-hand side of Equations (2.73) and (2.74) are close (e.g., within 1%) to the values on the left-hand side.

The IPF procedure is guaranteed to converge under the following two conditions: (1)  $O_k^{\text{data}}$  and  $D_k^{\text{data}}$  are positive for all  $k$ ; (2)  $\mathbf{f}$  has non-negative entries and it has no 0 rows and columns [84].

One uses doubly-constrained models when one knows both the out-flow  $O_k^{\text{data}}$  and the in-flow  $D_k^{\text{data}}$  at each location  $k$ . For example, doubly-constrained models have been used in transportation studies [29] in which one assumes that the number of departures and arrivals at each location is known. In Chapter 5, we use doubly-constrained models to model the mobility flow of customers between zones in a supermarket, as we know the number of arrivals and departures from each zone from shopping data.

Similar to attraction-constrained IO models, we also lose the interpretability of doubly-constrained IO models. Note that the OD matrix  $\mathbf{T}^{\text{model}}$  in doubly-constrained models is invariant under the scaling  $f_{ij} \mapsto c_i f_{ij}$  for  $c_i > 0$ , as the calibration parameter  $A_i$  scales as  $A_i \mapsto A_i / c_i$  under this scaling. Similarly, the OD matrix  $\mathbf{T}^{\text{model}}$  is invariant under the scaling  $f_{ij} \mapsto c_j f_{ij}$  for  $c_j > 0$ , as the calibration parameter  $B_j$  scales as  $B_j \mapsto B_j / c_j$  under this scaling.

#### 2.4.4 Goodness-of-fit measures

Mobility models aim to estimate or forecast mobility flow, which is described by an empirical OD matrix  $\mathbf{T}^{\text{data}}$ . To compare the model estimate  $\mathbf{T}^{\text{model}}$  with  $\mathbf{T}^{\text{data}}$ , we require a goodness-of-fit measure, which quantifies how ‘close’ these two OD matrices are. We present several goodness-of-fit measures below.

### Common part of commuters (CPC)

The *common part of commuters* (CPC) score is the proportion of trips that the OD matrices  $\mathbf{T}^{\text{data}}$  and  $\mathbf{T}^{\text{model}}$  have in common:

$$\text{CPC}(\mathbf{T}^{\text{data}}, \mathbf{T}^{\text{model}}) = \frac{\sum_{i,j} \min\{T_{ij}^{\text{data}}, T_{ij}^{\text{model}}\}}{\sum_{i,j} \frac{1}{2}(T_{ij}^{\text{data}} + T_{ij}^{\text{model}})}. \quad (2.77)$$

The CPC score is based on the Sørensen similarity index (SSI)<sup>19</sup> [94], and it varies from 0 (when there is no agreement between the model and data) to 1 (when  $\mathbf{T}^{\text{data}}$  and  $\mathbf{T}^{\text{model}}$  are identical). In production-constrained or doubly-constrained models,  $\sum_{i,j} T_{ij}^{\text{data}} = \sum_{i,j} T_{ij}^{\text{model}}$ , so we can interpret the CPC score as the fraction of people whose trip is assigned correctly by the model. In this case, we can simplify Equation (2.77) to obtain

$$\text{CPC}(\mathbf{T}^{\text{data}}, \mathbf{T}^{\text{model}}) = \frac{\sum_{i,j} \min\{T_{ij}^{\text{data}}, T_{ij}^{\text{model}}\}}{N}, \quad (2.78)$$

where  $N = \sum_{i,j} T_{ij}^{\text{data}}$  is the total number of trips.

The CPC score was introduced in [31, 60] and has been used in studies of human mobility [59, 72, 107]. In past studies, the CPC score gave similar results to the other goodness-of-fit measures that we describe in this section when comparing the performance of mobility models [59, 72]. It also appears to be the one of the most popular goodness-of-fit measure in recent studies.

### Normalized root-mean-square error (NRMSE)

The *normalized root-mean-square error* (NRMSE) is defined by

$$\text{NRMSE}(\mathbf{T}^{\text{data}}, \mathbf{T}^{\text{model}}) = \frac{\sum_{i,j} (T_{ij}^{\text{data}} - T_{ij}^{\text{model}})^2}{N}. \quad (2.79)$$

### Information-gain statistic

The *information-gain statistic* is an information-theoretic measure and is defined by [59]

$$I(\mathbf{T}^{\text{data}}, \mathbf{T}^{\text{model}}) = \sum_{i,j} \frac{T_{ij}^{\text{data}}}{N} \ln \left( \frac{T_{ij}^{\text{data}}}{T_{ij}^{\text{model}}} \right). \quad (2.80)$$

---

<sup>19</sup>Some studies call the CPC score simply the SSI (e.g., [66]).

## Correlation coefficients

We can also use measures of correlation (in the form of correlation coefficients) between  $\mathbf{T}^{\text{data}}$  and  $\mathbf{T}^{\text{model}}$ . Such measures treat the OD matrices as vectors  $\mathbf{X} = (x_1, \dots, x_Z)$  and  $\mathbf{Y} = (y_1, \dots, y_Z)$ , where  $Z = n^2 - n$  and we obtain the vectors  $(x_1, \dots, x_Z)$  and  $(y_1, \dots, y_Z)$  by concatenating<sup>20</sup> the off-diagonal entries of  $\mathbf{T}^{\text{data}}$  and  $\mathbf{T}^{\text{model}}$ , respectively. We give the formulas for three of the best-known correlation coefficient measures.

The *Pearson product-moment correlation coefficient*  $r_{XY}$  (or simply the *Pearson coefficient*) between  $\mathbf{X}$  and  $\mathbf{Y}$  is defined by

$$r_{XY} = \frac{\sum_k (X_k - \bar{X})(Y_k - \bar{Y})}{\sqrt{\sum_k (X_k - \bar{X})^2} \sqrt{\sum_k (Y_k - \bar{Y})^2}}, \quad (2.81)$$

where  $\bar{X} = 1/Z \sum_{k=1}^Z X_k$  and  $\bar{Y} = 1/Z \sum_{k=1}^Z Y_k$  are the mean values of the vectors  $\mathbf{X}$  and  $\mathbf{Y}$ , respectively.

We define the *Spearman rank-correlation coefficient*  $r_s$  between  $\mathbf{X}$  and  $\mathbf{Y}$  as the Pearson product-moment correlation coefficient between the rank variables  $\text{rg}_X$  and  $\text{rg}_Y$ , which are the vectors that we obtain after applying the rank function to  $\mathbf{X}$  and  $\mathbf{Y}$ . In other words,

$$r_s = r_{\text{rg}_X, \text{rg}_Y}. \quad (2.82)$$

The *rank function* maps each entry of a vector to their rank (in increasing order) in the vector. If there are any ties, the rank of the tied values is the mean of the ranks. For example, the vector (1.2, 10.2, 3.2, 3.2) is mapped to (1, 4, 2.5, 2.5).

The *Kendall tau rank-correlation coefficient*  $\tau_K$  between  $\mathbf{X}$  and  $\mathbf{Y}$  is defined in terms of the number  $n_c$  of *concordant* pairs and the number  $n_d$  of *discordant* pairs. A pair of two-tuples  $(x_i, y_i)$  and  $(x_j, y_j)$  for  $i < j$  is concordant, if both  $x_i > x_j$  and  $y_i > y_j$  or if both  $x_i < x_j$  and  $y_i < y_j$ . They are discordant, if both  $x_i > x_j$  and  $y_i < y_j$  or if both  $x_i < x_j$  and  $y_i > y_j$ . If  $x_i = x_j$  or  $y_i = y_j$ , then the pair is said to be *tied*. When there are no ties, then  $\tau_K$  is defined by

$$\tau_K = \frac{n_c - n_d}{n_0}, \quad (2.83)$$

where  $n_0 = Z(Z - 1)/2$ .

When ties exist, we define  $\tau_K$  (also called *Tau-b* statistic in this case) by

$$\tau_K = \frac{n_c - n_d}{\sqrt{(n_0 - n_1)(n_0 - n_2)}}, \quad (2.84)$$

---

<sup>20</sup>The order in which the off-diagonal entries of  $\mathbf{T}^{\text{data}}$  and  $\mathbf{T}^{\text{model}}$  are concatenated is arbitrary, as long as it is the same order for both matrices.



where

$$\begin{aligned} n_1 &= \sum_i t_i(t_i - 1)/2, \\ n_2 &= \sum_i u_i(u_i - 1)/2, \end{aligned} \tag{2.85}$$

and the quantities  $t_i, u_i$  are the number of tied values in the  $i^{\text{th}}$  group of ties for  $\mathbf{X}$  and  $\mathbf{Y}$ , respectively. For example, when  $\mathbf{X} = (1, 4, 4, 4, 5, 5)$ , there are two groups of ties. The first group consists of the values equal to 4 and the second group consists of the values equal to 5. There are three values of 4 and two values of 5, so  $t_1 = 3$  and  $t_2 = 2$ .

Both  $r_s$  and  $\tau_K$  are rank-correlation coefficients and are invariant under changes to the values of  $\mathbf{X}$  and  $\mathbf{Y}$  as long as the ranks  $\text{rg}_X$  and  $\text{rg}_Y$  remain unchanged. We anticipate that the Pearson correlation coefficient is more applicable than the rank correlations in most cases when comparing  $\mathbf{T}^{\text{data}}$  and  $\mathbf{T}^{\text{model}}$ , as we primarily seek for  $T_{ij}^{\text{model}}$  to be close to  $T_{ij}^{\text{data}}$ , rather than the ranks of the respective quantities.

### Common part of links (CPL)

The *Common part of links* (CPL) measures the extent to which a model recovers the topological structure of the *mobility-flow network* of  $\mathbf{T}^{\text{data}}$ . The mobility-flow network of an  $n \times n$  OD matrix  $\mathbf{T}$  is the unweighted, directed network with  $n$  nodes, where the directed edge  $(i, j)$  is present if and only if  $T_{ij} > 0$ . In other words, we obtain the adjacency matrix of the mobility-flow network from  $\mathbf{T}$  by setting all positive entries of  $\mathbf{T}$  to 1. The CPL is defined by

$$\text{CPL}(\mathbf{T}^{\text{data}}, \mathbf{T}^{\text{model}}) = \frac{\sum_i \sum_{j \neq i} \mathbf{1}_{T_{ij}^{\text{data}} > 0} \mathbf{1}_{T_{ij}^{\text{model}} > 0}}{\sum_i \sum_{j \neq i} \mathbf{1}_{T_{ij}^{\text{data}} > 0} + \mathbf{1}_{T_{ij}^{\text{model}} > 0}}, \tag{2.86}$$

where  $\mathbf{1}$  is the indicator function. The CPL measures the number of directed edges that the mobility-flow networks of  $\mathbf{T}^{\text{data}}$  and  $\mathbf{T}^{\text{model}}$  have in common divided by the total number of directed edges in the two networks. The value ranges between 0 (when the two networks have no edges in common) to 1 (when the two networks are identical). A CPL value of 1 does not imply that the two OD matrices are identical, as we measure only the topological structure with CPL (because one ignores edge weights). Additionally, for the models in this section,  $T_{ij} > 0$  whenever  $O_i = \sum_j T_{ij} > 0$  and  $D_j = \sum_i T_{ij} > 0$ . That is, there is a directed edge from  $i$  to  $j$  whenever node  $i$  has positive outflow and  $j$  has positive inflow. Therefore, it is likely that CPL is not a good measure for these cases. However, it is useful for models and situations that do not necessarily yield complete OD matrices  $\mathbf{T}^{\text{model}}$ . To avoid having complete OD

matrices, one can modify our models by thresholding  $\mathbf{T}^{\text{model}}$  and setting  $T_{ij}^{\text{model}}$  to 0 if it is below some threshold  $z$  (e.g.,  $z = 1$ ). Alternatively (and equivalently in terms of the value of CPL), we can change the definition of the mobility-flow network such that an edge is present if  $T_{ij} > z$ .

### Application-specific measures

One can also examine application-specific measures. For example, one may be interested in models that recover some statistical features of the mobility flow, such as the distance distribution of the trips [31]. A possible goodness-of-fit measure is then the Kolmogorov–Smirnov distance<sup>21</sup> [69] between the empirical trip-distance distribution and the trip-distance distribution of the model.

In our application of modelling the mobility flow of customers between zones in supermarkets (see Chapter 5), we are interested in the number  $v_i$  of customer visits to each zone  $i$ . We estimate  $v_i$  from the OD matrix  $\mathbf{T}$  by assuming that customers walk a shortest path in their trips. In Chapter 5, we introduce the application-specific goodness-of-fit measure  $\text{NRMSE}_v$ , which measures the normalized root-mean-square error (NRMSE) in  $v_i$  between the values of  $v_i$  estimated from the empirical transition matrix  $\mathbf{T}^{\text{data}}$  and those estimated from the model estimate  $\mathbf{T}^{\text{model}}$ . We describe the measure and the calculation of  $v_i$  in Section 5.3.4.

### 2.4.5 Parameter calibration

We calibrate the model parameters of mobility models by maximizing a goodness-of-fit measure. Such a method is model-agnostic; in other words, we can apply this method to any model, independent of the form of  $\mathbf{f}$ . In several past studies [59, 60, 72], the CPC score was maximized for parameter calibration. For models with one parameter (such as the IO model or extended radiation model), finding a (local) maximizer of the CPC score is generally straightforward with a standard optimization solver. However, this task is much harder for models with several parameters (such as for some versions of the gravity model).

There are also model-specific calibration methods for the gravity model with power-law deterrence function. In this model, one can apply generalized linear models (GLM) [77] to fit parameters of the gravity model [80].

---

<sup>21</sup>The *Kolmogorov–Smirnov distance* between two distributions is the supremum of the absolute difference between the cumulative distribution functions of the two distributions.

### 2.4.6 Framework for applying mobility models

Given data of the mobility flow  $\mathbf{T}^{\text{data}}$ , the distance between any two locations, and other information about each location, we apply the following framework for modelling mobility flow using mobility models. In this framework, we are interested mainly in achieving a good fit with our model.

1. Let  $O_k^{\text{data}}$  and  $D_k^{\text{data}}$  be the empirical number of departing and arriving trips (respectively) at location  $k$  (if available). (In other words,  $O_k^{\text{data}}$  and  $D_k^{\text{data}}$  are the row and column sums of  $\mathbf{T}^{\text{data}}$ .) We distinguish two types of population: The origin population  $m_i = O_i^{\text{data}}$  and the destination population  $m_j = D_j^{\text{data}}$ . Similarly, we distinguish between two types of opportunities: origin opportunities  $m_i = O_i^{\text{data}}$  and destination opportunities  $m_j = D_j^{\text{data}}$ .
2. Choose an appropriate version of each model (constrained or unconstrained version) depending on the application.
3. Choose an appropriate goodness-of-fit measure and calibrate the models to maximize the goodness-of-fit measure.
4. Evaluate models by comparing the values of their goodness-of-fit measures.

We apply this framework in Chapter 5.

## 2.5 Customer mobility in supermarkets

We now review the literature on studies of customer mobility in supermarkets. There have only been limited number of studies, and they can be divided roughly into clustering algorithms on data, empirical studies, and models of customer shopping journeys.

### 2.5.1 Clustering of customer shopping journeys

Researchers have applied various clustering methods to identify groups of customers with similar mobility patterns [44, 47, 57, 105]. A general framework for clustering customer shopping journeys is the following:

1. Represent each shopping journey as a vector  $\mathbf{v}$ .
2. Define a similarity measure  $S(\mathbf{v}_1, \mathbf{v}_2)$  that measures a distance between two vectors  $\mathbf{v}_1$  and  $\mathbf{v}_2$ .

3. Apply a clustering algorithm that attempt to group vectors that are sufficiently similar to each other.

For example, Larson *et al.* [57] represented each shopping journey as a sequence

$$\mathbf{v} = ((x_1, y_1), \dots, (x_{100}, y_{100})) \quad (2.87)$$

of coordinates, where  $(x_l, y_l)$  are the coordinates of a customer's trolley location after it has traversed  $l\%$  of the total journey distance. They used  $L_2$  distance as a similarity measure and then applied the  $k$ -medoids clustering algorithm (a variant of the  $k$ -means clustering algorithm). The  $k$ -medoids algorithm partitions a data set into  $k$  clusters and attempts to minimize the total distance between the data points (i.e., shopping journeys) in a cluster and a data point (called *medoid*) that is designated as the centre of the cluster. Unlike the  $k$ -means clustering algorithm, the centre needs to be an actual shopping journey from the data in the  $k$ -medoids algorithm. This ensures that the cluster centre is a feasible journey that does not violate the spatial constraints (e.g., aisles) of a store. Larson *et al.* [57] claimed that a cluster centre represents the canonical journeys of each group of customers and thus provides a summary of the travel behaviour of that group. However, they did not show that the medoid of each cluster is qualitatively similar to the journeys in the cluster. Another common approach is to divide a store into zones and to represent each shopping journey by a sequence of zones that a customer visits (in the order of visiting them) [44, 47, 105]. In other words,

$$\mathbf{v} = (z_1, \dots, z_M), \quad (2.88)$$

where  $z_k$  for  $k = 1, \dots, M$  (for some  $M \geq 1$ ) are the zones that a customer visited in a shopping journey. In such an approach, the distance between two shopping journeys is given by the length of the longest common subsequence divided by some normalization factor (e.g., the sum of the lengths of the shopping journeys) [47, 105]. Clustering approaches can be useful for identifying different types of customers who visit stores based on their mobility patterns, but they have not provided a way to forecast how mobility patterns change in new store layouts.

## 2.5.2 Empirical studies

In an empirical study, Gil *et al.* [33] examined behavioural and demographic attributes that characterize each of the customer groups that they obtained using the  $k$ -medoids algorithm. The authors applied the algorithm on customer shopping journey data from security cameras, and they obtained behavioural and demographic data from

surveys and interviews. They reported that most clusters have similar distributions of attributes and that no single variable (such as gender or shopping duration) is able to distinguish between the clusters.

Using customer movement and purchase data associated to each journey, Hui *et al.* [42] reported that customers systematically deviate from an optimal travelling-salesman-problem (TSP) route. For each shopping journey and associated purchase record, an optimal TSP route of a journey is defined as a path of minimal length through a store that visits the locations of the items that are purchased in the journey. Hui *et al.* [42] reported that customers do not tend to take a shortest route between two consecutive purchased items, but they also concluded that customers generally visit the items in the same order as in an optimal TSP route.

In another study, Hui *et al.* [43] reported a positive correlation between unplanned spending of customers and travel distance. In particular, the more a customer deviates from their optimal TSP path (technically, from one of their optimal TSP paths), the more money they spend on unplanned purchases. The authors conducted a field test to test this hypothesis. In this field test, they gave customers targeted promotions. Hui *et al.* reported that promotions that required customers to deviate more from their optimal TSP path resulted in an increase in unplanned spending in comparison to promotions on items that were close to their optimal TSP path.

Sorensen *et al.* [93] analysed a large data set of customer trajectories and described the distributions of some aspects of customers' mobility and purchase behaviour. In particular, they reported that the proportion of a store that is covered and the time spent in a store both follow logit-normal distributions and the number of purchased items follows a mixed Poisson log-normal distribution.

### 2.5.3 Models of customer behaviour

There are a few existing models for customer mobility. Farley *et al.* [27] proposed one of the earliest models for shopping journeys in supermarkets in 1966. Their model is a (Markovian) random walk model on zones in a supermarket. They calibrated a transition matrix, where the transition probabilities are determined by the number of sales in each location, whether a zone is in the perimeter, and whether the transition follows the circular tendency of movement that was observed in the store (e.g., Farley reported that customers tend to walk in a counter-clockwise direction in a store). However, Farley did not find a good fit to their empirical transition matrix.

Hui *et al.* [41] introduced an agent-based model for customers with a stochastic decision process that incorporates purchase behaviour and movement. Customers take

random decisions to perform actions (e.g., move, stay in a zone to consider items, shop for an item, and so on), where the decision probabilities depend on attraction values that a customer associates to each of the zones and the previous history of their journey (including purchase history). The authors of [41] additionally incorporated different biases (e.g., time pressure) into their model to test certain behavioural hypotheses (e.g., time pressure cause customers to buy less). They used a hierarchical Bayesian framework to calibrate their parameters and calculated the associated confidence interval and  $p$ -value for each calibrated parameter. They tested each behavioural hypothesis by assessing whether the value of the calibrated parameters that are associated with each bias is positive and statistically significant (indicating that a bias is present). A model of similar flavour to [41] was introduced by Yan *et al.* [106]. A major shortcoming of the previous two studies is that the models were not tested on new and/or unseen data from a store with a different layout, so the models may not forecast customer mobility well in new store layouts.

In [14], Boros *et al.* assumed a TSP model for shopping journeys to identify store layouts that maximize the shopping journey lengths. In the TSP model, customers take a shortest path that visits the locations of all the items that they buy. To avoid calculating a TSP path for all customers (which would be computationally intractable), they clustered customers of a small supermarket into 27 representative customer types based on their purchases. Note that they did not cluster based on the customer shopping journeys, as they did not have customer shopping journey data. Each customer type is defined by a set of item categories. For each customer type, they identified a shortest path that visits each of the associated item categories (i.e., a TSP path). They sought a store layout that maximizes the weighted mean of the TSP path lengths from each customer type. (They weighted it by the number of customers for each customer type.) The weighted mean of the TSP path lengths approximates the mean path length of customers, which has been reported to correlate with unplanned spending [43], as mentioned in Section 2.5.2. Boros *et al.* solved this optimization problem exactly. The algorithm took two days to finish for the store that they considered. While the TSP model appears to be a fairly accurate model for customer movement [42], it is very slow to identify a TSP path in large stores. This makes it prohibitively expensive to optimize a store layout under a TSP customer mobility model. Furthermore, Boros *et al.* did not demonstrate that the customer type of each cluster is, in fact, representative for the customers in the cluster.

#### 2.5.4 Summary

In summary, previous studies of customer mobility have mainly been of a descriptive nature. These studies used data clustering to describe different type of customers who visit supermarkets [33, 44, 47, 105], quantified how efficient customers travel in a store [41], and examined correlations between customer-journey length and spending on unplanned purchases [43]. There are a small number of existing models, but these models have either been unable (at least thus far) to successfully fit unseen (out-of-sample) data [27], have not been tested on unseen data [41, 106], or are prohibitively expensive to simulate [14] for a larger store. Our work attempts to fill this gap by developing and analysing models of customer mobility. Additionally, we examine congestion models for supermarkets, which (to the best of our knowledge) has not been done in the literature. Similar to [27, 41, 106], we represent our stores as networks.

## Chapter 3

# Representing stores as networks

In this chapter, we describe how we represent stores as undirected, unweighted spatial networks, which we call *store networks*, and present some of their network properties. We follow a similar approach to [41], in which each node in a store network represents a zone (i.e., an area) in a store. We manually divide the floor area of each store into rectangular zones. We identify the edges in an automated fashion: we develop an algorithm that adds edges between two nodes if one can walk between their corresponding zones without entering any other zones, so it connects contiguous zones.<sup>1</sup>

A network is an useful representation of a store’s layout, as it allows us to explore the set of store layouts by exploring the set of networks. Note, however, that not every network corresponds to a store layout that can be implemented in practice. While we do not explore in full the space of implementable store networks, we discuss some of their properties in Chapter 4.

The chapter is organized as follows. We describe the data about the stores in Section 3.1. We introduce the procedure that we use to create store networks in Section 3.2. We describe how we assign each shelf to a zone that contains the shelf in Section 3.3. Finally, we show some of the network properties of the store networks in Section 3.4.

### 3.1 Data

We have data about 17 large Tesco stores, which we name Stores A–Q. For each store, the data set contains the following information for each shelf of the store:

---

<sup>1</sup>The algorithm, which we describe in Section 3.2.2, is not guaranteed to detect all edges that satisfy this criterion, but in practice it detects a large majority of them. Furthermore, it may add a small number of edges between two non-contiguous zones.



1. Coordinates: Each shelf fills a floor area of rectangular shape and the data provides the  $(x, y)$ -coordinates of the corners of the rectangle, where we define the  $x$ -axis and  $y$ -axis to be the horizontal and vertical axis, respectively.
2. Direction in which the shelf is facing: The data provides a unit vector that points in the direction in which the shelf is facing.

We also have the coordinates of the approximate locations of the entrance and till area for each store. In Figure 3.1, we show the shelves of Store A in grey and the approximate locations of the entrance and till area.

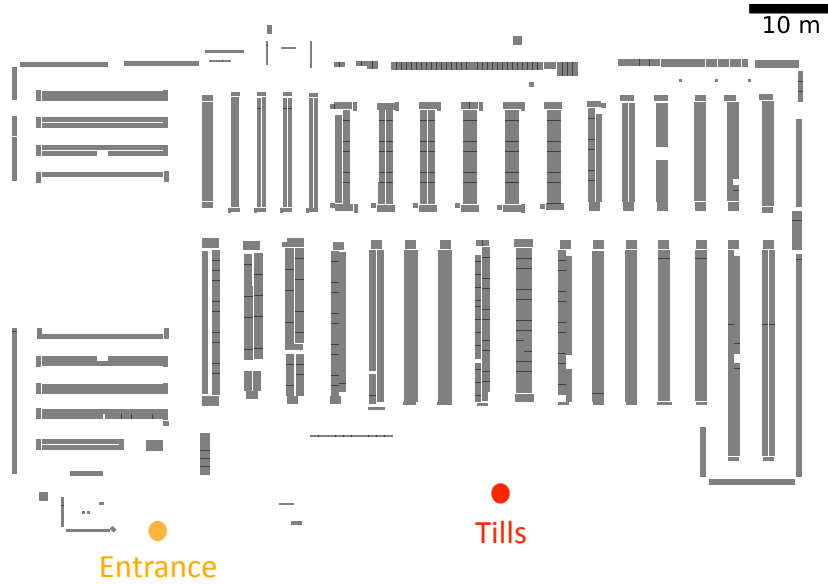
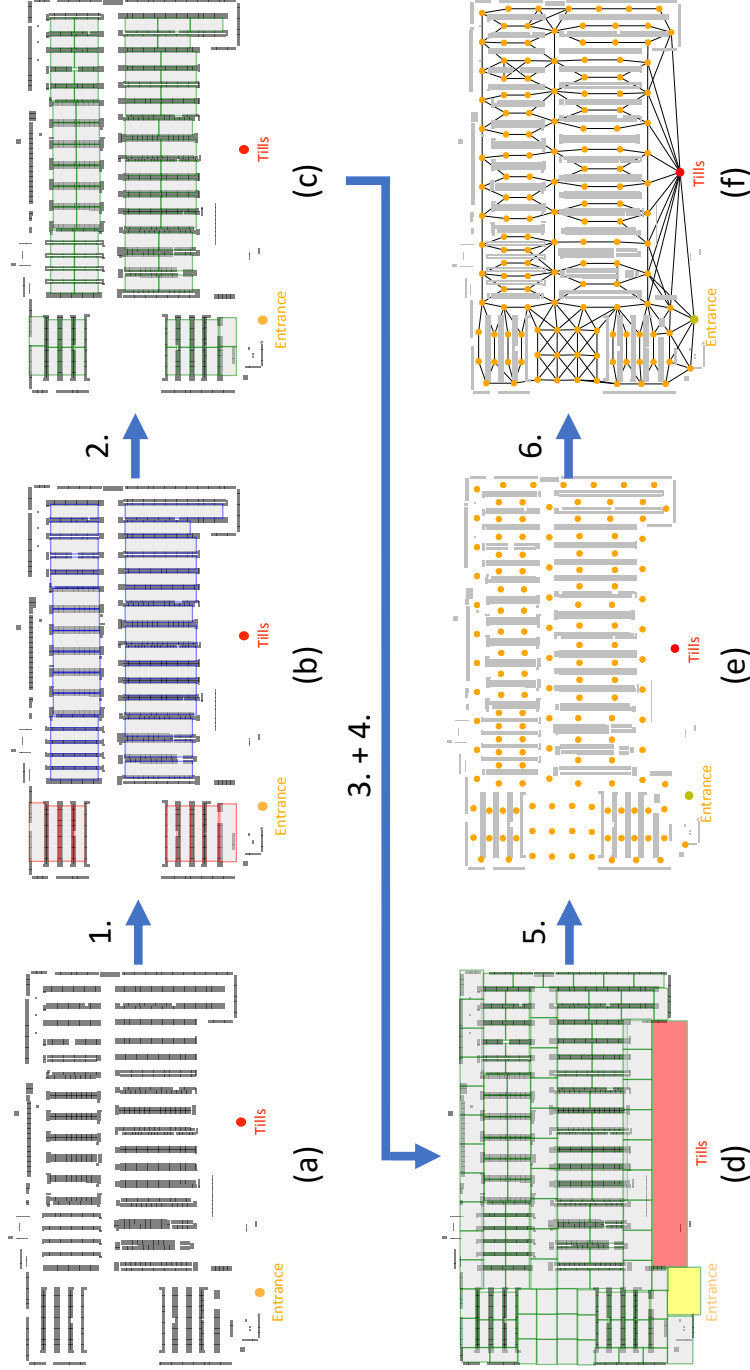


Figure 3.1. Location of shelves (grey rectangles) and approximate locations of the entrance (yellow circle) and till area (red circle) in Store A.

## 3.2 Creating the store network

We create the store network by creating the nodes first and then identifying the edges between them. We summarize our procedure in Figure 3.2 and explain each part of the procedure in detail in the next subsections.



1. Identify aisles.
2. Divide aisles into zones.
3. Place entrance and till zones.
4. Fill remaining floor area with zones.
5. Convert zones into nodes of store network.
6. Add edges of store network using fine-grained lattice-network.

Figure 3.2. Schematic summarizing our procedure for creating a store network. The gray rectangles are the shelves in a store. The red and blue rectangles in (b) are the horizontal and vertical aisles, respectively. The blue rectangles in (c) and (d) are the zones that we defined. The yellow circles in (e) and (f) are the nodes of the store network. Each circle is located at the centre of the corresponding zone. We show the final store network in (f).

### 3.2.1 Creating the nodes

We divide the floor space of each store into zones, which become the nodes of the store network. Zones are rectangular and their edges are either horizontal (parallel to the  $x$ -axis) or vertical (parallel to the  $y$ -axis). (In other words, no rectangle is tilted.) We define the *length* of a zone by the length of the longer side of the corresponding rectangle. Similarly, we define the *width* of a zone by the length of the shorter side of the corresponding rectangle. We aim to have zones of similar zone lengths, except for the entrance and till zones, which may be longer. We set a *target mean zone length*  $\bar{l} = 7$  m, which influences the length of the zones at the end of the process. The zone widths may vary depending on the width of the aisle and walkway that the zone is in.

To create the zones, we first manually identify the aisles in a store. Each aisle is a rectangular area between two parallel rows of shelves that face each other. In all of the stores that we consider, the edges of all aisles are parallel either to the  $x$ -axis or to the  $y$ -axis. For each aisle, let  $(x_{\min}, y_{\min})$ ,  $(x_{\min}, y_{\max})$ ,  $(x_{\max}, y_{\max})$ ,  $(x_{\max}, y_{\min})$  be the coordinates of the corners of the corresponding rectangle with  $x_{\min} < x_{\max}$  and  $y_{\min} < y_{\max}$ . Then  $\Delta x = x_{\max} - x_{\min}$  and  $\Delta y = y_{\max} - y_{\min}$  are the dimensions of the rectangle. If  $\Delta x > \Delta y$ , we call the aisle a *horizontal* aisle. Otherwise, it is a *vertical* aisle. We plot the aisles of Store A in Figure 3.2(b).

We then divide each rectangular aisle into rectangular zones of equal sizes. We describe below the process of dividing a horizontal aisle into zones, but the process is analogous for a vertical aisle (except rotated by 90 degrees). For each aisle, we calculate  $k = \lfloor \Delta x / \bar{l} \rfloor$  (i.e.,  $\Delta x / \bar{l}$  rounded to its nearest integer) and divide the aisle into  $k$  smaller rectangles of the same size (see Figure 3.3). These smaller rectangles then become the zones. Our choice of  $k$  ensures that horizontal length of each smaller rectangle is close to  $\bar{l}$ . We show the zones of Store A created from its aisles in Figure 3.2(c).

We place the entrance and till zones at their respective locations and manually divide the remaining floor space into zones (see Figure 3.2(d)). For most of the remaining floor space, we use a similar process as in the aisles: We select large rectangles which each covers an area (e.g., the walkway at the perimeter of the store). We divide each rectangle into smaller rectangles (as depicted in Figure 3.3), which become the zones. For any floor area that we cannot fill in this way with zones, we manually add zones in that area one by one, where we try to have zones of length close to  $\bar{l}$ . We show the final store zoning of Store A in Figure 3.2(d). (We show the store zoning for all 17 stores in Appendix A.) We assign the location of each node to

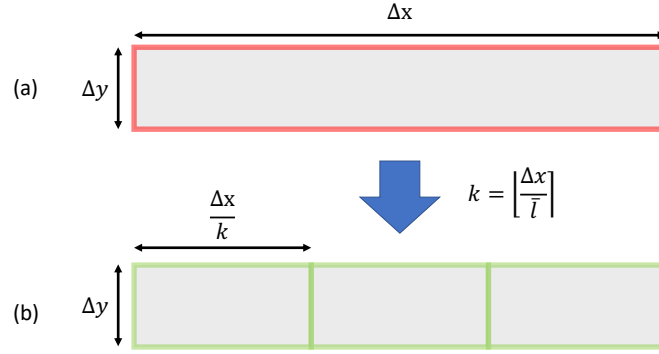


Figure 3.3. Dividing a horizontal rectangle into smaller rectangles.

be the centroid of the corresponding zone. We show the nodes of the store network of Store A in Figure 3.2(e)

Finally, we label the nodes from 1 to  $n$ , where  $n$  is the number of zones. We ensure that the entrance zone receives the label 1 and that the till zone receives the label  $n$ .

### 3.2.2 Creating the edges

To create the edges, we identify pairs of contiguous zones by using a *fine-grained lattice-network* representation of the store. The fine-grained lattice network is provided by Tesco and is a spatial network with its nodes arranged in a lattice. The horizontal and vertical spacing between nodes in the lattice is approximately 2 m. Additionally, there is a node in front of each shelf at a distance of approximately 0.5 m to the front face of the shelf. Nodes are located only on the floor space; no nodes intersect with shelves. Two nodes are adjacent via an edge if the line segment between the two nodes is less than 5 m long and does not intersect with any shelves. We show the fine-grained lattice network of Store A in Figure 3.4. The fine-grained lattice network is a discrete representation of a store at a higher resolution than our store network. We define zones  $i$  and  $j$  to be contiguous if there exists two nodes  $v_1$  and  $v_2$  in the fine-grained lattice network such that nodes  $v_1$  and  $v_2$  are in zones  $i$  and  $j$ , respectively, and nodes  $v_1$  and  $v_2$  are adjacent in the fine-grained lattice network. If this is the case, we add an undirected edge between  $i$  and  $j$  in the store network. We show the store network of Store A in Figure 3.2(f). (We show the store networks for all 17 stores in Appendix A.)

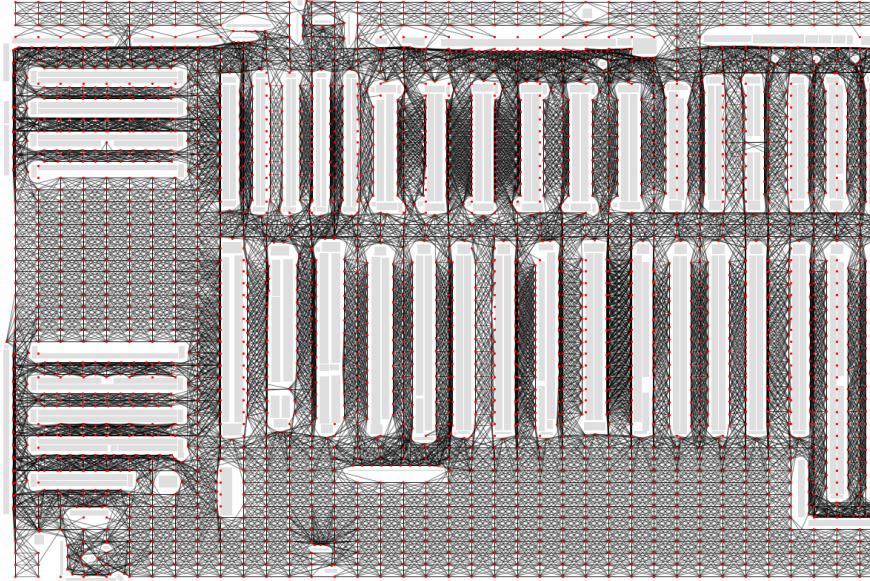


Figure 3.4. Fine-grained lattice network of Store A. We depict the nodes by red dots and the edges by black lines.

### 3.3 Assigning shelves to zones

In later chapters, we use the number of purchases that occur in each zone. To calculate this number, we assign each shelf to a zone from where customers generally pick up the items in the shelf. Our procedure is as follows. We find for each shelf the point  $P$  that is 1 m in front of the shelf. See Figure 3.5 for a diagram on how we find  $P$ . The point  $P$  represents the approximate location of where customers stand when they pick up items from the shelf. We then identify the zones (if there are any) that contain  $P$ . If there is a unique zone that contains  $P$ , and we map the shelf to this zone. Otherwise,  $P$  is contained in zero or multiple zones<sup>2</sup>. In this case, we select the zone whose centroid is closest to  $P$ . This case only occurs only for a small proportion (around 2%) of the shelves across the 17 stores.

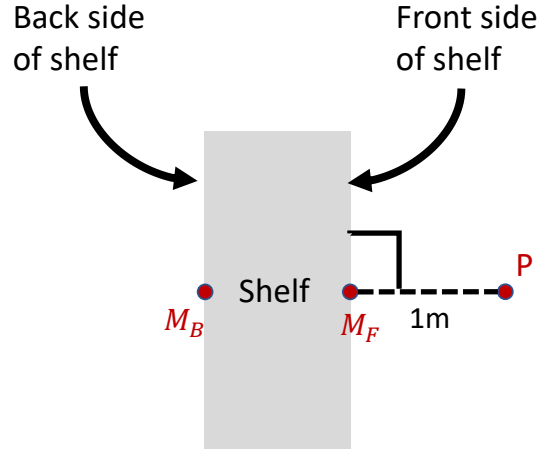


Figure 3.5. Schematic of how we identify the point  $P$  that is located in front of a shelf. We depict the shelf in grey, viewing from the top. The point  $M_F$  is the midpoint of the front side of the shelf (i.e., the side facing the customers), and the point  $M_B$  is the midpoint of the backside. We find the point  $P$  by  $P = M_F + (M_F - M_B)/\|M_F - M_B\|$ , which is 1 m in front of  $M_F$ .

### 3.4 Properties of the store networks

We create store networks for 17 stores (Stores A–Q) and summarize their properties in Table 3.1. The store networks have different sizes, ranging from 61 nodes (Store K) to 197 nodes (Store Q). The mean degrees are similar; they are between 3.33 and 4.39. This is because they are spatial networks, whose mean degrees and degree

<sup>2</sup>Some zones overlap over a small area.

distributions tend to be similar [8]. We assign a length  $l_{ij}$  to each edge  $\{i, j\}$  by the Euclidean distance between the nodes  $i$  and  $j$ . We then calculate the shortest-path length between any two nodes. For most store networks, the shortest-path length between the entrance node and the till node is shorter than the mean shortest-path length (see Table 3.1). We obtain the same result when we measure the length of a path by the number of traversed edges. Therefore, the entrance and the till nodes are relatively close to one another in the store networks. Furthermore, we note that the mean zone length is approximately 7 m and therefore close to our target mean zone length  $\bar{l}$ . The store networks are also approximately planar, in the sense that we only need to delete a small number of edges from each store network to obtain a completely planar network. For example, we can delete 21 edges (7.7% of the edges) from the store network of Store A to obtain a planar graph, which we show in Figure 3.6.

*Table 3.1. Statistics of the 17 store networks and store zoning (Stores A–Q). For each measure, we show the minimum, mean, and maximum value across the 17 stores.*

	Minimum value	Mean value	Maximum value
Number of nodes	61	124.29	197
Number of edges	128	236.24	401
Mean degree	3.33	3.82	4.39
Shortest path length from entrance to tills (in m)	18.24 m	37.46 m	59.98 m
Mean shortest-path length (in m)	31.89 m	45.09 m	56.45 m
Shortest path length from entrance to tills (in number of traversed edges)	1	2.41	6
Mean shortest-path length (in number of traversed edges)	3.76	5.62	7.20
Mean zone length (excl. entrance and tills)	6.42 m	6.91 m	7.65 m

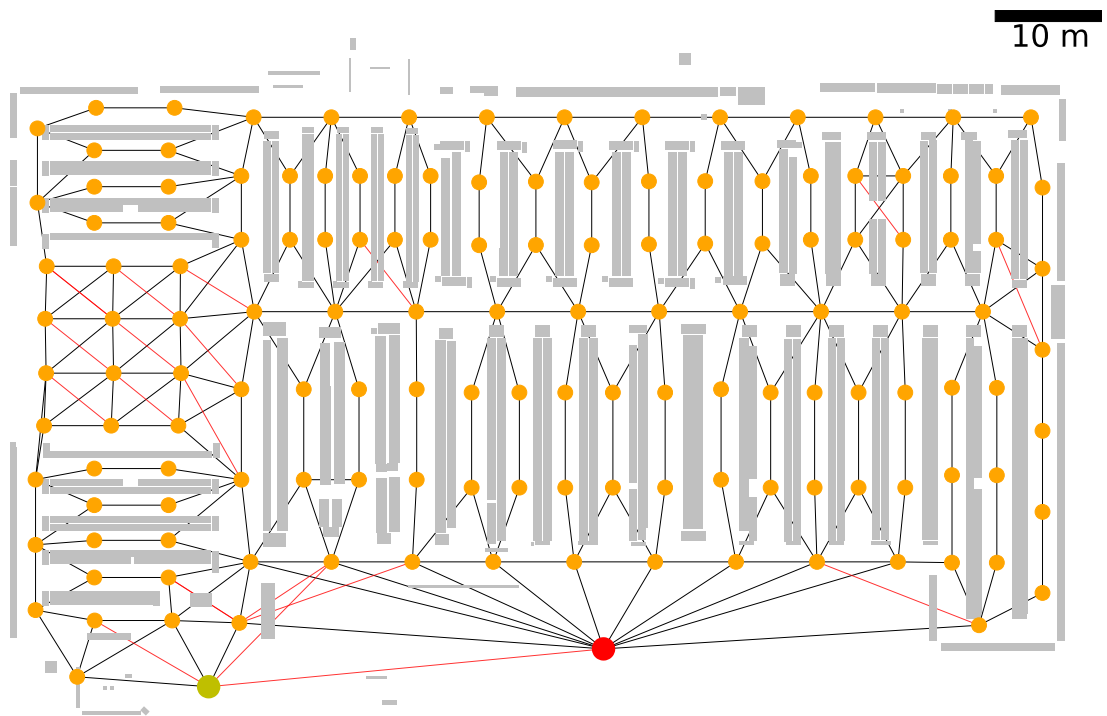


Figure 3.6. Planar network obtained after deleting 21 edges (in red) from the store network of Store A.



## Chapter 4

# Minimizing congestion in queueing networks

In this chapter, we study congestion in open queueing networks and analyse how network topology<sup>1</sup> affects congestion, which we measure by the total mean queue size  $Q$ . We focus on queueing networks with a single source node and a single sink node at stationarity. Walkers enter a network at the source node and walk according to an unbiased random walk. At each node, walkers queue up to be served. Once a walker reaches the sink node, it is removed from the network.

This problem is a simple model for customer mobility inside a supermarket. In this application, customers enter a store at the source node (representing the entrance) and walk randomly (‘diffuse’) around the supermarket. Each node represents a zone in a supermarket. Once a customer reaches the sink node (representing the till area), they leave the supermarket. This is a simplified model of mobility in a supermarket; it neglects shopping intention, customer heterogeneity, and many other factors. However, simple models are useful, as they can provide insight into the problem and the application. Furthermore, we can extend the model to capture more realistic aspects.

We are interested in modelling and measuring congestion in our model. For this, we treat congestion as a condition that increases the journey time of customers, and we seek to minimize the mean journey time. By Little’s Law (see Section 2.3.2), minimizing the mean journey time is equivalent to minimizing the total mean queue size  $Q$ . As it is typically easier to directly calculate  $Q$ , we measure congestion by  $Q$ .

Our analysis also extends the traffic-dynamics model (described in Section 2.2) by considering a special case of the model constrained on random walks with a single server and a single sink. While existing work on the traffic-dynamics model focused on minimizing  $\lambda_{\max}$  (the maximum arrival rate to a node) [17] instead of  $Q$ , our

---

<sup>1</sup>Recall that we consider queueing networks with unweighted network structure.

methods are applicable for both congestion measures, as we show in Section 4.3.3 and Appendix B.6.

In this chapter, we present the single-source, single-sink queueing model in Section 4.1 and we address the following questions:

- How does the total mean queue size  $Q$  depend on network topology (Section 4.2)?
- Which network topologies minimize  $Q$  (Section 4.3)?
- Given a network, which edges should one add or delete to reduce  $Q$  (Section 4.4)?

## 4.1 Single-source, single-sink open queueing network with unbiased random walkers

Our model is an open queueing network with a single source node (labelled 1) and a single sink node (labelled  $n$ ). Walkers arrive at node 1 from outside the system at rate  $\lambda_{01} = \lambda$ . We call  $\lambda$  the *external arrival rate*. We also assume that the sink node  $n$  has no out-edges, so that it is a point of no return. This is the case for most customers in a supermarket: once they reach the tills, they do not tend to go back into the store. There are further constraints on the network topology due to the reachability conditions (see Section 2.3.1), which in a single-source, single-sink open queueing network implies that each node  $k$  is in both the out-component of 1 and the in-component of  $n$ . That is,

$$C_{\text{out}}(1) = C_{\text{in}}(n) = \{1, \dots, n\}. \quad (4.1)$$

Therefore, for a fixed number  $n \in \mathbb{Z}_+$  of nodes, we consider directed networks in the following set:

$$\mathcal{C}_n = \{G = (V, E) : |V| = n, C_{\text{out}}(1) = C_{\text{in}}(n) = \{1, \dots, n\}, d_n^{\text{out}} = 0\}. \quad (4.2)$$

For any graph  $G \in \mathcal{C}_n$ , all nodes except  $n$  have at least one outgoing edge, as they are in the in-component of  $n$ , so  $d_i^{\text{out}} \geq 1$  for all  $i \in \{1, \dots, n-1\}$ .

We assume that walkers traverse the network according to an unbiased random walk, so the transition matrix  $\mathbf{P}$  has entries

$$P_{ij} = \begin{cases} A_{ji}/d_i^{\text{out}}, & \text{for } i = 1, \dots, n-1, \\ 0, & \text{for } i = n. \end{cases} \quad (4.3)$$

This implies that every edge has positive transition probability, so every network in  $\mathcal{C}_n$  satisfies the reachability conditions.

Each node  $i$  is a FIFO single-server node with fixed service rate  $\mu_i$ , with  $\mu_i$  assumed to exceed the arrival rate  $\lambda_i$  of  $i$ . For most of the calculations in this thesis, we assume that the service rates are homogeneous among all nodes (i.e.,  $\mu_i = \mu$ ). However, we present our model in the more general form. We consider the system at stationarity and we therefore neglect temporal variations in the external arrival rate and in the service rate. In a reality, however, congestion in supermarkets is likely a phenomenon that depends on time, as congestion tend to occur at certain times of the day (e.g., lunch time) or certain times of the year (e.g., Christmas). But if the system reaches the stationary state sufficiently quickly (relative to time scale of the temporal variations), one can approximate the dynamics of the system by its stationary state.

#### 4.1.1 Traffic equations

We calculate  $\lambda_i$  by solving the traffic equations (2.38), which in our single-source, single-sink model take the form

$$\lambda_i = \delta_{1i}\lambda + \sum_{j=1}^{n-1} \lambda_j P_{ji}, \quad \text{for } i = 1, \dots, n, \quad (4.4)$$

where  $\delta_{1i}$  is the Kronecker delta. The sum goes only to  $n - 1$ , as  $P_{ni} = 0$  for all  $i$ . We divide both sides of Equation (4.4) by the external arrival rate  $\lambda$  and define  $\tilde{\lambda}_i = \lambda_i/\lambda$ , which is the non-dimensional arrival rate of node  $i$  relative to the external arrival rate. Similarly, we non-dimensionalize the service rate  $\mu_i$  by defining  $\tilde{\mu}_i = \mu_i/\lambda$ . Because  $\mu_i > \lambda_i$ , it is also true that  $\tilde{\mu}_i > \tilde{\lambda}_i$ . For simplicity, we drop the tilde and redefine  $\tilde{\lambda}_i$  by  $\lambda_i$  and  $\tilde{\mu}_i$  by  $\mu_i$ , so from now on  $\lambda_i$  and  $\mu_i$  are non-dimensional quantities. (Our notation is therefore equivalent to setting  $\lambda = 1$ .) The traffic equations then become

$$\lambda_i = \delta_{1i} + \sum_{j=1}^n \lambda_j P_{ji}, \quad \text{for } i = 1, \dots, n. \quad (4.5)$$

In matrix form,

$$(\mathbf{I} - \mathbf{P}^T) \mathbf{l} = \mathbf{b}, \quad (4.6)$$

where  $\mathbf{l} = (\lambda_1, \dots, \lambda_n)^T$  and  $\mathbf{b} = (1, 0, \dots, 0)^T$ . The solution to Equation (4.6) is

$$\mathbf{l} = (\mathbf{I} - \mathbf{P}^T)^{-1} \mathbf{b}. \quad (4.7)$$

(Note that  $\mathbf{I} - \mathbf{P}^T$  is invertible, as discussed in Section 2.3.1.) The solution  $\lambda_i$  for Equation (4.5) is positive (as discussed in Section 2.3.1):

$$\lambda_i > 0, \quad \text{for all } i = 1, \dots, n. \quad (4.8)$$

Furthermore,

$$\lambda_n = 1, \quad (4.9)$$

as the departure rate from the system must equal the non-dimensional external arrival rate (which is equal to 1) at stationarity. Note that the arrival rates  $\lambda_i$  are independent of the service rates  $\mu_i$  (and only depend on the network topology), as long as the service rates  $\mu_i$  are sufficiently large (specifically,  $\mu_i > \lambda_i$  for all  $i$ ). In other words, the rate at which customers arrive at a node does not depend on how fast node  $i$  can serve customers (as long as it is sufficiently fast) and therefore does not depend on the level of congestion (measured by the queue size) at  $i$ .

We can also express Equation (4.6) in terms of the combinatorial Laplacian  $\mathbf{L}$  (defined in Equation (2.8)) as follows. By Equation (4.9), we can remove the  $n^{\text{th}}$  equation from Equation (4.5), leaving us with  $n-1$  equations for  $n-1$  variables. (Note that the first  $n-1$  equations do not include any occurrence of  $\lambda_n$ .) This corresponds to deleting the  $n^{\text{th}}$  row of  $\mathbf{I} - \mathbf{P}^T$  and the  $n^{\text{th}}$  entry of  $\mathbf{b}$ . We denote the resulting matrix by  $(\mathbf{I} - \mathbf{P}^T)_{1:n-1,1:n}$  and the resulting vector by  $\mathbf{b}_{\text{tr}}$  in Equation (4.6). (The subscript ‘tr’ stands for ‘truncated’.) Equation (4.5) then becomes

$$(\mathbf{I} - \mathbf{P}^T)_{1:n-1,1:n} \mathbf{l} = \mathbf{b}_{\text{tr}}, \quad (4.10)$$

with  $\lambda_n = 1$ . The  $n^{\text{th}}$  column of  $(\mathbf{I} - \mathbf{P}^T)_{1:n-1,1:n}$  is a column of zeros, as node  $n$  has no out-edges. We can therefore further simplify by deleting the  $n^{\text{th}}$  column of  $(\mathbf{I} - \mathbf{P}^T)_{1:n-1,1:n}$ , where we denote the resulting matrix by  $\mathbf{I}_{\text{tr}} - \mathbf{P}_{\text{tr}}^T$ , and deleting the  $n^{\text{th}}$  entry of  $\mathbf{l}$ , where we denote the resulting vector by  $\mathbf{l}_{\text{tr}}$ . Therefore,

$$(\mathbf{I}_{\text{tr}} - \mathbf{P}_{\text{tr}}^T) \mathbf{l}_{\text{tr}} = \mathbf{b}_{\text{tr}}. \quad (4.11)$$

The matrix  $\mathbf{I}_{\text{tr}} - \mathbf{P}_{\text{tr}}^T$  is related to  $\mathbf{I} - \mathbf{P}^T$  as follows:

$$\mathbf{I} - \mathbf{P}^T = \begin{bmatrix} & & & 0 \\ \mathbf{I}_{\text{tr}} - \mathbf{P}_{\text{tr}}^T & & & \vdots \\ & & & 0 \\ * & \cdots & * & 1 \end{bmatrix}, \quad (4.12)$$

where the  $*$  symbol can be any value. Because  $\mathbf{I} - \mathbf{P}^T$  is invertible (see Section 2.3.1),  $\mathbf{I}_{\text{tr}} - \mathbf{P}_{\text{tr}}^T$  is also invertible. Furthermore, as we have an unbiased random walk, it follows that

$$\mathbf{I}_{\text{tr}} - \mathbf{P}_{\text{tr}}^T = (\mathbf{D}_{\text{tr}} - \mathbf{A}_{\text{tr}}) \mathbf{D}_{\text{tr}}^{-1}, \quad (4.13)$$

where  $\mathbf{D}_{\text{tr}}$  and  $\mathbf{A}_{\text{tr}}$  are the matrices obtained by deleting the  $n^{\text{th}}$  row and  $n^{\text{th}}$  column of the degree matrix  $\mathbf{D}$  and the adjacency matrix  $\mathbf{A}$ , respectively. The diagonal entries

of  $\mathbf{D}_{\text{tr}}$  correspond to the out-degrees of nodes  $1, \dots, n-1$  and are therefore strictly positive, so  $\mathbf{D}_{\text{tr}}^{-1}$  is well-defined. The matrix  $\mathbf{L}_{\text{tr}} = \mathbf{D}_{\text{tr}} - \mathbf{A}_{\text{tr}}$  is an  $(n-1) \times (n-1)$  submatrix of the Laplacian  $\mathbf{L}$  that we obtain from  $\mathbf{L}$  by deleting its  $n^{\text{th}}$  row and  $n^{\text{th}}$  column. The matrix  $\mathbf{L}_{\text{tr}}$  is also called the *grounded Laplacian matrix*<sup>2</sup> [108]. We can rewrite Equation (4.11) as

$$\mathbf{L}_{\text{tr}} \mathbf{D}_{\text{tr}}^{-1} \mathbf{l}_{\text{tr}} = \mathbf{b}_{\text{tr}}. \quad (4.14)$$

The invertibility of  $\mathbf{L}_{\text{tr}}$  follows from the invertibility of  $\mathbf{I}_{\text{tr}} - \mathbf{P}_{\text{tr}}^T$  and Equation (4.13). Therefore, the arrival rates are given by

$$\mathbf{l}_{\text{tr}} = \mathbf{D}_{\text{tr}} \mathbf{L}_{\text{tr}}^{-1} \mathbf{b}_{\text{tr}}, \quad (4.15)$$

with  $\lambda_n = 1$ . The computational complexity for solving the traffic equation (Equation (4.5)) is  $\mathcal{O}(n^3)$ .

### 4.1.2 Total mean queue size $Q$

By Equation (2.46), the mean queue length  $\mathbb{E}[X_i]$  at each node  $i$  at stationarity is

$$\mathbb{E}[X_i] = \frac{\lambda_i}{\mu_i - \lambda_i}, \quad \text{for } i = 1, \dots, n. \quad (4.16)$$

Note that  $\mathbb{E}[X_i]$  is an increasing function of  $\lambda_i$  on  $\lambda_i \in [0, \mu_i)$  and an decreasing function of  $\mu_i$  for  $\mu_i > \lambda_i$ . In other words, increasing the arrival rate or decreasing the service rate result in longer queues, which is what we would expect from a queueing system.

By Equation (4.16), we can write

$$Q = \sum_{i=1}^n \mathbb{E}[X_i] = \sum_{i=1}^n \frac{\lambda_i}{\mu_i - \lambda_i}. \quad (4.17)$$

Note that the expression for  $Q$  is independent of whether we use the non-dimensional or dimensional versions of  $\lambda_i$  and  $\mu_i$ .

### 4.1.3 Summary of our model

Our model is a single-source, single-sink open queueing network with unbiased random walks. The model has two inputs: a network  $G = (V, E)$  and the (non-dimensional) service rates  $\mu_i > 0$ . Assuming that  $G \in \mathcal{C}_n$  (i.e.,  $G$  and the corresponding transition

---

<sup>2</sup>One can use the grounded Laplacian matrix to determine the node potentials in an electrical resistor network (which is an undirected network) when the potential of one node is fixed (i.e., grounded).

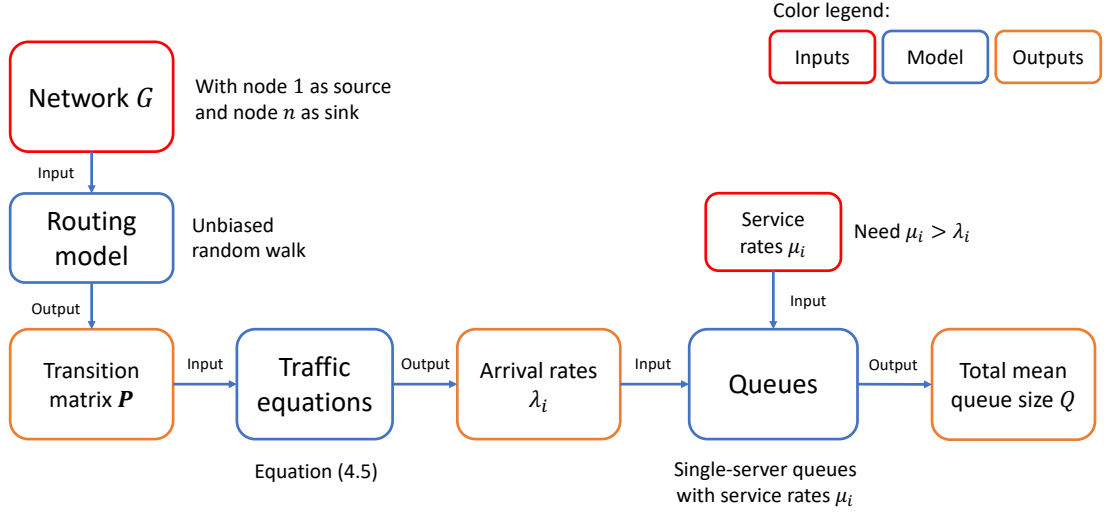


Figure 4.1. Summary of our open queueing network model. The model takes a network  $G$  and the (non-dimensional) service rates  $\mu_i$  as inputs and outputs the total mean queue size  $Q$ .

matrix  $P$  satisfy the reachability conditions) and that the service rates  $\mu_i$  are sufficiently large, the output of our model is the total mean queue size  $Q$ , which we use as our congestion measure. We also summarize the model with its inputs and outputs in Figure 4.1.

#### 4.1.4 Undirected single-source, single-sink open queueing network

In Sections 4.3 and 4.4, we consider single-source, single-sink open queueing networks  $G = (V, E)$  with undirected network topology. In other words, we consider graphs  $G$ , in which every edge  $(i, j) \in E$  implies that the reciprocal edge  $(j, i)$  is also present in  $G$ . As in the case of directed graphs, we assume that node  $n$  is a point of no return; walkers leave the network after being served at node  $n$ . We impose this constraint by assuming that  $P_{ni} = 0$  for all  $i$ . (As the network is undirected, we cannot impose that node  $n$  has no out-edges; otherwise  $n$  has no incident edges and is therefore disconnected from the rest of the graph.) The transition matrix corresponding to an unbiased random walk is then

$$P_{ij} = \begin{cases} A_{ji}/d_i, & \text{if } i = 1, \dots, n-1, \\ 0, & \text{if } i = n. \end{cases} \quad (4.18)$$

In this case, the reachability conditions are satisfied, provided  $G$  is connected and the subgraph of  $G$  that is induced on  $\{1, \dots, n-1\}$  is connected.<sup>3</sup> We define  $\mathcal{U}_n$  to be the set of undirected graphs with  $n$  nodes that satisfy these reachability conditions.

Every undirected single-source, single-sink open queueing network is equivalent to a directed one in terms of the queueing dynamics. We obtain the equivalent directed single-source, single-sink open queueing network by converting every undirected edge  $\{i, j\}$  to a pair of reciprocal edges  $(i, j)$  and  $(j, i)$  except for the edges incident to  $n$ . For the edges incident to  $n$ , we replace each edge  $\{i, n\}$  by the directed edge  $(i, n)$  and we do not add the reciprocal edge  $(n, i)$ , as  $n$  cannot have any no out-edges. (See Figure 4.2 for an example.) Therefore, when we consider the graphs in  $\mathcal{U}_n$ , we essentially consider the subset of graphs  $G = (V, E)$  in  $\mathcal{C}_n$  in which every edge  $(i, j) \in E$  for  $j \neq n$  has the reciprocal edge  $(j, i)$  in  $G$ .



Figure 4.2. Example of (left) an undirected single-source, single-sink queueing network represented by (right) its equivalent, directed version.

## 4.2 Effect of network topology on total mean queue size $Q$

It is not surprising that the total mean queue size  $Q$  depends greatly on network topology (assuming that all other parameters are fixed). In this section, we explore some of the features of our queueing model by considering small directed graphs with  $n = 5$  nodes. (We observe similar features for undirected graphs.)

To give an example of a ‘bad’ network (i.e., one with a large value of  $Q$ ), see the top network in Figure 4.3. In this network, random walkers accumulate in the middle nodes (nodes 2, 3, and 4), whose arrival rates are up to four times higher than those of the sink and source nodes. For fixed service rates  $\mu_i$ , faster arrival rates result in

<sup>3</sup>Note that  $G$  being connected is not enough to satisfy the reachability conditions. Any connected graph that includes a node  $k \neq n$  such that all paths from 1 to  $k$  traverse through  $n$  does not satisfy the reachability conditions. This is because any path  $(v_0, \dots, v_l)$  from 1 to  $k$  traverses an edge  $(v_i, v_{i+1})$ , where  $v_i = n$ . This edge  $(v_i, v_{i+1})$  has 0 transition probability. In other words, no walkers can ever reach  $k$  (as walkers leave the network after reaching  $n$ ), thus violating the reachability conditions.

larger values of  $Q$  by Equation (4.17). By contrast, the bottom network in Figure 4.3 has much lower arrival rates for these intermediate nodes (and the arrival rates of the source and sink node are the same in both networks). Therefore, the total mean queue size  $Q$  is smaller in the bottom network than in the top network; the exact difference in  $Q$  depends on the service rates  $\mu_i$ .

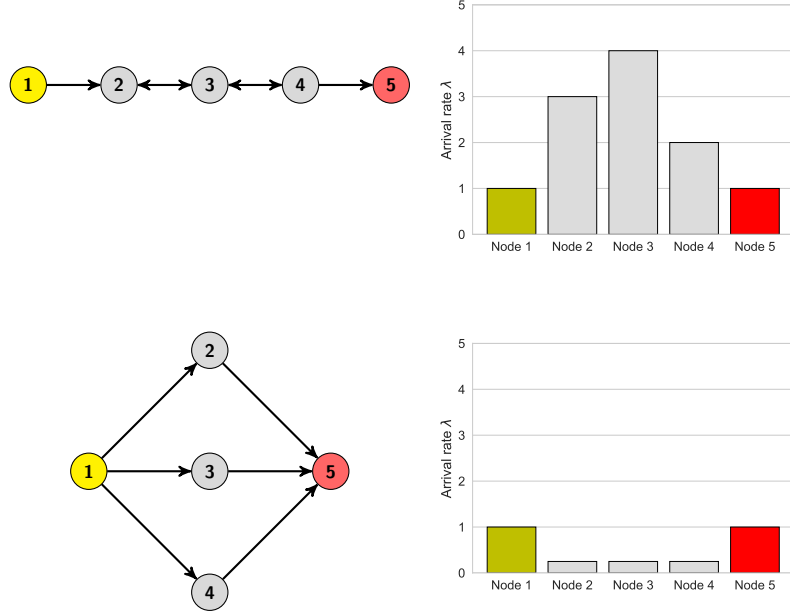


Figure 4.3. Examples of (top) a ‘bad’ network (with large  $Q$ ) and (bottom) a ‘good’ network (with smaller  $Q$  than the ‘bad’ network). The source and sink nodes are coloured yellow and red, respectively. The bar chart depicts the arrival rates for each node.

For any given graph, there are often edges that one can add to or remove from the graph to reduce  $Q$  (unless the graph is optimal). For example, in Figure 4.4, removing the dotted edges in the graph reduces the arrival rates of the intermediate nodes 2, 3, and 4; and it hence reduces  $Q$  for any values of  $\mu_i > 1$ . The dotted edges cause random walkers to take extra steps on these intermediate nodes, thereby increasing the mean journey time and the total mean queue size. Similarly, adding edges can reduce  $Q$ . For example, if we add an edge from node 1 to node 3 of the ‘bad’ network in Figure 4.3, we reduce the arrival rate of  $\lambda_2$  (see Figure 4.5). The edge from 1 to 3 is a shortcut, and walkers reach the sink faster.



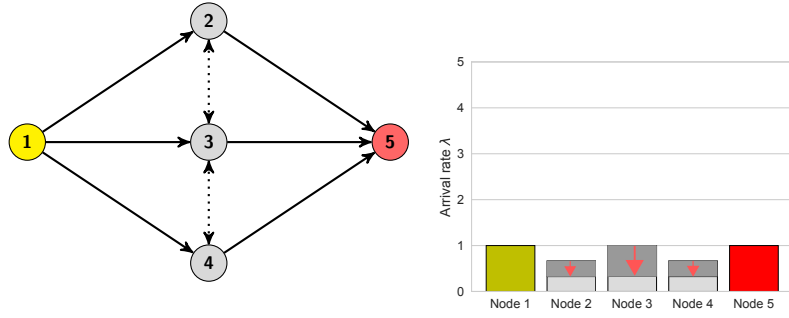


Figure 4.4. Example of (left) a network for which removing the dotted edges decreases the arrival rates and hence  $Q$  for any  $\mu_i > 1$ . The decrease in arrival rates is indicated by the dark grey areas (and red arrows) in the right plot.

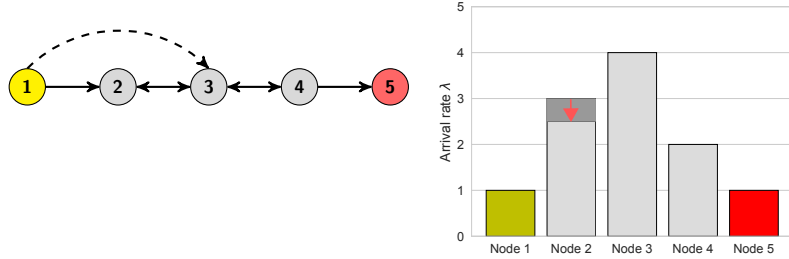


Figure 4.5. Example of (left) a network for which adding the dashed edge decreases the arrival rate of node 2 (while preserving the arrival rates of the other nodes). This reduces  $Q$  for sufficiently large service rates  $\mu_i$ . The decrease in arrival rates is indicated by the dark grey area (and red arrow) in the right plot.

### 4.3 Network topologies that minimize total mean queue size $Q$

We now explore which network topologies in  $\mathcal{C}_n$  (or in some subset of  $\mathcal{C}_n$ ) minimize the total mean queue length  $Q$  when all nodes have the same service rate  $\mu$  (i.e.,  $\mu_i = \mu$  for all  $i$ ). As  $\lambda_n = 1$ , we only consider service rates  $\mu > 1 = \lambda_n$ , as otherwise Equation (2.43) is not satisfied for any queueing network (and thus no stationary state exists for any queueing network). For a fixed  $\mu > 1$ , we calculate the total mean queue size  $Q$  of a network  $G \in \mathcal{C}_n$  using

$$Q = Q(G, \mu) = \begin{cases} \sum_{i=1}^n \frac{\lambda_i}{\mu - \lambda_i}, & \text{if } \mu > \lambda_i \text{ for all } i, \\ \infty, & \text{otherwise,} \end{cases} \quad (4.19)$$

where  $\lambda_i = \lambda_i(G)$  is the arrival rate of node  $i$  in  $G$  given by Equations (4.9) and (4.15). In other words, if  $\lambda_i < \mu$  for all  $i$ , the quantity  $Q$  is the total mean queue size of

$G$ , as given by Equation (4.17). Otherwise, we set it to  $\infty$ , as there is some node  $i$  whose arrival rate  $\lambda_i$  exceeds the service rate  $\mu$  (again note that the arrival rates are independent of  $\mu$ ), so no stationary state exists.

For any collection of graphs  $\mathcal{G} \subset \mathcal{C}_n$ , a network  $G \in \mathcal{G}$  is *Q-optimal over  $\mathcal{G}$*  if, for any  $\mu > 1$  and for all graphs  $G' \in \mathcal{G}$ , we have  $Q(G, \mu) \leq Q(G', \mu)$ . In other words,  $G$  is *Q-optimal over  $\mathcal{G}$* , if for *any* service rate  $\mu > 1$ , there is no other graph  $G' \in \mathcal{G}$  with strictly lower total mean queue size.

We consider the following collections of graphs  $\mathcal{G}$ :

- $\mathcal{C}_n$ : No (additional) constraints.
- $\bar{\mathcal{C}}_n = \{G = (V, E) \in \mathcal{C}_n : (1, n) \notin E\}$ : No edge from node 1 to node  $n$ .
- $\mathcal{U}_n$ : Undirected networks in  $\mathcal{C}_n$ .

We mainly consider  $\bar{\mathcal{C}}_n$  for technical reasons, as the extra condition allows us to provide an analytical proof for *Q-optimality over  $\bar{\mathcal{C}}_n$* .

In the next subsection, we prove that the network  $G_{\bar{\mathcal{C}}_n}$  in Figure 4.6 is *Q-optimal over  $\bar{\mathcal{C}}_n$* . We summarize the statement in Theorem 1, and we present the proof in Section 4.3.1. Our proof provides a method for proving *Q-optimality* for single-source, single-sink open queueing networks with unbiased random walkers. The proof also extends to other types of queues, such as two-server queues, as we describe in Section 4.3.3. Note, however, that  $G_{\bar{\mathcal{C}}_n}$  is not a practical supermarket layout, because a walker has to leave the store after visiting one zone (other than the entrance). In Section 4.3.2, we discuss two properties of *Q-optimal* networks and establish the relationship between  $Q$ , the maximum arrival rate  $\lambda_{\max}$ , and the total arrival rate  $\lambda_{\text{total}} = \sum_i \lambda_i$ . For  $\mathcal{C}_n$ , we conjecture that the network  $G_{\mathcal{C}_n}$  in Figure 4.7 is *Q-optimal over  $\mathcal{C}_n$*  for all  $n$ . We have no proof, but we give numerical evidence in Section 4.3.4. (We have numerically verified this statement for  $n \leq 6$ .) For undirected networks, we show in Section 4.3.5 that there may not exist *Q-optimal* networks. We show for  $n = 5, 6, 7$  that the optimal network (i.e., those with the lowest values of  $Q$ ) depends on the value of  $\mu$ .

**Theorem 1.** *The network  $G_{\bar{\mathcal{C}}_n} = (V, E)$  with  $n \geq 3$  nodes and  $E = \{(1, i) : i = 2, \dots, n-1\} \cup \{(i, n) : i = 2, \dots, n-1\}$  (depicted in Figure 4.6) is *Q-optimal over  $\bar{\mathcal{C}}_n$* .*

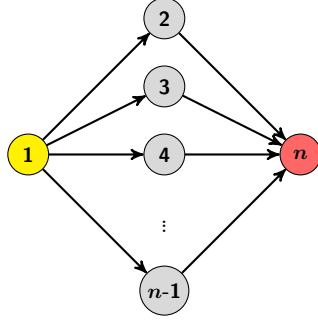


Figure 4.6.  $G_{\bar{C}_n}$ : The  $Q$ -optimal network over  $\bar{C}_n$ .

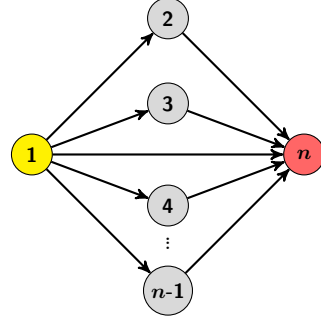


Figure 4.7.  $G_{C_n}$ : Conjectured  $Q$ -optimal network over  $C_n$ .

### 4.3.1 Proof of Theorem 1

We start with two lemmas, which we use in the proof of Theorem 1.

**Lemma 2.** *For any open queueing network  $G \in \mathcal{C}_n$ , the arrival rates  $\lambda_i$  of the nodes of  $G$  satisfy the following inequalities and equations:*

$$\lambda_1 \geq 1, \quad (4.20)$$

$$\lambda_i > 0, \quad i = 2, \dots, n-1, \quad (4.21)$$

$$\lambda_n = 1. \quad (4.22)$$

*Proof.* Equation (4.20) follows from Equation (4.5) and the non-negativity of  $\mathbf{P}$  and  $\lambda_i$ . We verified Equations (4.21) and (4.22) in Section 4.1.1; they correspond to Equations (4.8) and (4.9).  $\square$

**Lemma 3.** *For any network  $G \in \bar{C}_n$ , the arrival rates  $\lambda_i$  of the nodes of  $G$  satisfy*

$$\sum_{i=2}^{n-1} \lambda_i \geq 1. \quad (4.23)$$

*Proof.* First, note that because  $(1, n) \notin E$ , walkers can visit only nodes  $2, \dots, n-1$  from node 1 in the next step (and not node  $n$ ), so

$$\sum_{i=2}^{n-1} P_{1i} = \sum_{i=2}^n P_{1i} = 1. \quad (4.24)$$

The arrival rate  $\lambda_1$  of node 1 is equal to the rate of departure at stationarity. Each customer who departs from 1 goes to one of the intermediate nodes  $2, \dots, n-1$ , so

the sum of the arrival rates of all intermediate nodes  $2, \dots, n-1$  must exceed  $\lambda_1$ . Finally,  $\lambda_1 \geq 1$  by Equation (4.20), as required.

To prove this more formally, we sum the traffic equations (4.5) over  $i = 2, \dots, n-1$  to obtain

$$\sum_{i=2}^{n-1} \lambda_i = \sum_{i=2}^{n-1} \sum_{j=1}^{n-1} \lambda_j P_{ji} \quad (4.25)$$

$$= \sum_{i=2}^{n-1} \left( \lambda_1 P_{1i} + \sum_{j=2}^{n-1} \lambda_j P_{ji} \right) \quad (4.26)$$

$$= \sum_{i=2}^{n-1} \lambda_1 P_{1i} + \sum_{i=2}^{n-1} \sum_{j=2}^{n-1} \lambda_j P_{ji} \quad (4.27)$$

$$= \lambda_1 + \sum_{i=2}^{n-1} \sum_{j=2}^{n-1} \lambda_j P_{ji} \quad (4.28)$$

$$\geq \lambda_1 \quad (4.29)$$

$$\geq 1, \quad (4.30)$$

where we used Equation (4.5) in (4.26) and Equation (4.24) in (4.28). For the final step, we use Equation (4.20).  $\square$

The arrival rates  $\bar{\lambda}_{i,\text{opt}}$  of the nodes of  $G_{\bar{\mathcal{C}}_n}$  are

$$\bar{\lambda}_{i,\text{opt}} = \begin{cases} 1, & \text{if } i = 1 \text{ or } i = n, \\ \frac{1}{n-2}, & \text{if } i = 2, \dots, n-1, \end{cases} \quad (4.31)$$

with a total mean queue size

$$Q(G_{\bar{\mathcal{C}}_n}, \mu) = \bar{Q}_{\text{opt}} = \frac{2}{\mu-1} + \frac{1}{\mu-1/(n-2)}. \quad (4.32)$$

We will show that  $(\bar{\lambda}_{1,\text{opt}}, \dots, \bar{\lambda}_{n,\text{opt}})$  minimize  $Q$  over the space of all possible values of arrival rates  $\lambda_i$  of graphs in  $\bar{\mathcal{C}}_n$ . To do this, we prove a more general statement.

**Proposition 4.** *Fix  $\mu > 1$  and let  $C$  be a constant such that  $C \in [0, (n-2)\mu)$ . Let  $\Omega_{C,\mu}$  be the set of vectors  $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_{n-1}) \in \mathbb{R}^{n-1}$  that satisfy*

$$1 \leq \lambda_1 < \mu, \quad (4.33)$$

$$0 < \lambda_i < \mu, \quad i = 2, \dots, n-1, \quad (4.34)$$

$$\sum_{i=2}^{n-1} \lambda_i \geq C. \quad (4.35)$$

Let  $f(\boldsymbol{\lambda}) : \Omega_{C,\mu} \rightarrow \mathbb{R}$  be a function defined on  $\Omega_{C,\mu} \subset \mathbb{R}^{n-1}$ . Suppose there exists a non-decreasing function  $g : [1, \mu) \rightarrow \mathbb{R}$  and a non-decreasing, convex, and differentiable function  $h : (0, \mu) \rightarrow \mathbb{R}$  such that

$$f(\boldsymbol{\lambda}) = g(\lambda_1) + \sum_{i=2}^{n-1} h(\lambda_i), \quad (4.36)$$

It then follows that  $f$  is minimized on  $\Omega_{C,\mu}$  when

$$\lambda_i = \begin{cases} 1, & \text{for } i = 1, \\ \frac{C}{n-2}, & \text{for } i = 2, \dots, n-1. \end{cases} \quad (4.37)$$

In other words, for any  $\boldsymbol{\lambda} \in \Omega_{C,\mu}$ , we have that

$$f(\boldsymbol{\lambda}) \geq g(1) + (n-2)h\left(\frac{C}{n-2}\right). \quad (4.38)$$

The main outline of the proof of Proposition 4 is as follows. First, observe that we can decouple the problem and minimize  $g(\lambda_1)$  and  $\sum_{i=2}^{n-1} h(\lambda_i)$  separately. The quantity  $g(\lambda_1)$  is minimized when  $\lambda_1$  is minimized, because  $g$  is non-decreasing. Therefore,  $g$  is minimized when  $\lambda_1 = 1$ . For  $\lambda_2, \dots, \lambda_{n-1}$ , we note that their sum must be at least  $C$  by Equation (4.35). We then show the value of  $\lambda_i$  must be the same for all  $i$  at a minimizer of  $\sum_{i=2}^{n-1} h(\lambda_i)$ . For this, we use the symmetry and the convexity of  $h$  and the constraints on  $\lambda_i$ . Therefore, we only need to minimize  $h(\lambda_i)$  subject to  $\lambda_i \geq C/(n-2)$ . Because  $h$  is non-decreasing, it follows that  $\lambda_i = C/(n-2)$ , as required.

*Proof.* Because  $g$  is non-decreasing, we have

$$g(\lambda_1) \geq g(1) \quad (4.39)$$

for all  $\lambda_1 \geq 1$  (by Equation (4.33)).

By convexity of  $h$ , for any  $\lambda, a \in (0, \mu)$ , we have

$$h(\lambda) \geq h(a) + h'(a)(\lambda - a). \quad (4.40)$$

Using  $a = \lambda_{\text{opt},2} = C/(n-2) \in (0, \mu)$  and  $\lambda = \lambda_i$  in Equation (4.40) and summing over  $i = 2, \dots, n-1$ , we have

$$\begin{aligned} \sum_{i=2}^{n-1} h(\lambda_i) &\geq (n-2)h\left(\frac{C}{n-2}\right) + h'\left(\frac{C}{n-2}\right) \left(\sum_{i=2}^{n-1} \lambda_i - C\right) \\ &\geq (n-2)h\left(\frac{C}{n-2}\right), \end{aligned} \quad (4.41)$$

because  $h'(C/(n-2)) \geq 0$  (recall that  $h$  is non-decreasing) and  $\sum_{i=2}^{n-1} \lambda_i \geq C$  (by Equation (4.35)).

Combining Equations (4.39) and (4.41), we have for  $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_{n-1})$  that

$$f(\boldsymbol{\lambda}) = g(\lambda_1) + \sum_{i=2}^{n-1} h(\lambda_i) \geq g(1) + (n-2)h\left(\frac{C}{n-2}\right), \quad (4.42)$$

as required.  $\square$

**Proof of Theorem 1.** Fix  $\mu > 1$ . To ensure that the total mean queue size is bounded, we consider networks  $G \in \bar{\mathcal{C}}_n$  for which the arrival rates  $\lambda_i$  satisfy  $\lambda_i < \mu$ . By Lemmas 2 and 3, the arrival rates  $\lambda_i$  of the nodes in any such network  $G$  satisfy  $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_{n-1}) \in \Omega_{1,\mu}$  (i.e.,  $\Omega_{C,\mu}$  with  $C = 1$ ) and  $\lambda_n = 1$ .

We can write the total mean queue size  $Q$  of each graph  $G$  as a function of the arrival rates  $\lambda_i$ :

$$Q(G, \mu) = f(\boldsymbol{\lambda}) + \frac{1}{\mu - 1}, \quad (4.43)$$

where  $f(\boldsymbol{\lambda})$  is given by Equation (4.36) with  $g(\lambda) = h(\lambda) = \lambda/(\mu - \lambda)$ . Note that  $g$  is non-decreasing, convex, and differentiable on  $(0, \mu)$ .

By Proposition 4, we have

$$\begin{aligned} f(\boldsymbol{\lambda}) &\geq g(1) + (n-2)h(C/(n-2)) \\ &= \frac{1}{\mu - 1} + \frac{1}{\mu - 1/(n-2)}. \end{aligned} \quad (4.44)$$

Therefore, the total mean queue size  $Q$  of any graph  $G \in \bar{\mathcal{C}}_n$  satisfies

$$\begin{aligned} Q(G, \mu) &\geq \frac{1}{\mu - 1} + \frac{1}{\mu - 1/(n-2)} + \frac{1}{\mu - 1} \\ &= \bar{Q}_{\text{opt}}, \end{aligned} \quad (4.45)$$

so  $G_{\bar{\mathcal{C}}_n}$  is  $Q$ -optimal over  $\bar{\mathcal{C}}_n$ .  $\square$

### 4.3.2 Properties of $Q$ -optimal networks

In the next two propositions, we note two properties (i.e., necessary conditions) of  $Q$ -optimal networks. These two propositions relate  $Q$  with the maximum arrival rate  $\lambda_{\max} := \max_i \lambda_i$  and the total arrival rate  $\lambda_{\text{total}} := \sum_i \lambda_i$ .

**Proposition 5.** *Suppose  $G$  minimizes  $Q$  over all graphs in some collection  $\mathcal{G}$  of graphs for all values of  $\mu$  with  $\mu \leq \mu_{\max}$  for some  $\mu_{\max} > \lambda_{\max}(G)$ , where  $\lambda_{\max}(G)$  is the maximum arrival rate of  $G$ . Then*

$$\lambda_{\max}(G) \leq \lambda_{\max}(G') \quad (4.46)$$

for all  $G' \in \mathcal{G}$ . In other words,  $G$  minimizes the maximum arrival rate  $\lambda_{\max}$  over  $\mathcal{G}$ . Consequently, if  $G$  is  $Q$ -optimal over  $\mathcal{G}$ , then  $G$  minimizes the maximum arrival rate  $\lambda_{\max} := \max_i \lambda_i$  over all graphs in  $\mathcal{G}$ .

*Proof.* We prove this statement by contradiction. Suppose that  $G$  minimizes  $Q$  for all values of  $\mu$  that satisfy  $\mu \leq \mu_{\max}$  with  $\mu_{\max} > \lambda_{\max}(G)$  and suppose  $G$  does not minimize  $\lambda_{\max}$  over  $\mathcal{G}$ . There is then a graph  $G'$  with strictly lower maximum arrival rate  $\lambda'_{\max} < \lambda_{\max}$ . Choose any  $\mu \in (\lambda'_{\max}, \lambda_{\max}]$ . Let  $Q$  and  $Q'$  be the total mean queue sizes of  $G$  and  $G'$ , respectively. Because  $G$  minimizes  $Q$  when  $\mu < \lambda_{\max} \leq \mu_{\max}$ , it follows that  $Q \leq Q'$ . However, the total mean queue size  $Q$  of  $G$  is infinite (because  $\mu < \lambda_{\max}$ , whereas the total mean queue size of  $G'$  is finite (because  $\mu > \lambda'_{\max}$ ), so  $\infty = Q > Q'$ , which is impossible.  $\square$

Note that the converse of Proposition 5 only holds when there is a unique network that minimizes the maximum arrival rate  $\lambda_{\max}$ . In this case, a network that minimizes  $Q$  for sufficiently small<sup>4</sup> values of  $\mu$  also (uniquely) minimizes  $\lambda_{\max}$ . When there are multiple networks that minimize the maximum arrival rate  $\lambda_{\max}$ , the converse of Proposition 5 is not true in general. In other words, there may exist a network  $G$  that minimizes the maximum arrival rate  $\lambda_{\max}$ , but does not minimize  $Q$  for all values of  $\mu \leq \mu_{\max}$ , where  $\mu_{\max} > \lambda_{\max}(G)$ . For example, there are two networks that minimize  $\lambda_{\max}$  over  $\mathcal{U}_5$  (the set of undirected networks with  $n = 5$  nodes that satisfy the reachability conditions). The first network (see Figure 4.8a) has arrival rates (1.2, 0.3, 0.3, 0.8, 1), whereas the second (see Figure 4.8b) has arrival rates (1.2, 0.2, 0.2, 0.8, 1). The arrival rates of the first network are equal or larger than arrival rates of the second network, so it has higher values  $Q$  for all values of  $\mu > 1.2$ . Therefore, for any  $\mu_{\max} > 1.2$ , the first network does not minimize the  $Q$  for  $\mu \in (1.2, \mu_{\max}]$  and hence it does not minimize  $Q$  for all values of  $\mu \leq \mu_{\max}$ .

**Proposition 6.** *A graph  $G$  minimizes  $Q$  over all graphs in some collection  $\mathcal{G}$  of graphs for all values of  $\mu$  such that  $\mu > \mu_{\min}$  for some  $\mu_{\min} > 0$  if and only if*

$$\lambda_{\text{total}}(G) \leq \lambda_{\text{total}}(G') \quad (4.47)$$

for all  $G' \in \mathcal{G}$ , where  $\lambda_{\text{total}}(G)$  is the total arrival rate of  $G$ . In other words,  $G$  minimizes  $Q$  for all sufficiently large values of  $\mu$  if and only if  $G$  minimizes  $\lambda_{\text{total}}$ .

<sup>4</sup>In this case, sufficiently small values of  $\mu$  refers to all values  $\mu \leq \mu_{\max}$  for some  $\mu_{\max} > \lambda_{\max}(G)$ . We require a lower bound on  $\mu_{\max}$ , because when you take  $\mu$  smaller than the maximum arrival rate of every graph in  $\mathcal{G}$ , every graph in  $\mathcal{G}$  has infinite  $Q$  and thus every graph minimizes  $Q$  over  $\mathcal{G}$  for sufficiently small  $\mu$ .

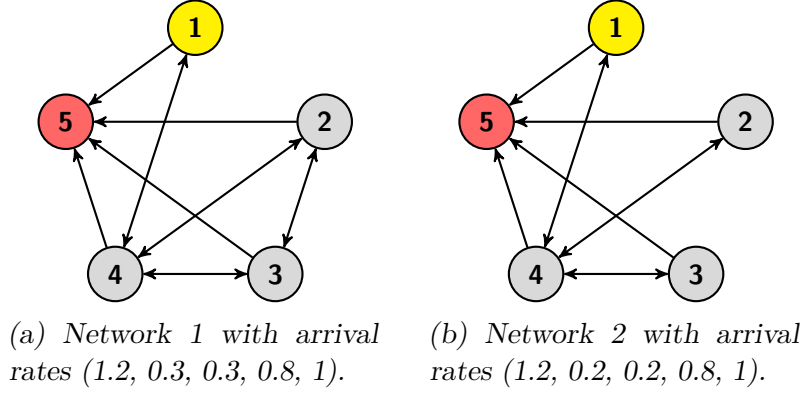


Figure 4.8. The two networks that minimize  $\lambda_{\max}$  over  $\mathcal{U}_5$ .

Consequently, if  $G$  is  $Q$ -optimal over  $\mathcal{G}$ , then  $G$  minimizes the total arrival rate  $\lambda_{\text{total}} := \sum_i \lambda_i$  over all graphs in  $\mathcal{G}$ .

*Proof.* For large  $\mu$ , we have that  $\mu - \lambda_i \approx \mu$  for any node  $i$ , as the arrival rate  $\lambda_i$  is independent of  $\mu$ . Consider the objective function  $\mu Q$ . Note that for a fixed value  $\mu$ , a network minimizes  $Q$  if and only if it minimizes  $\mu Q$ .

For large  $\mu$ , we have

$$\begin{aligned}
 \mu Q &= \mu \sum_i \frac{\lambda_i}{\mu - \lambda_i} \\
 &= \sum_i \lambda_i + \mathcal{O}\left(\frac{\sum_i \lambda_i^2}{\mu}\right) \\
 &= \lambda_{\text{total}} + \mathcal{O}\left(\frac{\sum_i \lambda_i^2}{\mu}\right),
 \end{aligned} \tag{4.48}$$

which tends to  $\lambda_{\text{total}}$  as  $\mu \rightarrow \infty$ . Therefore, if  $G$  does not minimize  $\lambda_{\text{total}}$ , there is some  $G'$  with a smaller value of  $\lambda'_{\text{total}}$  and hence a smaller value of  $\mu Q$  (and thus  $Q$ ) when  $\mu$  is large, which is a contradiction. Conversely, if  $G$  does not minimize  $Q$  for sufficiently large values of  $\mu$ , it does not minimize  $\mu Q$  for sufficiently large values of  $\mu$  and hence does not minimize  $\lambda_{\text{total}}$  by Equation (4.48).  $\square$

Propositions 5 and 6 describe the properties of  $Q$ -optimal networks (over some collection  $\mathcal{G}$  of graphs) at extreme values of  $\mu$ . To minimize  $Q$  in the *small- $\mu$  regime* (i.e., when  $\mu$  is small), a  $Q$ -optimal network must have the smallest  $\lambda_{\max}$  among all graphs in  $\mathcal{G}$ . To minimize  $Q$  in the *large- $\mu$  regime* (i.e., when  $\mu$  is sufficiently large), a  $Q$ -optimal network must have the smallest  $\lambda_{\text{total}}$  among all graphs in  $\mathcal{G}$ .

Propositions 5 and 6 imply that  $G_{\bar{\mathcal{C}}_n}$  (a  $Q$ -optimal network over  $\bar{\mathcal{C}}_n$ ) minimizes  $\lambda_{\max}$  and  $\lambda_{\text{total}}$  over  $\bar{\mathcal{C}}_n$ . Note that we could have arrived at these results directly using



Lemmas 2 and 3 (without having to first prove the  $Q$ -optimality of  $G_{\bar{\mathcal{C}}_n}$ ).<sup>5</sup>

Furthermore, Proposition 6 states that minimizing  $Q$  in the large- $\mu$  regime is equivalent to minimizing  $\lambda_{\text{total}}$ . While Proposition 5 does not state that minimizing  $Q$  in the small- $\mu$  regime is equivalent to minimizing  $\lambda_{\text{max}}$ , it allows us to identify the possible candidates that minimize  $Q$  in the small- $\mu$  regime; the candidates are the networks that minimize  $\lambda_{\text{max}}$ .

### 4.3.3 Extensions

We can extend our results from Section 4.3.1 to other objective functions and other types of queues.

#### Other objective functions

One can adapt the proof of Theorem 1 to show that  $G_{\bar{\mathcal{C}}_n}$  is a minimizer of any objective function  $f(\lambda_1, \dots, \lambda_{n-1})$  that can be written as  $f(\lambda_1, \dots, \lambda_{n-1}) = g(\lambda_1) + \sum_{i=2}^{n-1} h(\lambda_i)$ , where  $g$  is a non-decreasing function on  $[1, \mu)$  and  $h$  is a non-decreasing, convex, and differentiable function on  $(0, \mu)$ . Examples include the total arrival rate  $\lambda_{\text{total}}$  (where  $g(\lambda) = h(\lambda) = \lambda$ ), the total arrival rates  $\sum_{i=2}^{n-1} \lambda_i$  of the intermediate nodes (where  $g(\lambda) = 0$  and  $h(\lambda) = \lambda$ ), and the total mean queue size when node 1 has a different service rate  $\mu_1$  to the other nodes (where  $g(\lambda) = \lambda/(\mu_1 - \lambda)$  and  $h(\lambda) = \lambda/(\mu - \lambda)$ ). We can also directly show that  $G_{\bar{\mathcal{C}}_n}$  minimizes the maximum arrival rate  $\lambda_{\text{max}}$  and the maximum arrival rate  $\max_{i=2}^{n-1} \lambda_i$  of the intermediate nodes over  $\bar{\mathcal{C}}_n$ . We proved that  $G_{\bar{\mathcal{C}}_n}$  minimizes  $\lambda_{\text{max}}$  in the previous section. To prove that  $G_{\bar{\mathcal{C}}_n}$  minimizes  $\max_{i=2}^{n-1} \lambda_i$  over  $\bar{\mathcal{C}}_n$ , note that the arrival rates  $\lambda_i$  of any graph  $G \in \bar{\mathcal{C}}_n$  satisfy  $\sum_{i=2}^{n-1} \lambda_i \geq 1$  by Lemma 3. Because  $\lambda_i \geq 0$ , we must have that  $\max_{i=2}^{n-1} \lambda_i \geq 1/(n-2)$ . This lower bound is achieved by the arrival rates of  $G_{\bar{\mathcal{C}}_n}$ .

#### Other types of queues

When we use other types of queues, we can also show that under certain conditions the arrival rates  $\lambda_{\text{opt}}$  of  $G_{\bar{\mathcal{C}}_n}$  minimize the total mean queue size over  $\bar{\mathcal{C}}_n$ . We no longer have a single service rate  $\mu_i$  for each node  $i$ , but an infinite-dimensional vector of service rates  $\{\mu_{ik}\}_{k=1,2,\dots}$  for each node  $i$  (see Section 2.3.1), where  $\mu_{ik}$  is the service rate of node  $i$  when there are  $k$  customers in node  $i$ . As before when we considered

<sup>5</sup>Lemma 2 shows that  $\lambda_1 \geq 1$ , which implies that  $\lambda_{\text{max}} \geq 1$ . The network  $G_{\bar{\mathcal{C}}_n}$  achieves  $\lambda_{\text{max}} = 1$ , so it minimizes  $\lambda_{\text{max}}$  over  $\bar{\mathcal{C}}_n$ . For  $\lambda_{\text{total}}$ , we can combine results from Lemmas 2 and 3 to give  $\lambda_{\text{total}} \geq \lambda_1 + 1 + \lambda_n \geq 1 + 1 + 1 = 3$ . This lower bound to  $\lambda_{\text{total}}$  is achieved by  $G_{\bar{\mathcal{C}}_n}$ .

single-server queues, we assume that each node has the same service rates, so  $\mu_{ik} = \mu_{1k}$  for all nodes  $i$  and for all  $k \in \mathbb{Z}_+$  for some constants  $\mu_{1k} > 0$ .

We need to extend our definition of  $Q$ -optimality for general queues, as our current definition of  $Q$ -optimality for homogeneous single-server queues is defined in terms of the homogeneous service rate  $\mu$ . To do this, let  $U$  be

$$U = \sup \left\{ x \geq 0 : \sum_{k=1}^{\infty} \frac{x^k}{\prod_{l=1}^k \mu_{1l}} < \infty \right\}. \quad (4.49)$$

The quantity  $U$  depends only on the service rates  $\mu_{1k}$  and it also corresponds to  $U_i$  which we defined in Equation (2.44). As discussed in Section 2.3.1, a stationary state exists for a given queueing network that satisfies the reachability conditions if the arrival rates  $\lambda_i$  satisfy  $\lambda_i < U$  for all  $i$ . Therefore, for given service rates  $\mu_{1k}$ , the quantity  $U$  represents the supremum<sup>6</sup> value that the arrival rates  $\lambda_i$  can take to ensure that Equation (2.43) is satisfied and a stationary state exists. One can verify that  $U = \mu$  for single-server queues with service rate  $\mu$ . We assume that the service rates are sufficiently large such that  $U > 1$ , as otherwise there exists no stationary state for any network  $G \in \bar{\mathcal{C}}_n$ . Our extended definition for  $Q$ -optimality is as follows: We say a network  $G \in \mathcal{G}$  is *Q-optimal* over a collection  $\mathcal{G}$  of networks if, for any service rates  $\mu_{1k}$  such that  $U > 1$ , the total mean queue size  $Q$  of  $G$  does not exceed the total mean queue size of any graph  $G' \in \bar{\mathcal{C}}_n$ . (As before, we define the total mean queue size  $Q$  of a queueing network  $G \in \bar{\mathcal{C}}_n$  that does not have a stationary state to be infinity.)

Provided that the service rates are sufficiently large (to ensure that there exists a stationary state), the arrival rates  $\lambda_i$  of each node  $i$  in a queueing network depend only on the network topology and are independent of the type of queue. Therefore, using different type of queues does not change the arrival rates. In particular, the arrival rates still satisfy Lemmas 2 and 3. Furthermore, for any network  $G \in \bar{\mathcal{C}}_n$  with finite  $Q$ , we have  $\lambda_i < U$  (as otherwise no stationary state exists, and  $Q = \infty$ ). Consequently, the arrival rates  $\lambda_i$  of any queueing network  $G \in \bar{\mathcal{C}}_n$  satisfy  $\lambda = (\lambda_1, \dots, \lambda_n) \in \Omega_{1,U}$  and  $\lambda_n = 1$ .

We can show that  $G_{\bar{\mathcal{C}}_n}$  is  $Q$ -optimal over  $\bar{\mathcal{C}}_n$  for any type of queue, for which the total mean queue size  $Q$  can be written as

$$Q = f(\lambda_1, \dots, \lambda_{n-1}) = g(\lambda_1) + \sum_{i=2}^{n-1} h(\lambda_i), \quad (4.50)$$

---

<sup>6</sup>As discussed in Section 2.3.1, if  $\lambda_i = U$ , there may or may not exist a stationary state, depending on whether (2.43) is satisfied.

where  $g$  is a non-decreasing function on  $[1, \mu)$  and  $h$  is a non-decreasing, convex, and differentiable function on  $(0, \mu)$ . To show  $Q$ -optimality of  $G_{\bar{\mathcal{C}}_n}$  over  $\bar{\mathcal{C}}_n$ , we use Proposition 4 with  $\mu$  replaced by  $U$ .

For example, the proof extends to queueing networks in which each node is a two-server queue (for which the service rate of each server is  $\mu/2$ ). In this case,  $U = \mu$ .<sup>7</sup> We choose  $g(\lambda) = h(\lambda) = 2\mu\lambda/(\mu^2 - \lambda^2)$ , so the total mean queue size is

$$Q = \sum_{i=1}^n \frac{2\mu\lambda_i}{\mu^2 - \lambda_i^2}. \quad (4.51)$$

Therefore,  $G_{\bar{\mathcal{C}}_n}$  is a network in  $\bar{\mathcal{C}}_n$  that minimizes  $Q$  when all queues are two-server queues.

#### 4.3.4 Numerical evidence for $Q$ -optimality of $G_{\mathcal{C}_n}$ over $\mathcal{C}_n$

When we consider networks in  $\mathcal{C}_n$  (instead of  $\bar{\mathcal{C}}_n$ ), the bound on  $\sum_{i=2}^{n-1} \lambda_i$  given by Equation (4.23) in Lemma 3 does not hold any more. Therefore, to prove  $Q$ -optimality of  $G_{\mathcal{C}_n}$  over  $\mathcal{C}_n$  using the same approach as for  $\bar{\mathcal{C}}_n$ , we need to prove a different bound on  $\sum_{i=2}^{n-1} \lambda_i$ . The conjectured bound is

$$\sum_{i=2}^{n-1} \lambda_i \geq 1 - \frac{1}{n-1} \quad (4.52)$$

for all  $G \in \mathcal{C}_n$ . If Equation (4.52) is true, we can show that  $G_{\mathcal{C}_n}$  is  $Q$ -optimal by following the same steps as in the proof of Theorem 1 except that we use  $C = 1 - 1/(n-1)$  instead of  $C = 1$ . Note that the arrival rates of  $G_{\mathcal{C}_n}$  are

$$\lambda_{i,\text{opt}} = \begin{cases} 1, & \text{if } i = 1 \text{ or } i = n, \\ \frac{1}{n-1} = \frac{C}{n-2}, & \text{if } i = 2, \dots, n-1, \end{cases} \quad (4.53)$$

where  $C = 1 - 1/(n-1)$ . In particular, the arrival rates satisfy  $\sum_i \lambda_{i,\text{opt}} = 1 - 1/(n-1)$ .

We do not have an analytical proof for Equation (4.52), but we have verified Lemma 3 for all networks of size 7 or smaller by exhaustive enumeration. We have also employed a simulated-annealing (SA) algorithm that tries to find a network with minimal  $\sum_{i=2}^{n-1} \lambda_i$  for larger network sizes. The SA algorithm was unable to find any networks with lower values for  $\sum_{i=2}^{n-1} \lambda_i$ . Furthermore, the values of  $\sum_{i=2}^{n-1} \lambda_i$  of the networks that the SA algorithm found were close to  $1 - 1/(n-1)$ . For more details on the SA algorithm and our results of using it, see Appendix B.1.

<sup>7</sup>Therefore,  $U$  is the same as in a single-server queue.

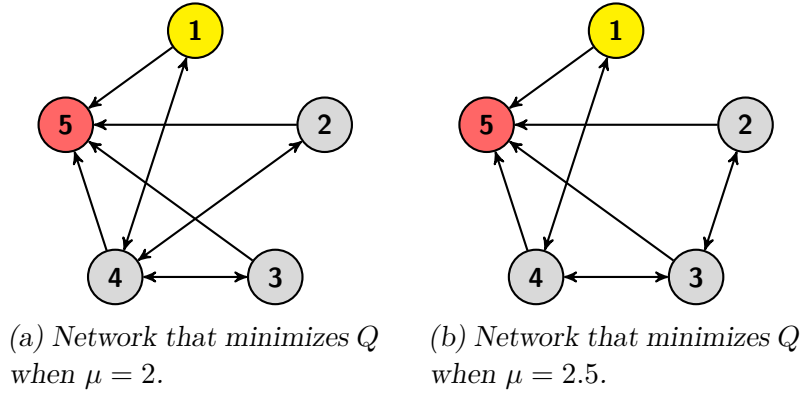


Figure 4.9. Networks that minimize  $Q$  over  $\mathcal{U}_5$  for different values of  $\mu$ .

### 4.3.5 $Q$ -optimality of undirected networks

When we consider  $\mathcal{U}_n$  (i.e., the set of undirected networks with  $n$  nodes that satisfy the reachability conditions) with  $n \geq 5$ , we observe different behaviour than for directed networks. For small network sizes (specifically  $n = 5, 6$ , or  $7$  nodes), we find using exhaustive enumeration that there are no  $Q$ -optimal networks over  $\mathcal{U}_n$ . Instead, there are different networks that minimize  $Q$  depending on the value of the homogeneous service rate  $\mu$ . For example, for  $n = 5$ , the network that minimizes  $Q$  over  $\mathcal{U}_5$  when  $\mu = 2$  (see Figure 4.9a) is different from the network that minimizes  $Q$  when  $\mu = 2.5$  (see Figure 4.9b). (Note we plot all undirected networks using their directed equivalent. See Section 4.1.4 for more details on the equivalence.) For  $n = 6$  and  $n = 7$ , we again find that the network that minimizes  $Q$  depends on the service rate  $\mu$ . (For networks with 3 or 4 nodes, there exists  $Q$ -optimal networks, as we show in Appendix B.3.)

To understand why there may not exist  $Q$ -optimal networks over  $\mathcal{U}_n$  for  $n \geq 5$ , note that Propositions 5 and 6 imply that a  $Q$ -optimal network over  $\mathcal{G}$  can only exist if there exists a network in  $\mathcal{G}$  that minimizes both  $\lambda_{\max}$  and  $\lambda_{\text{total}}$  over  $\mathcal{G}$ . This is not the case when we consider  $\mathcal{G} = \mathcal{U}_n$ , at least for  $n = 5, 6, 7$ . For example, for  $\mathcal{U}_5$ , the network in Figure 4.9b is the unique network that minimizes  $\lambda_{\text{total}}$ . However, this network does not minimize  $\lambda_{\max}$ . Instead, the network in Figure 4.9a is a network<sup>8</sup> that minimizes  $\lambda_{\text{total}}$  with minimum value of  $\lambda_{\max} = 1.2$ , whereas the maximum arrival rate of the network in Figure 4.9b is  $\lambda_{\max} = 1.23 > 1.2$ . For  $n = 3$  and  $n = 4$ , the number of different networks is small, and the same network minimizes both  $\lambda_{\max}$  and  $\lambda_{\text{total}}$ .

<sup>8</sup>The network in Figure 4.9a is not the unique network that minimizes  $\lambda_{\max}$  over  $\mathcal{U}_5$ ; there are two distinct networks (in total) in  $\mathcal{U}_5$  that achieve the minimum value of  $\lambda_{\max} = 1.2$ . We show them in Figure 4.8.

We conjecture that the network  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  (depicted in Figure 4.10) uniquely minimizes  $\lambda_{\text{total}}$  over  $\mathcal{U}_n$  for  $n \geq 3$ . We verified this conjecture by exhaustive enumeration for  $n = 3, \dots, 7$ . For  $n \geq 8$ , we employed an SA algorithm to try to find networks with minimum  $\lambda_{\text{total}}$ . All the networks found by the SA algorithm have larger values of  $\lambda_{\text{total}}$  than  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$ . See Appendix B.1 for more details on the description and the results of the SA algorithms. In Appendix B.2, we show that the network  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  has larger maximum arrival rate  $\lambda_{\text{max}}$  than  $G_{\mathcal{U}_n}^{\lambda_{\text{max}}}$  (depicted in Figure 4.11) for  $n \geq 5$ , so  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  does not minimize  $\lambda_{\text{max}}$  for  $n \geq 5$ . Consequently, if  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  uniquely minimizes  $\lambda_{\text{total}}$  over  $\mathcal{U}_n$  for  $n \geq 5$ , no  $Q$ -optimal networks over  $\mathcal{U}_n$  for  $n \geq 5$  exists, as a  $Q$ -optimal network minimizes both  $\lambda_{\text{total}}$  and  $\lambda_{\text{max}}$ . For  $n = 3$  and  $n = 4$ , the two networks  $G_{\mathcal{U}_n}^{\lambda_{\text{max}}}$  and  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  are the same. In this case, they are the unique  $Q$ -optimal network, as we show in Appendix B.3.

We also conjecture that  $G_{\mathcal{U}_n}^{\lambda_{\text{max}}}$  is a network with minimum  $\lambda_{\text{max}}$ . We verified this conjecture by exhaustive enumeration for  $n = 3, \dots, 7$ . We also employed an SA algorithm to try to find networks with minimum  $\lambda_{\text{max}}$ . All the networks found by the SA algorithm have larger values of  $\lambda_{\text{max}}$  than  $G_{\mathcal{U}_n}^{\lambda_{\text{max}}}$ .

In summary, our results suggest (but do not prove) that there are no  $Q$ -optimal networks over  $\mathcal{U}_n$  for  $n \geq 5$ , as  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  seem to uniquely minimize  $\lambda_{\text{total}}$ , but not  $\lambda_{\text{max}}$  over  $\mathcal{U}_n$ . If our conjecture is true, then  $G_{\mathcal{U}_n}^{\lambda_{\text{max}}}$  minimizes  $Q$  over  $\mathcal{U}_n$  for sufficiently large values of  $\mu$ , but does not minimize  $Q$  for sufficiently small values of  $\mu$ . We conjecture that  $G_{\mathcal{U}_n}^{\lambda_{\text{max}}}$  minimizes  $\lambda_{\text{max}}$  over  $\mathcal{U}_n$  for  $n \geq 5$  and therefore  $G_{\mathcal{U}_n}^{\lambda_{\text{max}}}$  may be a network that minimizes  $Q$  for sufficiently small values of  $\mu$ . For  $n = 3$  or  $4$ , the networks  $G_{\mathcal{U}_n}^{\lambda_{\text{max}}}$  and  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  coincide and we show in Appendix B.3 that they are  $Q$ -optimal networks over  $\mathcal{U}_n$ .

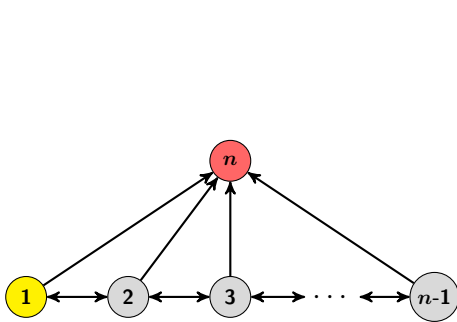


Figure 4.10.  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$ : Conjectured undirected network (with  $n \geq 3$  nodes) that uniquely minimizes  $\lambda_{\text{total}}$  over  $\mathcal{U}_n$ .

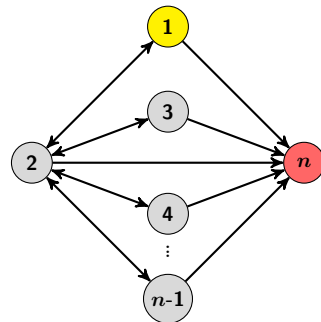


Figure 4.11.  $G_{\mathcal{U}_n}^{\lambda_{\text{max}}}$ : Conjectured undirected network (with  $n \geq 3$  nodes) that minimizes  $\lambda_{\text{max}}$  over  $\mathcal{U}_n$ .

## 4.4 Reducing $Q$ by adding or deleting edges in undirected networks

For most real-world networks, it is typically infeasible to rewire the network to an optimal network (or conjectured optimal network), such as those presented in Section 4.3. Instead, one can make only local perturbations to the original network. The simplest examples of network perturbations are (a small number of) additions or deletions of edges or nodes. In the Internet network (on a router level), for example, adding connections between routers (which are the nodes of the network) is costly for Internet service providers (ISPs), so it is an important task for ISPs to identify the best edges that they can add to improve the traffic capacity. In road networks, some roads are closed (corresponding to edge deletions) at rush hours to alleviate congestion [17].

In this section, we restrict ourselves to edge addition and deletion (which we call *edge perturbations*) and consider the following question:

Given a single-source, single-sink open queueing network  $G = (V, E)$  with unbiased random walkers, how do we add and delete edges from  $G$  to reduce  $Q$ ?

For the supermarket application, we are interested in this question because large changes in a supermarket are generally undesirable and disruptive. Completely changing the layout of a store is costly, as it would be necessary to close the store for some period of time while rearranging the shelves. Furthermore, such a major change may reduce customer satisfaction, because regular customers have to spend more time searching for items in a new, unfamiliar store layout. We do not consider the case of adding nodes, as supermarkets are generally constrained in space and cannot be expanded (or are very costly to expand). We also do not consider removing nodes, as such changes would reduce shelf space, which is undesirable in a supermarket. As store networks are undirected, we present our results on queueing networks with undirected network topology only. Furthermore, existing algorithms for edge deletion on queueing networks have been tested only on undirected graphs [17]. However, our methodology can be adapted in a straightforward way to directed graphs. We consider three scenarios:

Scenario 1: We allow only edge deletions.

Scenario 2: We allow only edge additions.

Scenario 3: We allow both edge deletions and additions.

We propose an efficient greedy method that is applicable for all three scenarios. We test the performance of the algorithms to queueing networks with a network topology given by random-graph models with the source and sink node chosen at random. (We describe the random selection process for the source and sink node in more detail in Section 2.1.3.) We also test the algorithms on a store network. When we generate random graphs, we choose parameters such that the models typically generate sparse networks, in which the number of edges in a graph is much smaller than the number of non-edges, as store networks (and, more generally, spatial networks) are sparse. In Scenario 1 (edge deletions only), our greedy method (which we call the GREEDY ED algorithm) reduces  $Q$  more than previously existing algorithms [67, 110]. In Scenario 2 (edge additions only), we compare our greedy method (called GREEDY EA algorithm in this scenario) with two simpler algorithms: the RESTRICTED GREEDY EA algorithm, in which we only allow adding edges that are incident to the sink node, and the EDGE-TO-SINK-FIRST (ETSF) algorithm, in which we add the edge between the sink node and a node with the largest arrival rate that is not already incident to the sink node. The performance of all three methods are similar, which suggests that adding edges between the sink node and high-arrival-rate nodes gives the largest decrease in  $Q$ . Based on these findings, we combine the GREEDY ED algorithm with the RESTRICTED GREEDY EA algorithm to obtain an efficient algorithm (which we call the RESTRICTED GREEDY EDA algorithm) that both adds and deletes edges from a queueing network to reduce  $Q$  for Scenario 3. This combined greedy method performs better than the GREEDY ED or GREEDY EA algorithms individually.

#### 4.4.1 Mathematical set-up

Given an undirected network  $G = (V, E)$ , we define an *edge perturbation*  $e = \{i, j\}$  on  $G$  to be the process of adding an undirected edge  $e$  to  $G$  if  $e \notin E$  or the process of deleting an undirected edge  $e$  if  $e \in E$ . In the former case,  $e$  is also called an *edge addition*; in the latter case,  $e$  is also called an *edge deletion*. We also define  $C_p : \mathcal{U}_n \rightarrow \binom{V}{2}$  to be the *permissible-edge-perturbation function*. It maps a network  $G \in \mathcal{U}_n$  to the set  $C_p(G)$  of permissible edge perturbations (i.e., the edge perturbations that we allow in a given scenario). Scenarios 1–3 then correspond to the following choices of  $C_p(G)$ :

Scenario 1:  $C_p(G) = E$  (edge deletions only)

Scenario 2:  $C_p(G) = \binom{V}{2} \setminus E$  (edge additions only)

Scenario 3:  $C_p(G) = \binom{V}{2}$  (both edge deletions and additions)

#### 4.4.2 Greedy algorithms

Our greedy algorithms only differ in the choice of permissible-edge-perturbation function  $C_p$ . We present in this subsection the general greedy algorithm for an arbitrary  $C_p$ . The intuition of the greedy algorithm is, at each step, to apply an edge perturbation  $e$  to a queueing network  $G$  that results in the largest decrease in  $Q$ . For each permissible perturbation  $e$  in  $C_p(G)$ , we compute the total mean queue size  $Q'$  of the graph  $G'$  that we obtain after performing the edge perturbation  $e$  on  $G$ . We rank each perturbation in increasing order of  $\Delta Q = Q' - Q$  (the change of  $Q$ ) or, equivalently, in increasing order of  $Q'$ . As in Section 4.3, we define  $Q'$  to be infinity for any network  $G'$  that contains a node  $i$  whose arrival rate  $\lambda_i$  exceeds the service rate  $\mu_i$ . Furthermore, we define  $Q'$  for any  $G'$  that does not satisfy the reachability conditions to be infinity. As described in Section 4.1.4, the reachability conditions are satisfied for an undirected graph  $G$  when both the graph  $G$  and the subgraph of  $G$  that is induced on  $\{1, \dots, n-1\}$  are connected graphs. Therefore, only edge deletions may cause  $G'$  to violate the reachability conditions (assuming the original network  $G$  satisfies the reachability conditions). In our greedy algorithms, we perform  $K$  edge perturbations at each iteration for some integer  $K \geq 1$ . We apply edge perturbations one at a time according to the ranking (in terms of  $\Delta Q$ ) of the edge perturbations, but we skip<sup>9</sup> any edge deletions that would result in a violation of the reachability conditions. After performing  $K$  edge perturbations, we recompute the ranking of each of the permissible edge perturbations  $C_p(G)$  for the current graph  $G$ , and we then apply another  $K$  edge perturbations. We repeat this procedure of applying edge perturbations and recalculating the edge-perturbation rankings until either we have performed  $T$  edge perturbations (for some integer  $T \geq 1$ ) or there are no more edge perturbations that we can apply. In particular, the algorithm stops as soon as  $T$  edge perturbations have been performed or when there are no more edge perturbations that we can apply; it does not complete its current iteration of applying edge perturbations. Therefore, the algorithm applies at most  $T$  edge perturbations. We summarize our general greedy algorithm in Algorithm 1.

When  $K = 1$ , we apply, at each step, the edge perturbation that results in the largest decrease in  $Q$ ; and we recalculate the ranking after each step. Calculating

---

<sup>9</sup>If  $K \geq 1$ , we recompute the ranking after  $K$  edge perturbations. After applying the first edge perturbation, the next edge perturbation may result in the graph violating the reachability conditions, so we skip any that do so.



the rankings of the permissible edge perturbation is computationally expensive, as it requires solving a linear system to obtain  $\Delta Q$  for each permissible edge perturbation in  $C_p(G)$ . Therefore, larger values of  $K$  reduce the number of such calculations and thereby reduce the computational cost. Note, however, that when we use  $K \geq 2$ , the second edge perturbation (and any subsequent ones) in each iteration may not give the largest decrease in  $Q$ , as the ranking of the edges is based on the network before we applied the first edge perturbation. In other words, the ranking of the edge perturbations may change after applying the first edge perturbation, but we still use the ‘old’ ranking in the same iteration. The ‘old’ ranking may still be close to the actual ranking, if we have only applied a small number of edge perturbations. We expect that the algorithm performs worse (i.e., achieves lower decrease in  $Q$ ) when we use larger values of  $K$  and that there is a trade-off between performance and computation time. Note that a greedy algorithm is a heuristic algorithm and is therefore not guaranteed to find an optimal perturbed network for any value of  $K$ . As we show in Sections 4.4.4–4.4.6, the algorithm does, however, achieve a significant decrease in  $Q$  for the random-graph models that we tested. For many practical applications, such as a supermarket setting, it is often not important to find an optimal solution as long as one can find a good solution, which is why heuristic methods are useful.

As mentioned earlier, the computationally expensive part of our general greedy algorithm (Algorithm 1) is to compute  $\Delta Q$  (the change in total mean queue size) for each edge perturbation in  $C_p(G)$ . In the following subsections, we describe a naive algorithm and a more efficient algorithm for calculating  $\Delta Q$ .

### Naive algorithm

In the naive algorithm, we calculate the change  $\Delta Q$  in  $Q$  for an edge perturbation  $e$  on a graph  $G$  by first solving the traffic equations (using Equations (4.9) and (4.15)) to find  $\lambda'_i$  of the perturbed network  $G'$  (i.e.,  $G$  with the edge perturbation  $e$ ). We then calculate  $Q'$  from  $\lambda'_i$  using Equation (4.17) and compute  $\Delta Q = Q' - Q$ , where  $Q$  is the total mean queue size of  $G$ . See Algorithm 2 for the pseudocode of the naive algorithm. The computational complexity of Algorithm 2 is  $\mathcal{O}(|C_p(G)|n^3)$ , as solving the traffic equation requires  $\mathcal{O}(n^3)$  operation and we solve the traffic equations once for each possible edge perturbation in  $C_p(G)$ .

### Efficient algorithm

We now present a more efficient implementation, which requires  $\mathcal{O}(|C_p(G)|n^2 + n^3)$  operations. This implementation relies on using the inverse of the grounded Laplacian

---

**Algorithm 1** General greedy algorithm for reducing  $Q$  using edge perturbations.

---

```

1: procedure GREEDYALGORITHM
2: Input: network  $G = (V, E)$ , permissible-edge-perturbation function  $C_p$ , service
   rates  $\mu_i$ , total number  $T$  of edge perturbations, number  $K$  of edge perturbations
   before recalculating rankings
3: Initialize:
4:   Calculate  $\Delta Q$  for each edge perturbation  $e \in C_p(G)$  (using Algorithm 2 or
   Algorithm 3).
5:   Rank all edge perturbations in increasing order of  $\Delta Q$  and save in list  $R$ .
6:    $num\_edges\_perturbed = 0$ 
7: Algorithm:
8:   while True do
9:      $num\_edges\_perturbed\_in\_loop = 0$ 
10:    while  $num\_edges\_perturbed\_in\_loop < K$  do
11:      Take first edge perturbation  $e$  in  $R$  and remove it from  $R$ .
12:      if edge  $e$  already exists in  $G$  then
13:         $G' \leftarrow G$  with  $e$  removed
14:      else
15:         $G' \leftarrow G$  with  $e$  added
16:        if  $G'$  satisfies reachability conditions then
17:           $G \leftarrow G'$ 
18:          Increment  $num\_edges\_perturbed\_in\_loop$  by 1.
19:          Increment  $num\_edges\_perturbed$  by 1.
20:        else
21:          continue ▷ Skip because not a valid graph
22:        if  $num\_edges\_perturbed \geq T$  or  $R$  is empty then
23:          return  $G$ 
24:      Recalculate  $\Delta Q$  for each edge perturbation  $e \in C_p(G)$  (using Algorithm 2
      or Algorithm 3).
25:      Rank all edge perturbations  $C_p(G)$  in increasing order of  $\Delta Q$  and save in
      list  $R$ .
26: Output: Network  $G$  with up to  $T$  edge perturbations.

```

---

---

**Algorithm 2** Naive algorithm for calculating  $\Delta Q$  for each edge perturbation

---

```

1: procedure CALCULATEDELTAQNAIVE
2: Input: network  $G = (V, E)$ , service rates  $\mu_i$ , and set  $C_p(G)$  of permissible edge
   perturbation
3: Initialize:
4:   Calculate  $\lambda_i$  using Equations (4.9) and (4.15) (traffic equations).
5:   Calculate total mean queue size  $Q$  from  $\lambda_i$  using Equation (4.17).
6:   Initialize empty list  $\Delta Q_{\text{list}}$ .
7: Algorithm:
8:   for each edge perturbation  $e \in C_p(G)$  do
9:     if  $e \in E$  then
10:       $G' \leftarrow G$  with edge  $e$  added
11:     else
12:       $G' \leftarrow G$  with edge  $e$  removed
13:     if  $G'$  does not satisfy reachability conditions then
14:       $Q' \leftarrow \infty$ 
15:     else
16:      Calculate  $\lambda'_i$  using Equations (4.9) and (4.15) (traffic equations).
17:      Calculate total mean queue size  $Q'$  of  $G'$  from  $\lambda'_i$  using Equation (4.17).
18:      Calculate  $\Delta Q = Q' - Q$  and append  $\Delta Q$  to  $\Delta Q_{\text{list}}$ .
19: Output: List  $\Delta Q_{\text{list}}$  of the change  $\Delta Q$  in  $Q$  for each edge perturbation  $e \in C_p(G)$ .

```

---

$\mathbf{L}_{\text{tr}}$  to find the arrival rates  $\lambda_i$  (instead of directly solving the traffic equations), and it uses the Sherman–Morrison formula to calculate the updated inverse of  $\mathbf{L}_{\text{tr}}$  of the perturbed network  $G'$ . The ideas are similar to work in [88], except that we use the grounded Laplacian instead of the combinatorial Laplacian. This procedure is faster than the naive implementation, because we only need to do one calculation (computing the inverse of  $\mathbf{L}_{\text{tr}}$ ) that requires  $\mathcal{O}(n^3)$  operations, instead of performing  $\mathcal{O}(n^3)$  operations (specifically, solving the traffic equations) for each edge perturbation in  $C_p(G)$  (so  $|C_p(G)|\mathcal{O}(n^3)$  operations in total). (Typically,  $|C_p(G)|$  grows at least as fast as  $n$ .) After calculating the inverse of  $\mathbf{L}_{\text{tr}}$ , we can then solve each traffic equation using only a small number of matrix–vector multiplications, using only  $\mathcal{O}(n^2)$  operations for each permissible edge perturbation. Therefore, calculating the  $\Delta Q$  for all edge perturbations requires only  $\mathcal{O}(|C_p(G)|n^2 + n^3)$  operations in total with our efficient implementation. We describe the full procedure below and present the pseudocode in Algorithm 3.

Analogous to Equation (2.12) for  $\mathbf{L}$ , the grounded Laplacian  $\mathbf{L}_{\text{tr}}$  can be expressed as

$$\mathbf{L}_{\text{tr}} = \sum_{e \in E} \tilde{\mathbf{b}}_e \tilde{\mathbf{b}}_e^T, \quad (4.54)$$

where  $\tilde{\mathbf{b}}_e$  is the  $(n-1)$ -dimensional vector that we obtain from  $\mathbf{b}_e$  by deleting the  $n^{\text{th}}$  entry of  $\mathbf{b}_e$ .

Therefore, the grounded Laplacian  $\mathbf{L}'_{\text{tr}}$  of the perturbed network  $G'$  with edge perturbation  $e$  is

$$\mathbf{L}'_{\text{tr}} = \mathbf{L}_{\text{tr}} \pm \tilde{\mathbf{b}}_e \tilde{\mathbf{b}}_e^T, \quad (4.55)$$

where there is a plus when  $e$  is an edge addition and a minus when it is an edge deletion. Using the Sherman–Morrison formula [9], the inverses of  $\mathbf{L}_{\text{tr}}$  and  $\mathbf{L}'_{\text{tr}}$  are related by

$$(\mathbf{L}'_{\text{tr}})^{-1} = \mathbf{L}_{\text{tr}}^{-1} \mp \frac{\mathbf{L}_{\text{tr}}^{-1} \tilde{\mathbf{b}}_e \tilde{\mathbf{b}}_e^T \mathbf{L}_{\text{tr}}^{-1}}{1 \pm \tilde{\mathbf{b}}_e^T \mathbf{L}_{\text{tr}}^{-1} \tilde{\mathbf{b}}_e}. \quad (4.56)$$

The  $\mp$ -sign in Equation (4.56) is a minus when we add  $e$  and a plus when we remove  $e$  and vice versa for the  $\pm$ -sign.

For each perturbed network  $G'$ , we need to calculate the arrival rates  $\lambda'_i$  of its nodes to determine the total mean queue size  $Q'$  of  $G'$ . To do this, we first pre-compute the inverse  $\mathbf{L}_{\text{tr}}^{-1}$  of the grounded Laplacian. For each possible edge perturbation  $e$ , the arrival rates of the nodes of  $G'$  are

$$\begin{aligned} \mathbf{l}'_{\text{tr}} &= \mathbf{D}'_{\text{tr}} \mathbf{L}'_{\text{tr}}{}^{-1} \mathbf{b}_{\text{tr}}, \\ \lambda'_n &= 1, \end{aligned} \quad (4.57)$$

where  $\mathbf{l}'_{\text{tr}} = (\lambda'_1, \dots, \lambda'_{n-1})$ . The matrix  $\mathbf{D}'_{\text{tr}}$  is obtained by deleting the  $n^{\text{th}}$  row and  $n^{\text{th}}$  column from the degree matrix of  $G'$ . We calculate  $(\mathbf{L}'_{\text{tr}})^{-T}$  using Equation (4.56). Finally, we use Equation (4.17) to calculate the total mean queue size  $Q'$  of  $G'$ .

We can adapt our general greedy algorithm to any objective function  $f(\lambda_1, \dots, \lambda_n)$  that depends only on the arrival rates  $\lambda_i$ . Instead of calculating  $Q$  in Lines 6 and 15 of Algorithm 3, we calculate the objective function  $f(\lambda_1, \dots, \lambda_n)$ . For example, we can use the greedy algorithm to reduce the maximum arrival rate  $\lambda_{\max}$  or the total arrival rate  $\lambda_{\text{total}}$  of a graph.

For directed graphs, we can define edge perturbations<sup>10</sup> as additions or deletions of directed edges. In this case, we adapt Equation (2.14) and write

$$\mathbf{L}_{\text{tr}} = \sum_{e \in E} \tilde{\mathbf{a}}_e \tilde{\mathbf{c}}_e^T, \quad (4.58)$$

where  $\tilde{\mathbf{a}}_e$  and  $\tilde{\mathbf{c}}_e$  are vectors of size  $n-1$  obtained from  $\mathbf{a}_e$  and  $\mathbf{c}_e$  (defined in Section 2.1), respectively, by removing the  $n^{\text{th}}$  entry. (The edge set  $E$  consists of directed edges  $e$ .) The equation that corresponds to Equation (4.56) is

$$(\mathbf{L}'_{\text{tr}})^+ = \mathbf{L}_{\text{tr}}^+ \mp \frac{\mathbf{L}_{\text{tr}}^+ \tilde{\mathbf{a}}_e \tilde{\mathbf{c}}_e^T \mathbf{L}_{\text{tr}}^+}{1 \pm \tilde{\mathbf{c}}_e^T \mathbf{L}_{\text{tr}}^+ \tilde{\mathbf{a}}_e}, \quad (4.59)$$

---

<sup>10</sup>So far, we have only defined edge perturbations for undirected graphs.

where we use the Moore–Penrose pseudoinverse  $\mathbf{L}_{\text{tr}}^+$  (instead of the full inverse) of the grounded Laplacian.

---

**Algorithm 3** Calculating  $\Delta Q$  for each edge perturbation (efficient implementation)

---

```

1: procedure CALCULATEDELTAQEFFICIENT
2:   Input: network  $G = (V, E)$ , service rates  $\mu_i$ .
3:   Initialize:
4:     Calculate inverse  $\mathbf{L}_{\text{tr}}^{-1}$ .
5:     Calculate  $\lambda_i$  using Equations (4.9) and (4.15).
6:     Calculate total mean queue size  $Q$  from  $\lambda_i$  using Equation (4.17).
7:     Initialize empty list  $Q'_{\text{list}}$ .
8:   Algorithm:
9:     for each edge  $e \in E$  do
10:       $G' \leftarrow G$  with edge  $e$  removed
11:      if  $G'$  does not satisfy reachability conditions then
12:         $Q' \leftarrow \infty$ 
13:      else
14:        Calculate  $\lambda'_i$  using Equation (4.57).
15:        Calculate total mean queue size  $Q'$  of  $G'$  from  $\lambda'_i$ .
16:        Calculate  $\Delta Q = Q' - Q$  and append  $\Delta Q$  to  $\Delta Q_{\text{list}}$ .
17:      Rank list of edges  $E$  by  $Q'_{\text{list}}$  in increasing order and save sorted list in  $R$ .
18:   Output: List  $\Delta Q_{\text{list}}$  of the change  $\Delta Q$  in  $Q$  for each edge perturbation  $e \in C_p(G)$ .

```

---

The computational cost of calculating the inverse  $\mathbf{L}_{\text{tr}}^{-1}$  is  $\mathcal{O}(n^3)$ , and we only need to calculate this once at each step of our greedy algorithms. For each edge perturbation  $e \in C_p(G)$ , the calculation of  $\lambda'_i$  then involves only matrix–vector multiplications and therefore needs only  $\mathcal{O}(n^2)$  operations. Note that we do not have to explicitly calculate the updated inverse  $(\mathbf{L}'_{\text{tr}})^{-1}$ , as we can substitute Equation (4.56) into Equation (4.57) and calculate  $v'_i$  using only matrix–vector multiplications. Calculating  $Q'$  from  $\lambda'_i$  involves  $\mathcal{O}(n)$  operations. Therefore, the total computational cost of our efficient algorithm (Algorithm 3) is  $\mathcal{O}(|C_p(G)|n^2 + n^3)$ . If  $|C_p(G)| \geq n$ , the computational cost is a factor of  $\mathcal{O}(n)$  smaller than for the naive algorithm (Algorithm 2).

In practice, our efficient implementation is at least 10 times faster than our naive implementation for the networks (with up to  $n = 1000$  nodes) that we tested. For example, the greedy algorithm for edge deletion is able to handle networks with up to  $n = 750$  nodes within a minute of computation on a standard desktop machine, whereas the naive implementation for edge deletion can only handle networks with up to  $n = 200$  nodes in the same time (see Appendix B.4).

### 4.4.3 Random network topologies for queueing networks

We apply our edge-perturbation algorithms to queueing networks with a network topology generated from each of the following six random-graph models (see Section 2.1.3 for their definitions):

1. Barabási–Albert (BA) network with  $n = 100$  nodes and mean degree  $m_{BA} = 3$ .
2. Erdős–Rényi (ER) graph with  $n = 100$  nodes and connection probability  $p = 0.6$ .
3. Random regular graph with  $n = 100$  nodes and node degree  $d = 6$ .
4. Watts–Strogatz graph with  $n = 100$  nodes, parameter  $k_{WS} = 3$ , and rewiring probability  $p = 0.5$ .
5. Random geometric graph (RGG) on a unit square with  $n = 100$  nodes and connection radius  $r = 0.19$ .
6. Chung–Lu model with  $n = 100$  nodes and a degree distribution given by a degree sequence from a BA network<sup>11</sup> with  $n = 100$  nodes and  $m_{BA} = 3$ .

Note that each of the models generate undirected networks. The parameters of the random-graph models ensure that each network has the same number of nodes ( $n = 100$ ) and the same mean number  $\mathbb{E}[m]$  of edges ( $\mathbb{E}[m] = 300$ ). We generate 100 networks from each random-graph model and perform the following steps to each of the network  $G$  to convert it to a queueing network that satisfies the reachability conditions. To ensure that  $G$  is connected, we remove all nodes and incident edges from  $G$  that do not belong to the largest connected component. Therefore, the number of nodes in the graph may be less than 100 and the mean number of edges may be less than 300. We choose a source node  $s$  uniformly at random from all nodes in  $G$ . We then choose a sink node uniformly at random from all *sink-node candidates*. A node  $k$  is a *sink-node candidate* if  $k \neq s$  and removing  $k$  and its incident edges does not disconnect the subgraph of  $G = (V, E)$  that is induced on  $V \setminus \{k\}$ . Note that for  $n \geq 2$ , there is at least one sink-node candidate for any choice of  $s$ . To see this, consider the node  $k$  whose shortest path to  $s$  is of maximum length. Node  $k$  is a sink-candidate node, because for every other node  $j$  a shortest path from  $j$  to  $s$  does not go through  $k$  (otherwise the shortest path from  $j$  to  $s$  is longer than the shortest path from  $k$  to  $s$ ). Therefore, the subgraph of  $G = (V, E)$  that is induced on  $V \setminus \{k\}$  is

---

<sup>11</sup>We use the same parameters as when we use BA networks, but do not use the same, identical networks.

Table 4.1. List of edge-perturbation algorithms

Algorithm	Types of edge perturbation	Description
GREEDY ED	edge deletion	Deletes at each step an edge that reduces $Q$ the most.
HDF	edge deletion	Deletes at each step an edge $(i, j)$ with the largest $d_i \times d_j$ (where $d_k$ is degree of node $k$ ).
HARF	edge deletion	Deletes at each step an edge $(i, j)$ with largest $(\lambda_i \times \lambda_j)$ (where $\lambda_k$ is the arrival rate of $k$ ).
GREEDY EA	edge addition	Adds at each step an edge that reduces $Q$ the most.
RESTRICTED GREEDY EA	edge addition	Adds at each step an edge that reduces $Q$ the most. Only allowed to add edges incident to the sink.
RESTRICTED GREEDY EDA	edge deletion and addition	Adds or deletes at each step an edge that reduces $Q$ the most. Only allowed to add edges incident to the sink.

connected, as required. We must choose a sink node among the sink-node candidates to ensure that the reachability conditions are satisfied in the queueing network. We then relabel all nodes such that node 1 is the source node and node  $n$  is the sink node.

After performing the above steps to each of the networks, we set the homogeneous service rate  $\mu$  to  $3\lambda_{\max}$ , where  $\lambda_{\max}$  is the maximum arrival rate in the network, and perform our edge-perturbation algorithms to each of the networks. We list all the edge-perturbation algorithms that we use in Table 4.1. In Appendix B.6, we present our results when minimizing  $\lambda_{\max}$  and minimizing  $\lambda_{\text{total}}$ . As explained in Section 4.3.2, minimizing  $\lambda_{\text{total}}$  is equivalent to minimizing  $Q$  in the large- $\mu$  regime. Furthermore, a network that minimizes  $Q$  in the small- $\mu$  regime also minimizes  $\lambda_{\max}$ , so minimizing  $\lambda_{\max}$  is a proxy<sup>12</sup> for minimizing  $Q$  in the small- $\mu$  regime. The results that we present in the main text are consistent to the results for when we minimize  $\lambda_{\text{total}}$ . This suggests that a service rate of  $\mu = 3\lambda_{\max}$  is in the large- $\mu$  regime for the networks that we tested. The results for when we minimize  $\lambda_{\max}$  are also qualitatively similar to the results presented here with a few differences, as we explain in Appendix B.6.2.

<sup>12</sup>It is an imperfect proxy, because we are not guaranteed to find a network that minimizes  $Q$  in the small- $\mu$  regime by minimizing  $\lambda_{\max}$ .

#### 4.4.4 Scenario 1: Edge deletions only

We compare Algorithm 1 for edge deletion, which we call the GREEDY ED algorithm, with the high-degree-first (HDF) and high-arrival-rate-first (HARF) algorithms [67, 110]. The HDF and HARF algorithms have been used to decrease congestion in the traffic-dynamics model (which we described in Section 2.2). Both of these algorithms first assign a score to each edge  $\{i, j\}$ . The score of an edge  $\{i, j\}$  is the product  $d_i \times d_j$  of the degrees of  $i$  and  $j$  in the HDF algorithm and is the product  $\lambda_i \times \lambda_j$  of the arrival rates of  $i$  and  $j$  in the HARF algorithm. In both algorithms, edges are ranked in decreasing order of their score and deleted one by one according to this ranking,<sup>13</sup> skipping any edges whose deletion result in the graph not satisfying the reachability conditions.<sup>14</sup> The algorithm stops deleting edges after  $T$  successful edge deletions or if no more edges can be deleted without violating the reachability conditions. Unlike the GREEDY ED algorithm, these two algorithms do not recalculate the ranking of the remaining edges after  $K$  edge deletion; the order in which the edges are deleted is determined at the beginning of the algorithm. In Appendix B.5, we also consider variants of the HDF and HARF algorithms in which we recalculate the ranking of the remaining edges after deleting  $K$  edges. The HDF algorithm was first introduced in [67]. The authors of [67] reported good performance in decreasing congestion (measured by the traffic capacity  $\rho_c$ ) in the traffic-dynamics model (introduced in Section 2.2) for Barabási–Albert networks. The HARF algorithm is an adapted version of the high-betweenness-first (HBF) algorithm from [110]. In the HBF algorithm, the score of each edge  $\{i, j\}$  is  $B_i \times B_j$ , where  $B_i$  and  $B_j$  are the geodesic node betweenness centrality values of their incident nodes  $i$  and  $j$ . The geodesic node betweenness centrality of node  $i$  is equal to the effective betweenness centrality  $B_i^{\text{eff}}$  when we have shortest-path routing in the traffic-dynamics model. As described in Section 2.3.4, the effective betweenness  $B_i^{\text{eff}}$  in the traffic-dynamics model is proportional to its arrival rate  $\lambda_i$  in the queueing network. The idea behind HBF is that, by removing edges between nodes with large betweenness centralities (and therefore with large arrival rates), flow will (hopefully) redistribute from these nodes to nodes with lower arrival rates. Therefore, the corresponding algorithm for single-source, single sink queueing networks is the HARF algorithm, where we rank edges in decreasing order according to  $\lambda_i \times \lambda_j$  instead of  $B_i \times B_j$ .

---

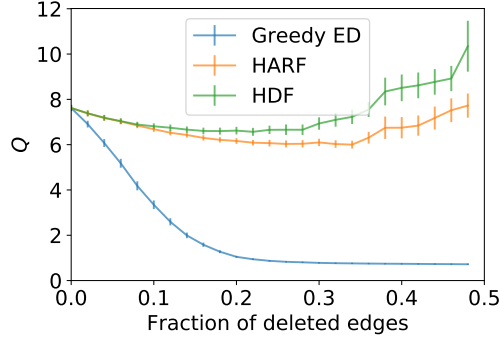
<sup>13</sup>In other words, we first delete the edge  $\{i, j\}$  with the largest value of  $d_i \times d_j$  (HDF) or  $\lambda_i \times \lambda_j$  (HARF).

<sup>14</sup>This happens if the deletion of the edge results in a graph whose subgraph that is induced on the nodes  $\{1, \dots, n-1\}$  is disconnected.

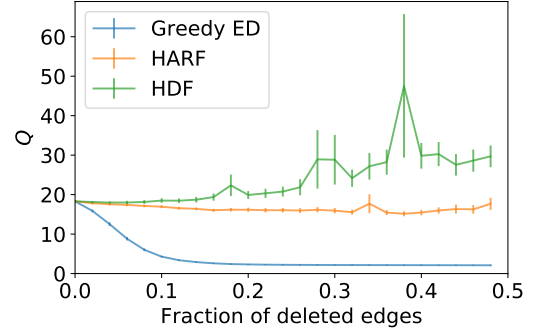


We apply the GREEDY ED, HDF, and HARF algorithms to 100 queueing networks for each of the six random-graph models that we described in Section 4.4.3. We set the total number  $T$  of edges deleted to be  $T = \lceil m/2 \rceil$ , and we recalculate rankings after deleting  $K = \lceil m/50 \rceil$  edges for the GREEDY ED algorithm. In each simulation, we generate a random graph according to the network model and convert it to a queueing network  $G$  as described in Section 4.4.3. We run each of the three algorithms on the queueing network  $G$ . We record the total mean queue size  $Q$  after every  $K$  edge deletions. If the maximum arrival rate  $\lambda_{\max}$  exceeds the service rate  $\mu$ , we set  $Q$  to infinity. We plot the total mean queue size  $Q$ , averaged over all simulations with finite  $Q$  values, as a function of the fraction of edges deleted for the six network models in Figure 4.12. There are some realizations, in which the maximum arrival rate is very close to  $\mu$  — although always smaller than  $\mu$ , so it has large values of  $Q$  — which causes large spikes in the mean and standard error of  $Q$  (e.g., see Figure 4.12f). For all six network models, the GREEDY ED algorithm on average yields networks with lower total mean queue size  $Q$  than the HDF and HARF algorithms. Over the course of the GREEDY ED algorithm, the value of  $Q$  initially descends rapidly, but then plateaus after deleting only 10–20% of the total edges (the precise percentage depends on the network model). We find that HDF does not perform well on these six random-graph models, because it increases  $Q$  on average as we delete more edges. The performance of the HARF algorithm lies between the GREEDY ED algorithm and the HDF algorithm. When we use the HARF algorithm,  $Q$  decreases in regular networks as more edge are deleted according to the HARF ranking, whereas  $Q$  remains approximately the same for the other networks.

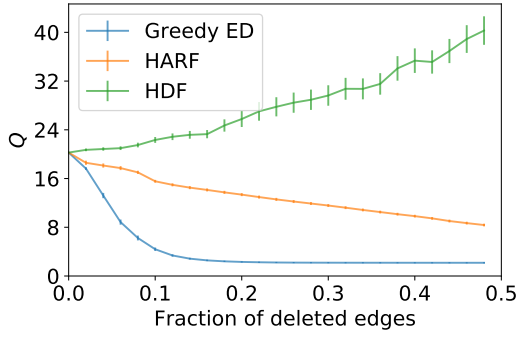
In Appendix B.6, we show that the GREEDY ED algorithm also achieves a larger decrease in  $\lambda_{\max}$  and  $\lambda_{\text{total}}$  than the HDF and HARF algorithms.



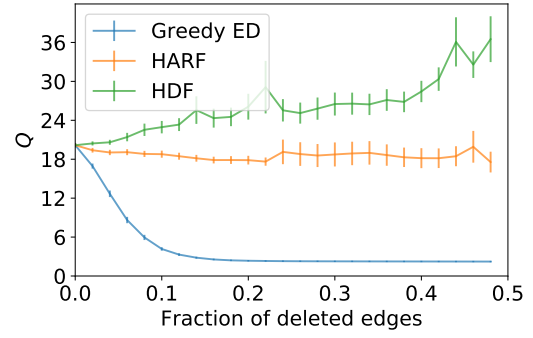
(a) Barabási–Albert network



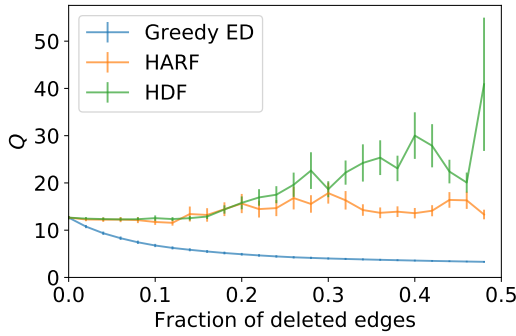
(b) Erdős–Rényi graph



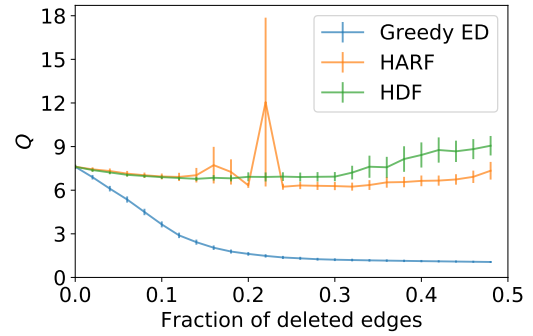
(c) Random regular graph



(d) Watts–Strogatz graph



(e) Random geometric graph



(f) Chung–Lu model

Figure 4.12. Comparison of the performance of the GREEDY ED, HADF, and HARF algorithms in decreasing the total mean queue size  $Q$  with edge deletion. We plot the mean and standard error of  $Q$  as a function of the fraction of edges deleted.

#### 4.4.5 Scenario 2: Edge additions only

We now consider Algorithm 1 for edge addition (i.e., with  $\mathcal{C}_p(G) = \binom{V}{2} \setminus E$ ); we call this the GREEDY EA algorithm. We compare the GREEDY EA algorithm with the RESTRICTED GREEDY EA algorithm and the EDGE-TO-SINK-FIRST (ETSF) algorithm, which we describe below.

The RESTRICTED GREEDY EA algorithm is a variant of the GREEDY EA algorithm. In this variant, we further restrict the permissible edge perturbations to consider only edges that are incident to the sink node. In other words, the RESTRICTED GREEDY EA algorithm is Algorithm 1 with the permissible edge-perturbation function  $\mathcal{C}_p(G) = \{\{i, n\} \notin E : i \neq n\} \subseteq \binom{V}{2} \setminus E$ .

In the ETSF algorithm, we identify all nodes in  $G$  that are not neighbours of the sink node. We then rank these nodes in decreasing order by their arrival rate, where nodes with the same arrival rate are ordered randomly. We then select the first  $K$  nodes in this ranking (i.e., the  $K$  nodes with the largest arrival rates), and we add an edge from each of these nodes to the sink node. The ETSF algorithm stops if  $T$  edges have been added or if every node is a neighbour of the sink node. (The latter situation occurs first if  $T > n - 1 - d_n$ .) The motivation behind ETSF is as follows. Adding an edge from a node  $i$  with a large arrival rate  $\lambda_i$  to the sink node decreases the arrival rate of its neighbouring nodes, as a significant number of walkers go to the sink node (i.e., the tills) instead of going to other neighbouring nodes and contributing to congestion in neighbouring nodes. Furthermore, adding the edge  $\{i, n\}$  also decreases the arrival rate  $\lambda_i$  of node  $i$ . This is because some walkers from  $i$  go to the sink, where they do not return, so fewer walkers traverse the neighbours of  $i$  and thus fewer walkers come (back) from these neighbours.

We apply GREEDY EA, the RESTRICTED GREEDY EA, and ETSF algorithms to the six random-graph models listed in Section 4.4.3. As in Section 4.4.4, we set the simulation parameters to  $K = \lceil m/50 \rceil$  and  $T = \lceil m/2 \rceil$ . The algorithms reduce  $Q$  significantly more than the GREEDY ED algorithm (see Figure 4.13). Furthermore, the reductions in  $Q$  are almost the same for all three algorithms on all six random-graph models.

The above results suggest the following question: Are the three algorithms similar (or even almost identical) in the way they add edges? To answer this question, let  $(g_1, \dots, g_T)$  denote the sequence of edges that the GREEDY EA algorithm added to  $G$ . Similarly, we define  $(e_1, \dots, e_{T_r})$  to be the corresponding sequence for the RESTRICTED GREEDY EA algorithm, where  $T_r$  are the total number of edges that the RESTRICTED GREEDY EA algorithm adds. In all graphs that we considered, we have  $T_r < T$ , so

there are always less than  $T$  nodes that are not adjacent to  $n$  in the original graph. We seek to compare the first  $T_r$  edge additions  $(g_1, \dots, g_{T_r})$  of the GREEDY EA algorithm with the  $T_r$  edge additions of the RESTRICTED GREEDY EA algorithm. To do so, for each  $k = 1, \dots, T_r$ , we calculate the number  $E_k$  of common edges that are in both  $(g_1, \dots, g_k)$  and  $(e_1, \dots, e_k)$ . In other words, the number  $E_k$  is the number of common edges that both the GREEDY EA and RESTRICTED GREEDY EA algorithm have added after  $k$  edge additions. (Note that common edges do not have to occupy the same position in the sequences.) The sequence  $(E_1, \dots, E_{T_r})$  captures the discrepancy between the two algorithms. If  $E_k = k$  for all  $k = 1, \dots, T_r$ , the sequences  $(g_1, \dots, g_{T_r})$  and  $(e_1, \dots, e_{T_r})$  are identical and the two algorithms add the same edges in the same order. If  $E_k = 0$  for all  $k = 1, \dots, T_r$ , the sequences  $(g_1, \dots, g_{T_r})$  and  $(e_1, \dots, e_{T_r})$  have no elements in common, so both algorithms add completely different edges. (In practice,  $E_k$  will typically lie between 0 and  $k$ .) To visually compare the discrepancy between the two algorithms on different graphs (which have different values of  $T_r$ ), we define the function  $f_r$  on  $[0, 1]$  by

$$f_r(x) = \frac{E_{\lfloor xT_r \rfloor}}{T_r}, \quad (4.60)$$

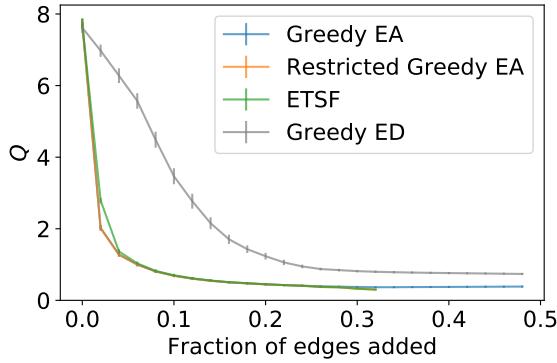
which is a step function that interpolates between the points  $(k/T_r, E_k/T_r)$  for  $k = 1, \dots, T_r$ . The quantity  $f_r(x)$  is equal to the number of common edges that both the GREEDY EA and its restricted variant add after each algorithm performed  $\lfloor xT_r \rfloor$  edge additions divided by  $T_r$ . The function  $f_r$  is an increasing function with  $f_r(x) \in [0, 1]$ . When the two algorithms add the same edges in the same orders,  $f_r(x)$  is a step function that approximates the identity line. If the two algorithms add completely different edges,  $f_r(x) = 0$  for  $x \in [0, 1]$ . We use  $f_r(x)$  instead of  $E_k$  because different graphs have different values of  $T_r$ , so the sequence  $E_1, \dots, E_{T_r}$  have variable lengths with different maximum values. The function  $f_r(x)$  takes values in  $[0, 1]$  for  $x \in [0, 1]$  for all graphs, so it is possible to compare the discrepancy in the added edges between different graphs using  $f_r(x)$ .

Similarly, we calculate the corresponding quantity  $f_{\text{ETSF}}(x)$  for the ETSF algorithm for  $x \in [0, 1]$ , where  $f_{\text{ETSF}}(x)$  is equal to the number of common edges that both the GREEDY EA and RESTRICTED GREEDY EA add after each algorithm performs  $\lfloor xT_r \rfloor$  edge additions divided by  $T_r$ .

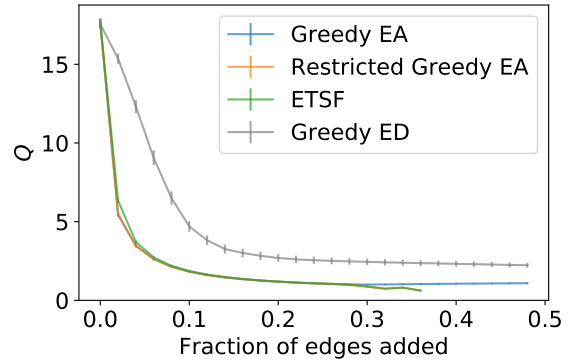
In Figure 4.14, we plot the mean  $f_r(x)$  and the mean  $f_{\text{ETSF}}(x)$  for each of the random-graph models, averaged over 100 simulations. On average, the functions  $f_r(x)$  and  $f_{\text{ETSF}}(x)$  are close to the identity line for all random-graph models, except for RGGs, so the three algorithms largely add the same edges on these graphs. The

function  $f_r(x)$  is closer to the identity line than  $f_{\text{ETSF}}(x)$ , so the GREEDY EA algorithm is (unsurprisingly) more similar to the RESTRICTED GREEDY EA algorithm than to the ETSF algorithm with respect to the edges that the algorithms add. In our RGGs, the GREEDY EA algorithm adds significantly more edges that are distinct from the added edges of both the restricted GREEDY EA and ETSF algorithms than in other random-graph models. Note that at the end of the RESTRICTED GREEDY EA and ETSF algorithm, all nodes are incident to the sink node  $n$  (as  $T_r < T$ ). In other words, the set of all edges that these two algorithms add is the set of edges incident to  $n$  that are not present in the original graph. Therefore,  $f_r(1)$  is equal to the proportion of edges that the GREEDY EA algorithm adds which are not incident to the sink node. The mean value of  $f_r(1)$  and  $f_{\text{ETSF}}(1)$  are both at about 0.8 (i.e., 80%) in RGGs, which implies that 20% of the first  $T_r$  edges that the GREEDY EA algorithm adds are not incident to the sink node. By contrast, the mean of  $f_r(1)$  and  $f_{\text{ETSF}}(1)$  is above 0.98 in the other five random-graph models, so only 2% of the first  $T_r$  added edges by the GREEDY EA algorithm are not incident to the sink node. Despite the GREEDY EA algorithm adding different edges than the RESTRICTED GREEDY EA or the ETSF algorithms for RGGs, the decrease in the value of  $Q$  is almost the same for all three algorithms (see Figure 4.13e).

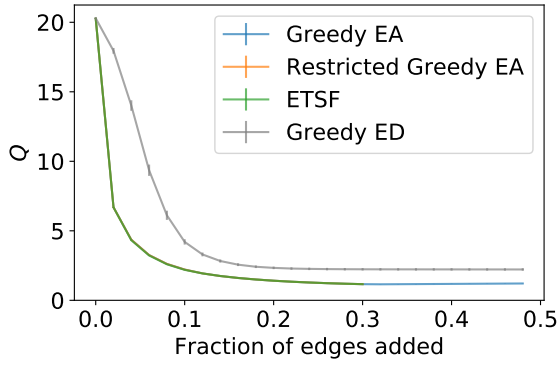
In summary, the GREEDY EA algorithm is effective at decreasing  $Q$  for open queueing networks with network topologies generated from the our six random-graph models and with a random source and sink. We compared the edges that the GREEDY EA adds with the edges that two simpler algorithms (GREEDY EA and ETSF) add, and we found that the added edges of the GREEDY EA algorithm correspond mainly to edges that are incident to the sink node. Most of these edges connect the sink node with a large-arrival-rate node.



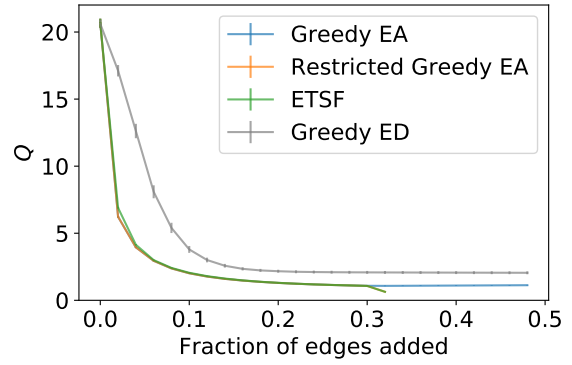
(a) Barabási–Albert network



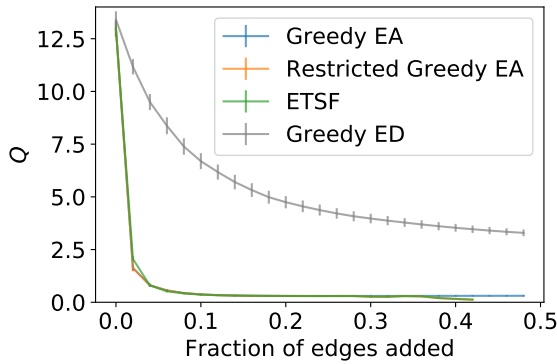
(b) Erdős–Rényi graph



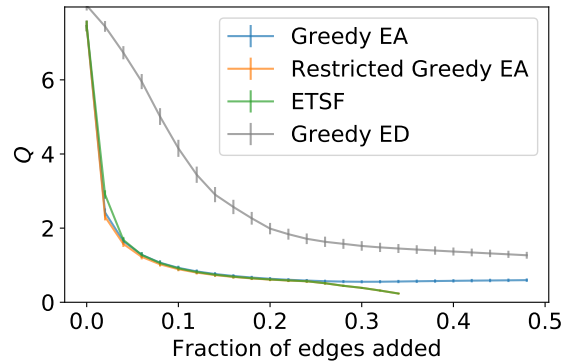
(c) Random regular graph



(d) Watts–Strogatz graph

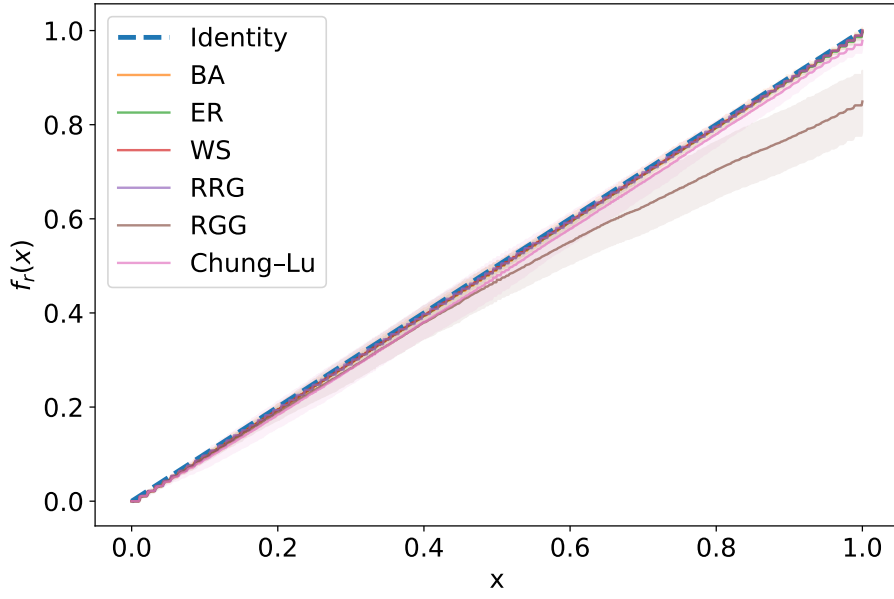


(e) Random geometric graph

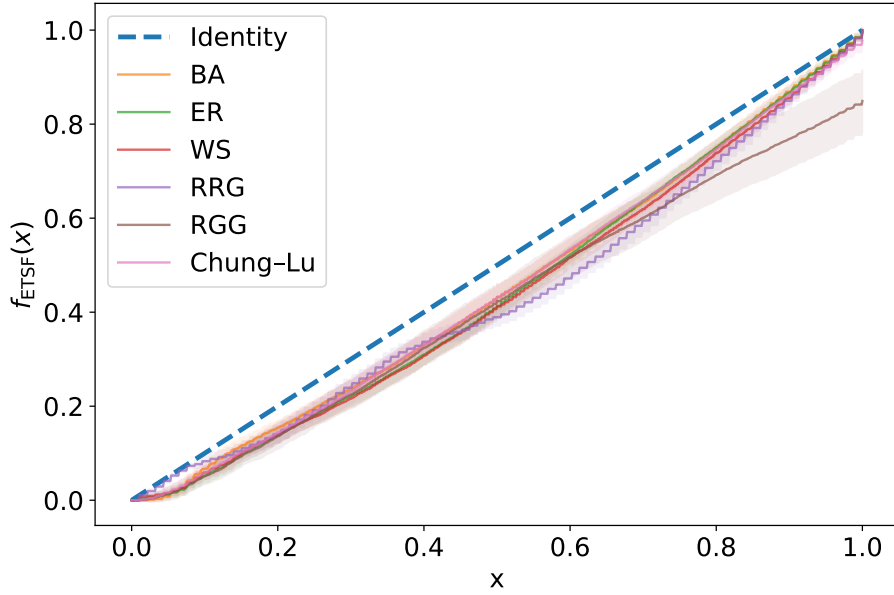


(f) Chung–Lu model

Figure 4.13. Comparison of the performance of the GREEDY EA, RESTRICTED GREEDY EA, and ETSF algorithms in decreasing the total mean queue size  $Q$  with edge addition. We plot the mean and standard error of  $Q$  as a function of the fraction of edges added (i.e., the number of edges added divided by the number of edges in the original graph).



(a) Function  $f_r(x)$  that shows the agreement between the added edges in the GREEDY EA algorithm and the added edges in the RESTRICTED GREEDY EA algorithm.



(b) Function  $f_{\text{ETSF}}(x)$  that shows the agreement between the added edges in the GREEDY EA algorithm and the added edges in the ETSF algorithm.

Figure 4.14. Comparison between the GREEDY EA algorithm with (a) the RESTRICTED GREEDY EA algorithm and (b) the ETSF algorithm. We plot the mean values of  $f_r(x)$  or  $f_{\text{ETSF}}(x)$  (solid curve) in the respective subplot and standard deviation (shaded area) for each of the six random-graph models. We also plot the identity line (dashed) in blue. The mean values of  $f_r(x)$  and  $f_{\text{ETSF}}(x)$  are close to the identity line, so the three algorithms largely add the same edges.

#### 4.4.6 Scenario 3: Edge deletions and edge additions

Based on the insights from applying the different edge-perturbation algorithms in Scenarios 1 and 2, we propose a greedy algorithm, which we call the RESTRICTED GREEDY EDA<sup>15</sup> algorithm, for edge-deletion and edge-addition. The RESTRICTED GREEDY EDA algorithm is Algorithm 1, where we allow only edge perturbations on existing edges or on non-edges that are incident to the sink node; that is,  $C_p(G) = \{\{i, j\} : \{i, j\} \in E \text{ or } j = n \text{ or } i = n\}$ . The algorithm is a combination of the GREEDY ED algorithm and the RESTRICTED GREEDY EA algorithm. In sparse networks, where  $|C_p(G)|$  is much smaller than  $|E|$ , the computation time of this greedy algorithm is much smaller than the computation time of the GREEDY EDA algorithm, in which we allow any edge perturbation (i.e., when  $C_p(G) = E$ ). (We do not consider the GREEDY EDA algorithm in this section.)

The RESTRICTED GREEDY EDA performs slightly better than the RESTRICTED GREEDY EA algorithm (which has similar performance to GREEDY EA algorithm) and performs much better than the GREEDY ED algorithm (see Figure 4.12). For a given  $\mu$ , we estimate a lower bound to  $Q$  using the total mean queue size  $Q_{\text{opt}}$  of  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  (depicted in Figure 4.10), the conjectured optimal undirected network for large values of  $\mu$ . The chosen value of  $\mu = 3\lambda_{\text{max}}$ , where  $\lambda_{\text{max}}$  is the maximum arrival rate of the initial graph of each simulation, is generally much larger (by a factor of at least 6.5–42, depending on the graph model) than the arrival rates of the optimized networks. Furthermore, the results in Sections 4.4.4–4.4.6 are qualitatively similar to the results when applying the greedy algorithms to reduce  $\lambda_{\text{total}}$  (see Appendix B.6.1). Therefore, we are likely in the large- $\mu$  regime (defined in Section 4.3.2). To see how close the obtained values of  $Q$  from our algorithms are to  $Q_{\text{opt}}$ , we calculate the ratio  $R_Q = Q/Q_{\text{opt}}$  for each simulation. We report the mean value of  $R_Q$  in Table 4.2 for each of the algorithms and for each of the graph models. The RESTRICTED GREEDY EDA algorithm achieves a mean value of  $R_Q$  that is close to 1, which suggests that the algorithm is able to perturb the networks to networks with total mean queue sizes close to (or even equal to)  $Q_{\text{opt}}$ . The mean values of  $R_Q$  for the RESTRICTED GREEDY EA algorithm are around 2–4, which suggests that the RESTRICTED GREEDY EA algorithm finds networks with about two to four times the optimal total mean queue size. The GREEDY ED algorithm yields much larger mean values of  $R_Q$ ; they are between 4.1 (for RRGs) to 47.5 (for RGGs).

---

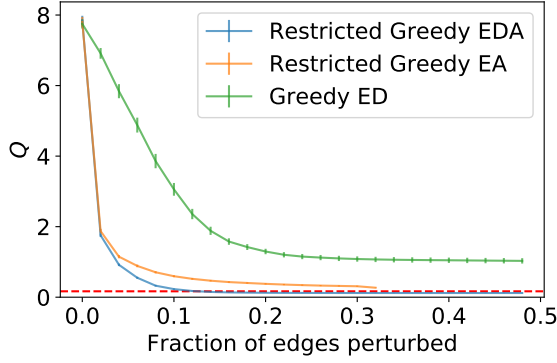
<sup>15</sup>EDA stands for ‘edge deletion and addition’.



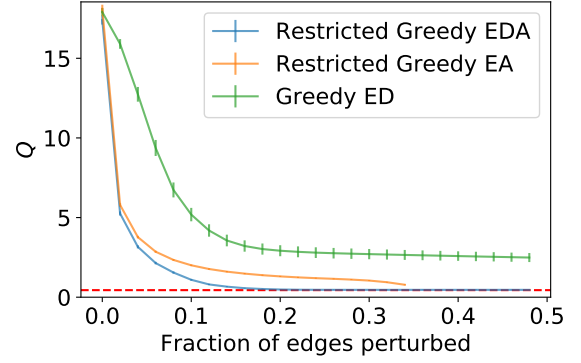
In summary, our results from Sections 4.4.4–4.4.6 suggest that edge additions are in general more effective than edge deletions in decreasing  $Q$ , but we decrease  $Q$  the most by allowing both types of edge perturbations.

*Table 4.2. Mean value of  $R_Q$  for the RESTRICTED GREEDY EDA, RESTRICTED GREEDY EA, and GREEDY ED algorithms for the six different random-graph models. We generate 100 random graphs for each random-graph model and run the three algorithm on them.*

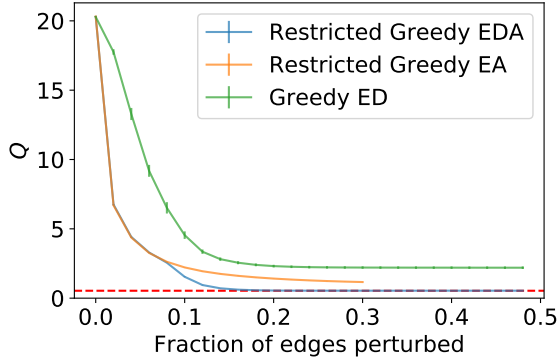
Model	RESTRICTED GREEDY EDA	RESTRICTED GREEDY EA	GREEDY ED
BA	1.003	2.736	9.352
ER	1.012	3.190	7.263
WS	1.000	2.435	4.707
RRG	1.000	2.167	4.117
RGG	1.032	3.808	47.512
Chung–Lu	1.036	3.578	11.471



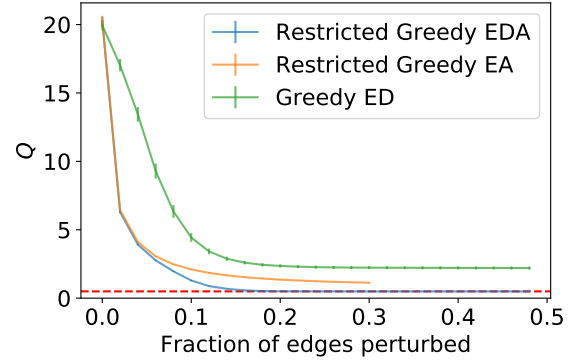
(a) Barabási–Albert network



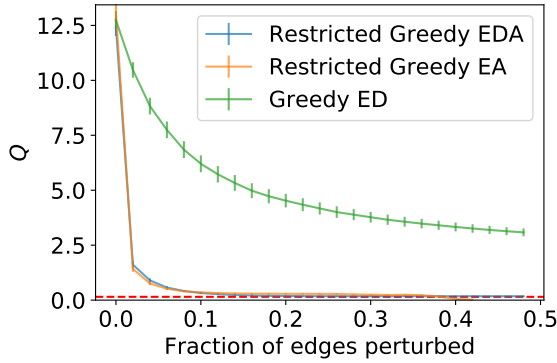
(b) Erdős–Rényi graph



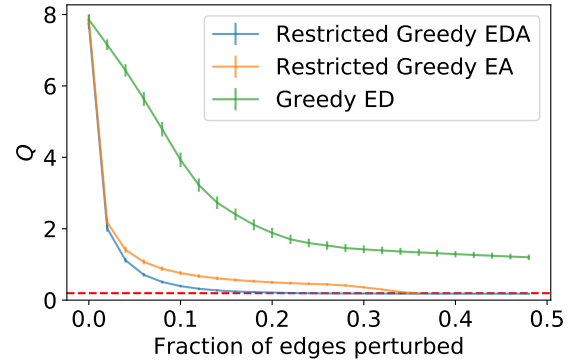
(c) Random regular graph



(d) Watts–Strogatz graph



(e) Random geometric graph



(f) Chung–Lu model

Figure 4.15. Comparison of the performance of the RESTRICTED GREEDY EDA, RESTRICTED GREEDY EA, and GREEDY ED algorithms in decreasing the total mean queue size  $Q$  with edge perturbations. We plot the mean and standard error of  $Q$  as a function of fraction of perturbed edges (i.e., the number of edge perturbations divided by the number of edges in the original graph). The red dashed line shows the mean value of  $Q_{\text{opt}}$  (averaged over the 100 simulations), the conjectured lower bound to  $Q$ .

#### 4.4.7 Applying greedy algorithms to store networks

We now consider applying the different edge-perturbation algorithms to a store network  $G$  (see Figure 4.16) of Store F with  $n = 179$  nodes and  $m = 384$  edges. What do the different edge perturbations correspond to in the store-network setting? As described in Chapter 3, zones are connected by an edge if they are contiguous. Deleting an edge  $\{i, j\}$  corresponds to blocking the direct walkway between  $i$  and  $j$ , such as by adding an extra shelf. Adding an edge  $\{i, j\}$  corresponds to creating a direct walkway between  $i$  and  $j$ . This is possible<sup>16</sup> only if zones  $i$  and  $j$  are next to each other but separated by shelves. In this case, we can add an edge  $\{i, j\}$  by opening a new walkway through a shelf that separates  $i$  and  $j$ . Automatically identifying which edges can be added is a difficult task and may not be possible without manual oversight. In this subsection, our procedure for identifying possible edge additions is as follows. We represent each edge  $\{i, j\}$  as a straight line between the centroids of zones  $i$  and  $j$ . We allow an edge addition  $\{i, j\}$  if the edge does not intersect with any other edges. We call this the *planarity constraint*, as it ensures that new edges do not violate the planarity of the store network.<sup>17</sup> This is only an approximation, as it may not be possible in practice to add every edge that we identify in this way to a store network. One has to check manually whether any perturbed store networks are realisable in practice.

We apply the various greedy algorithms with planarity constraint to the store network  $G$  from Store F. The simulation parameters are  $K = \lceil m/50 \rceil = 8$  and  $T = \lceil m/5 \rceil = 77$ . The service rate is  $3\lambda_{\max} \approx 6.68$ , where  $\lambda_{\max}$  is the maximum arrival rate of the original network.

First, we apply the GREEDY ED algorithm to the store network  $G$ . This algorithm is able to decrease  $Q$  significantly with a final value of  $Q \approx 1.02$  (see Figure 4.17a). In the final network (see the left part of Figure 4.17a), random walkers in the network are effectively ‘directed’ towards the sink node, as we have deleted edges that lead them further away from the sink.

Second, we apply the GREEDY EA algorithm (with planarity constraint) to  $G$ . (We do not apply the RESTRICTED GREEDY EA algorithm with planarity constraint, as the number of possible edges that it can add is small; there are only a small number of edge additions in which the added edge is incident to the sink node and does not cross existing edges.) The algorithm is able to decrease  $Q$  to final value of  $Q \approx 6.30$ . The minimum value of  $Q$  over the course of the algorithm is  $Q \approx 6.16$  after

<sup>16</sup>We assume that we cannot build bridges over shelves in a store.

<sup>17</sup>Although an original store network is not planar, it is approximately planar, as we only need to delete a small number of edges to obtain a planar graph.

8 edge additions. In contrast to the six random-graph models from Section 2.1.3, the decrease in  $Q$  is much smaller in the edge-addition algorithm than in the GREEDY ED algorithm. This is primarily because we are using the more restrictive GREEDY EA algorithm with planarity constraint on the store networks. This limits the possible edges that the algorithm can add. Without the planarity constraint, the decrease in  $Q$  is much larger; in this case, it has a final value of  $Q \approx 1.25$ . However, this value is still larger than the total mean queue size  $Q$  from applying the GREEDY ED algorithm.

Lastly, we apply the RESTRICTED GREEDY EDA algorithm (with planarity constraint) to  $G$ . The algorithm yields the largest decrease in  $Q$ ; we obtain a final value of  $Q \approx 0.565$  (see Figure 4.17c). This value of  $Q$  is close to the total mean queue size  $Q_{\text{opt}} \approx 0.559$  of the conjectured optimal network  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  for large  $\mu$ . When applying the RESTRICTED GREEDY EDA (with planarity constraint) to the store network, most edge perturbations are edge deletion; only two edge perturbations are edge additions. The perturbed network effectively directs random walkers to the sink node (i.e., the tills) and thus reduces congestion in supermarkets. Our result, however, is not very useful in practice. In such a network, customers are directed to the tills and will not buy much in the supermarket, as they do not traverse many other zones.

The random-walker model is not an accurate representation of a customer, as it does not capture any shopping behaviour. Reducing congestion in queueing networks, where customers are modelled as random walkers, lead to store networks that reduce congestion, but they prevent customers from shopping. Random walks, which do not capture the shopping behaviour, are therefore not a good model — at least on their own — for modelling customer movement in supermarket. We present a mobility model that better captures shopping behaviour in Chapter 5.

10 m

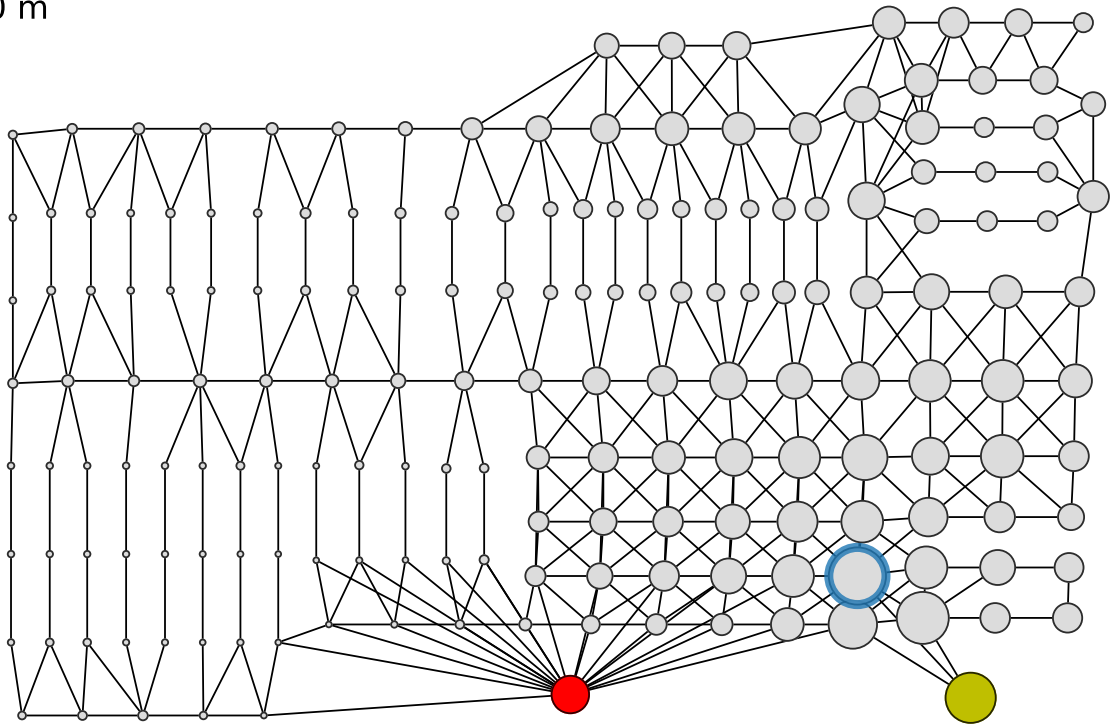
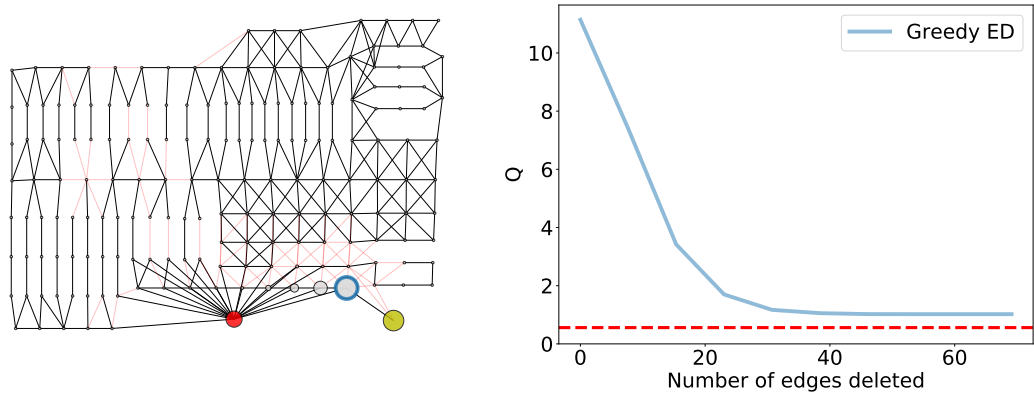
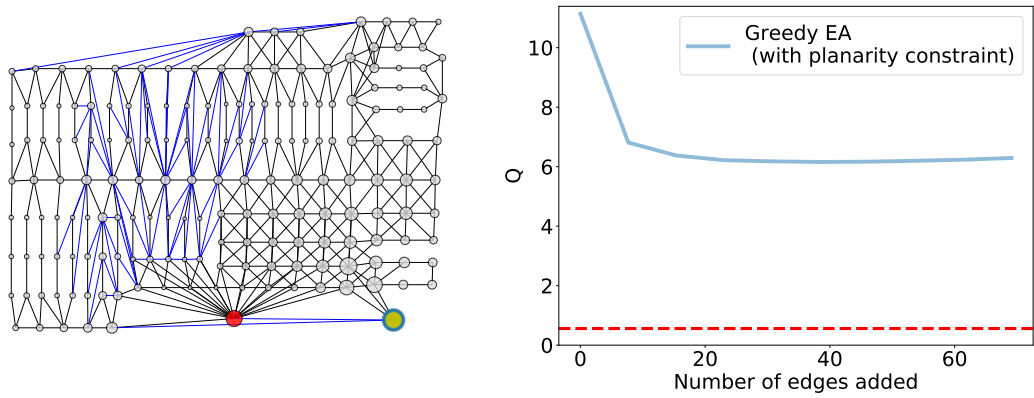


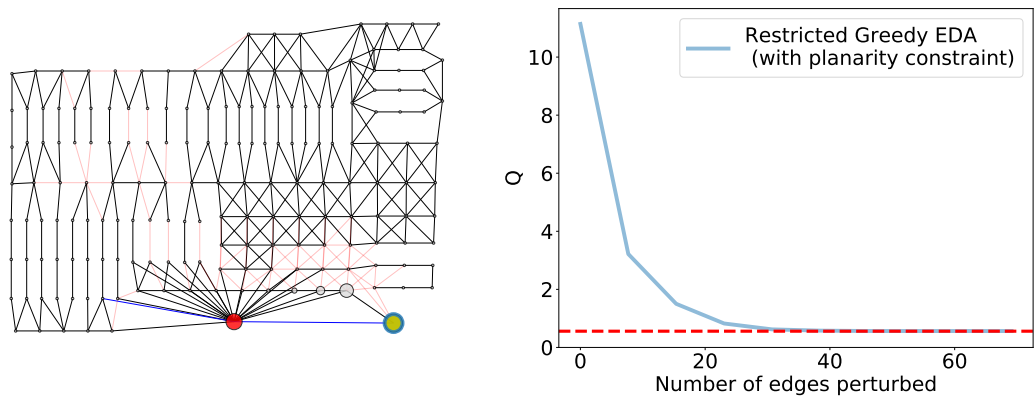
Figure 4.16. The store network of Store F with  $n = 179$  nodes and  $m = 384$  edges. We colour the source node (i.e., entrance) in yellow and the sink node (i.e., tills) in red. The node size is proportional to the arrival rate in the corresponding queueing network. We circle the node with the highest arrival rate in blue.



(a) Edge deletion



(b) Edge addition



(c) Edge deletion and addition

Figure 4.17. Applying the three greedy algorithms to the store network of Store F. The left part of each panel shows the final network after applying the indicated greedy algorithm. We colour the source node (i.e., entrance) in yellow and the sink node (i.e., tills) in red. We colour deleted edges in red and added edges in blue. The node size of each node is proportional to its arrival rate. The node with the highest arrival rate is circled in blue. The right part of each panel shows how  $Q$  changes as a function of the number of perturbed edges. The red dashed line shows the conjectured theoretical minimum value of  $Q$ .

## 4.5 Summary and Discussion

Inspired by modelling customer mobility and congestion in supermarkets, we introduced a model of a single-sink, single-source queueing network with unbiased random walkers. We used the total mean queue size  $Q$  as our congestion measure, as (by Little’s Law) minimizing  $Q$  is equivalent to minimizing the mean journey time of customers.

We first examined which network topologies minimize  $Q$  among all permissible networks (i.e., those that satisfy the reachability conditions of queueing networks) for any value of our model parameter  $\mu$  (which represents a homogeneous service rate). For any network with  $n \geq 3$  nodes, we found that the network  $G_{\bar{\mathcal{C}}_n}$  (depicted in Figure 4.6) minimizes  $Q$  among  $\bar{\mathcal{C}}_n$ , the set of all permissible networks with  $n$  nodes that do not have a directed edge from the source to the sink. We proved analytically the optimality (which we call  $Q$ -optimality) of this network for any value of  $\mu$ . When allowing an edge from the source to the sink, we identified a network  $G_{\mathcal{C}_n}$  that we conjecture to minimize  $Q$  among  $\mathcal{C}_n$  (the set of permissible networks with  $n$  nodes). We gave numerical evidence in support of this conjecture. We also examined queueing networks with undirected network topologies. We identified the network that is  $Q$ -optimal over  $\mathcal{U}_n$  (the set of undirected networks with  $n$  nodes that satisfy the reachability conditions) with  $n = 3$  and  $n = 4$  nodes. For  $n \geq 5$ , our results suggest that there is no single network that minimizes  $Q$ ; instead, an optimal network depends on the values of  $\mu$ . We proved that a network minimizes  $Q$  in the small- $\mu$  regime (i.e., for sufficiently small values of  $\mu$ ) also minimizes  $\lambda_{\max}$ . We further proved that a network minimizes  $Q$  in the large- $\mu$  regime (i.e., for sufficiently large values of  $\mu$ ) if and only if it minimizes  $\lambda_{\text{total}}$ . It follows that a  $Q$ -optimal network must minimize both  $\lambda_{\max}$  and  $\lambda_{\text{total}}$ . We found the unique networks that minimize  $\lambda_{\text{total}}$  over  $\mathcal{U}_n$  for  $n = 5, 6, 7$ . These networks do not minimize  $\lambda_{\max}$ , so there cannot exist a  $Q$ -optimal network for  $n = 5, 6, 7$ . For  $n \geq 8$ , we identified the network  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$ , that we conjecture to uniquely minimize  $\lambda_{\text{total}}$ ; and we found numerical evidence in support of this conjecture.  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  does not minimize  $\lambda_{\max}$ , so if the conjecture is true, no  $Q$ -optimal networks over  $\mathcal{U}_n$  exist for  $n \geq 5$ .

The second question that we explored in this chapter is how to perform edge additions or edge deletions or both (collectively called edge perturbations) on an existing network to decrease  $Q$ . We introduced an efficient greedy algorithm, which at each step performs the edge perturbation that decreases  $Q$  the most.<sup>18</sup> We only

---

<sup>18</sup>If there is a tie, we choose the edge perturbation randomly chosen among those that decrease  $Q$  the most.

considered edge perturbations on undirected networks, as supermarket networks are undirected, but one can adapt our efficient greedy algorithms in a straightforward way for directed networks. We applied our edge-perturbation algorithms to graphs from six random-graph models. We chose model parameters such that the graphs are sparse. We explored three scenarios: (1) allowing edge deletions only, (2) allowing edge additions only, and (3) allowing both edge deletions and edge additions.

For the first scenario, we presented the GREEDY ED algorithm and compared its performance with two simple algorithms (highest-degree-first (HDF) and highest-arrival-rates-first (HARF)) that have been used for reducing congestion in a related traffic-dynamics model. We found that the GREEDY ED algorithm consistently outperforms the other two algorithm on networks from the six random-graph models.

For the second scenario (edge additions only), we presented the GREEDY EA algorithm, which achieves a significantly larger reduction in  $Q$  than the GREEDY ED algorithm for all six random-graph models. This is not surprising, because we consider sparse networks, in which there are significantly more non-edges than edges. Therefore, there are significantly more edge perturbations that an edge-addition algorithm can apply compared to an edge-deletion algorithm. We also compared its performance to the performances of two simpler algorithms: the RESTRICTED GREEDY EA algorithm and the EDGE-TO-SINK-FIRST (ETSF) algorithm. The RESTRICTED GREEDY EA algorithm is a variant of the GREEDY EA algorithm. In this variant, we only allow adding edges that are incident to the sink node. The ETSF algorithm is an algorithm in which we add the edge between the sink node and the node with the highest arrival rate that is not already incident to the sink node. We found that the performances of all three edge-addition algorithms are similar, both in terms of their decrease in  $Q$  and with respect to which edges they add. Therefore, to decrease  $Q$  with edge additions, our results suggest that we should connect nodes with large arrival rates to the sink node.

For the third scenario (i.e., allowing edge additions and edge deletions), we presented the RESTRICTED GREEDY EDA algorithm. This algorithm is a combination of the GREEDY ED algorithm and the RESTRICTED GREEDY EA algorithm, as it is allowed (1) to add only edges that are incident to the sink node (similar to the RESTRICTED GREEDY EA algorithm) and (2) to delete any edge (similar to the GREEDY ED algorithm). The RESTRICTED GREEDY EDA algorithm achieves a slightly larger decrease in  $Q$  compared to the RESTRICTED GREEDY EA algorithm. Moreover, we found that the RESTRICTED GREEDY EDA algorithm yields networks with  $Q$  close to the conjectured minimum value  $Q_{\text{opt}}$  in the large- $\mu$  regime given by the total mean



queue size of  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$ . In other words, the RESTRICTED GREEDY EDA algorithm is able to achieve close to the conjectured maximum reduction in  $Q$ .

Finally, we applied edge-perturbation algorithms to a store network (i.e., a network that we constructed from a real supermarket). To ensure that our perturbations yield store networks that are realistic (or at least plausible) in practice, we allowed only additions of edges that do not intersect with existing edges. This constraint imposes planarity (assuming the original store network is planar) and therefore we call this constraint the planarity constraint. We found that the GREEDY ED algorithm, GREEDY EA algorithm with planarity constraint, and the RESTRICTED GREEDY EDA algorithm with planarity constraint are all able to decrease  $Q$ , with the largest decrease from the RESTRICTED GREEDY EDA algorithm with planarity constraint, followed by the GREEDY ED algorithm, and then by the GREEDY EA algorithm with planarity constraint. The network that we obtained after applying the RESTRICTED GREEDY EDA algorithm with planarity constraint has a total mean queue size  $Q$  that is close to our conjectured minimum. However, this network is not a practical one for supermarkets, as customers are directed towards the exit and do not shop. This illustrates the limitation of our random-walk model; it fails to capture the shopping behaviour of customers.

In summary, our work in this chapter presents an important step towards understanding how network topology affects the total mean queue size  $Q$  in supermarkets, modelled by open queueing networks with a single source and a single sink and with unbiased random walkers. We explored the optimal network topologies and several greedy algorithms for perturbing an existing graph towards an optimal network topology. However, our work also reveals that a random-walker model is too simplistic as a model for customer mobility behaviour. It is necessary to develop more accurate models, and we present one in Chapter 5. Furthermore, we have thus far considered only the special case of queueing networks with a single source and a single sink and with homogeneous service rates. There are extensions to our model that would make it either closer to real scenarios in a supermarket or other applications. We list possible extensions below.

**Optimal network topologies with heterogeneous service rates.** In Section 4.3, we considered open queueing networks in which the service rates are homogeneous. In real networks, however, the service rate is likely to be heterogeneous. For example, in a supermarket, the service rate may depend on the size and dimensions of a zone. Long and thin zones may have lower service rates, as it takes longer for a customer to

traverse such a zone. If we have information about the mean customer dwell time  $W_k$  at each node  $k$ , we can infer the empirical service rate  $\mu_k$  of each node  $k$  using Little's Law (see Section 2.3.2), which states that the mean queue size  $E[X_k]$  is equal to the mean dwell time  $W_k$  multiplied by the rate of arrivals  $\lambda_k$ :

$$W_k \lambda_k = E[X_k], \quad (4.61)$$

where

$$E[X_k] = \frac{\lambda_k}{\mu_k - \lambda_k} \quad (4.62)$$

is the mean queue size at  $k$  (see Equation (2.46)), which contains a single-server queue. Combining Equations (4.61) and (4.62), we obtain a formula for the empirical service rate:

$$\mu_k = \frac{1}{w_k} + \lambda_k. \quad (4.63)$$

Given empirical service rates  $(\mu_1, \dots, \mu_n)$  estimated from a store, it would be interesting to identify optimal network topologies in this case. We expect different optimal network topologies for different empirical service rates.

**Biased random walk.** A common extension to unbiased random walks is a degree-biased random walk [24, 98], in which a random walk visits a neighbour with probability proportional to the (out-)degree of the node. The transition matrix  $\mathbf{P}$  has entries

$$P_{ij} = \begin{cases} \frac{d_j A_{ji}}{\sum_{k \neq i} d_k A_{ki}}, & \text{if } \sum_k d_k A_{ki} > 0, \\ 0, & \text{otherwise.} \end{cases} \quad (4.64)$$

We expect it to be an interesting challenge to extend our results to degree-biased random walks.

We can also consider a different extension of random walks, in which random walkers tend to avoid more congested nodes. We call such a random walk a *congestion-biased* random walk. For example, we can consider the transition matrix  $\mathbf{P}$  with the following entries

$$P_{ij} = \begin{cases} \frac{q_j A_{ji}}{\sum_{k \neq i} q_k A_{ki}}, & \text{if } \sum_{k \neq i} q_k A_{ki} > 0, \\ 0, & \text{otherwise,} \end{cases} \quad (4.65)$$

where  $q_k = \lambda_k / (\mu_k - \lambda_k)$  is the mean queue size and  $\lambda_k$  is the arrival rate calculated from the traffic equations:

$$\lambda_i = \delta_{1i} \lambda + \sum_{j=1}^{n-1} \lambda_j P_{ji}, \quad \text{for } i = 1, \dots, n. \quad (4.66)$$

In this model, the dynamics is coupled with the congestion. Equations (4.65) and (4.66) are a coupled set of equations and we do not know whether solutions exist or whether a solution is unique (if one exists).

More work is required to understand queueing networks with such biased random walks.

## Chapter 5

# Estimating mobility flow in supermarkets

In this chapter, we consider the problem of modelling mobility flow between zones inside supermarkets and investigate how the flow changes when we rearrange store layouts. Understanding how customer mobility is influenced by a store’s layout is of great interest to retailers, as it can help suggest better store layouts with low congestion or increased foot traffic (and hence sales) in desired areas. Our approach is to use population-level mobility models — specifically, gravity and intervening-opportunities (IO) models — to model customer mobility flow in supermarkets. To the best of our knowledge, these models have been used only on large spatial scales, ranging from inter-country level on a scale of thousands of kilometres (e.g., estimating trade flow [4, 11]) to city and region levels at the scale of tens of kilometres (e.g., estimating commuting patterns [70]). In our work, by contrast, we consider building-level scales, where the prevalent approach is to use pedestrian models, such as mobility models for individuals (e.g., random walks [36]) or models for crowd dynamics [37]. Because we consider aggregate flow, population-level mobility models are more suitable than random walks<sup>1</sup> or crowd-dynamic models for our problem (despite the small spatial scales of these systems). The small, building-level spatial scale may affect the fundamental features of mobility dynamics (and therefore the performance of the models) in important ways. For example, it has been reported that some models (such as the radiation model) perform worse on small spatial scales [70]. A possible reason for this observation is that the spatial ‘force’ is much smaller on small scales than on large ones, due to the smaller cost of making a trip in the former situation, so other non-spatial ‘forces’ that are not captured in these models may instead be important

---

<sup>1</sup>However, as we show in Section 5.3, we can interpret population-level mobility models as flows that arise from random walks.

factors that underlie the flow. In a supermarket, for example, the distance between two zones (a spatial ‘force’) may be less relevant than the number of their complementary items (a non-spatial ‘force’) for the flow between the two zones.<sup>2</sup> Furthermore, these models are inherently memoryless, as they describe the mobility flow from an origin location to a destination location using local attributes of the origin and destination locations, without considering the location from which (or how) a person who leaves the origin location entered it in the first place. When one models humans walking in a building (e.g., in a supermarket or a museum), the direction from which a person came likely influences where that person goes next, so there is memory in the system. For example, Farley and Ring [27] observed that customers in supermarkets tend to move from the entrance unidirectionally along the outer perimeter after entering a store.

The main contribution of this chapter is to show that several different mobility models can successfully estimate the majority of observed trips in supermarket customer-flow data (which we estimate from anonymized, ordered customer-basket data), demonstrating that these models can work on small (specifically, building-level) spatial scales, despite the aforementioned drawbacks.

The chapter proceeds as follows. In Section 5.1, we describe our mathematical set-up. In Section 5.2, we describe the data set from which we infer the origin–destination (OD) trips in 17 supermarkets. In Section 5.3, we describe the mobility models and goodness-of-fit measures that we use in our investigation. We also describe how we estimate the parameters of our models. In Section 5.4, we present our results when applying these models to supermarket store data using both (in-sample) fitting and (out-of-sample) estimation of customer flows. We also examine how the size of the data set and the model parameter values affect the performance of our models. In Section 5.5, we describe a framework for extending our models by incorporating zone–zone correlations and we present the results when fitting these models to the data. Finally, we summarize our results and discuss potential applications in Section 5.6.

## 5.1 Mathematical set-up

In this section, we set up our approach for analysing mobility flow in supermarkets. We discuss how we discretize space in a supermarket, how we model shopping journeys, and how we characterize the flow between zones of a supermarket.

---

<sup>2</sup>We propose a way to integrate such a non-spatial force in our models in Section 5.5.

In our investigation, we employ mobility models that require us to discretize space (i.e., a supermarket). As in Chapter 4, we represent each supermarket by its corresponding store network  $G$ . (See Chapter 3 for how we create each store network.) Recall that the store network  $G$  is a spatial, undirected network with  $n$  nodes (representing rectangular zones) and  $m$  edges, which connect neighbouring zones. Although we have distances between supermarket zones,  $G$  itself is unweighted. For each edge  $(i, j)$ , we assign an edge length  $l_{ij}$  which we take to be the Euclidean distance between its two incident nodes  $i$  and  $j$ . (The edge length approximates the walking distance between two adjacent nodes.) We define an  $n \times n$  distance matrix  $\mathbf{\Lambda}$  associated with  $G$  with entries  $d_{ij}$ . The quantity  $d_{ij}$  is equal to the shortest-path distance between  $i$  and  $j$ ; this distance is the minimum length of a path between  $i$  and  $j$ .

One customer's *shopping journey* is a sequence of  $K + 2$  zones  $(s_0, \dots, s_{K+1})$ , where  $K$  is the number of items that the customer buys,  $s_0 = 1$  (entrance),  $s_{K+1} = n$  (tills), and  $s_1, \dots, s_K$  are the zones at which a customer picks up items, which we order by their pick-up times. A customer can purchase multiple items in the same zone, so  $s_0, \dots, s_{K+1}$  may not be distinct. Each consecutive pair  $(s_k, s_{k+1})$  of distinct zones (so  $s_k \neq s_{k+1}$ ) for  $0 \leq k \leq K$  constitutes an origin–destination (OD) trip (or simply a trip). That is, a trip is a segment of a customer's shopping journey that is either between consecutive purchases in different zones, from the entrance to the first purchase, or from the last purchase to the tills. We are interested in the number  $T_{ij}$  of OD trips in a store from each origin zone  $i$  to each destination zone  $j$  (over some duration  $\tau$ ). We do not consider flow within a zone and thus set  $T_{kk} = 0$  for  $k = 1, \dots, n$ . The  $n \times n$  matrix  $\mathbf{T}$ , with entries  $T_{ij}$ , is an origin–destination (OD) matrix (see Section 2.4); its off-diagonal entries record the mobility flow between zones. We denote an empirical OD matrix by  $\mathbf{T}^{\text{data}}$  and an OD matrix from a model by  $\mathbf{T}^{\text{model}}$ . Throughout this chapter, we denote an origin node of a trip by  $i$  and a destination node of a trip by  $j$ ; we index other nodes using the symbol  $k$ .

## 5.2 Data

We use anonymized, ordered customer-basket data from 17 Tesco supermarket stores (Stores A–Q) over a common three-month period (91 days). The data consists of a fraction  $\theta \approx 0.07$  of all customer baskets in these stores. We summarize the properties of the data in Table 5.1.

For each store, we infer the number  $T_{ij}^{\text{data}}$  of OD trips from zone  $i$  to zone  $j$  over the  $\tau = 91$  days from the data as follows. Each ordered customer basket is a list of item purchases, which we order by pick-up time. We use item-location data to map each ordered list of purchases to their associated zones in a supermarket. For example, we map a list of purchases (e.g., bread, milk, butter, and pasta) to its corresponding shopping journey (1, 10, 16, 28, 26, 53), where bread is in zone 10, milk is in zone 16, butter is in zone 28, pasta is in zone 26, and the tills are in zone 53 (see Figure 5.1). In this example, each item has a unique item location, so we can recover the corresponding shopping journey in a straightforward way. However, about 10% of the purchased items have unknown item locations and about 8% of the purchased items have multiple item locations; we refer to the latter items as *multi-located items*. We remove items with unknown item locations from customer baskets. For each basket with one or more multi-located items, we consider all combinations of possible purchase locations for those items. (For example, there are  $2^r$  combinations for a basket with  $r$  multi-located items with 2 locations each.) For each combination, we calculate the sum of the shortest-path distances between the locations of consecutive purchases in the basket. We then choose a combination of the purchase locations that minimizes this sum. The above procedure of mapping a basket to a shopping journey was done by Tesco. The data that we received therefore consists of sequences of zones (i.e., the shopping journeys).

We decompose each customer shopping journey into a sequence of OD trips and estimate the total number  $T_{ij}^{\text{data}}$  of trips from zone  $i$  to zone  $j$  by counting all OD trips  $(i, j)$  from the data. For example, the previous example shopping journey (1, 10, 16, 28, 26, 53) contains 5 trips: (1, 10), (10, 16), (16, 28), (28, 26), and (26, 53). Assuming that the observed mobility patterns in our data set are representative for the mobility of all customers, we rescale  $T_{ij}^{\text{data}}$  by multiplying it by  $1/\theta$  to estimate the mobility flow of all customers that visit a store.

Table 5.1. Summary of the data set, which comes from 17 Tesco stores. For each quantity, we give the minimum, mean, and maximum values across the 17 stores.

	Min	Mean	Max
Number of zones ( $n$ )	61	123	197
Number of edges ( $m$ )	128	236	401
Number of baskets	2479	13672	29201

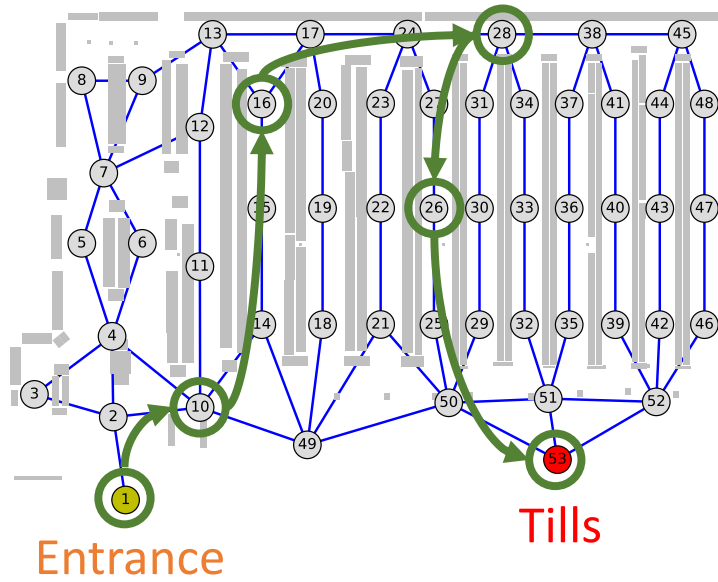


Figure 5.1. Example of a customer journey in a supermarket. The customer purchases an item (e.g., bread) in zone 10, another item (e.g., milk) in zone 16, a third item (e.g., butter) in zone 28, and a fourth item (e.g., pasta) in zone 26. We map this journey to the shopping journey (1, 10, 16, 28, 26, 53) and then divide it into 5 origin–destination (OD) trips: (1, 10), (10, 16), (16, 28), (28, 26), and (26, 53). Each green arrow represents an OD trip.



### 5.3 Mobility models

We use several mobility models (including those described in Section 2.4) to estimate  $\mathbf{T}^{\text{data}}$ . Let  $O_k^{\text{data}} = \sum_j T_{kj}^{\text{data}}$  and  $D_k^{\text{data}} = \sum_i T_{ik}^{\text{data}}$ , respectively, be the empirical numbers of trips that depart from and arrive at each node  $k$ . Note that  $O_k^{\text{data}}$  and  $D_k^{\text{data}}$  are also the row and column sums, respectively, of  $\mathbf{T}^{\text{data}}$ . Each of the mobility models yields an  $n \times n$  OD matrix  $\mathbf{T}^{\text{model}}$  from  $O_k^{\text{data}}$ ,  $D_k^{\text{data}}$ , the store network, and either one or zero fitting parameters. As discussed in Section 2.4, we seek for the models to yield OD matrices  $\mathbf{T}^{\text{model}}$  that are close to  $\mathbf{T}^{\text{data}}$ . In Section 5.3.4, we discuss the goodness-of-fit measures that we use in this chapter. The models use  $2n$  pieces of information of  $\mathbf{T}^{\text{data}}$  to estimate the  $(n-1) \times (n-1)$  off-diagonal entries of  $\mathbf{T}^{\text{data}}$ .

In our problem,  $O_k^{\text{data}} = D_k^{\text{data}}$  for all nodes  $k$  except for  $k = 1$  (entrance) and  $k = n$  (tills), as every customer who finishes a trip in zone  $k$  (except for  $k = 1, n$ ) continues their journey with a trip that starts from  $k$ . The quantity  $O_k^{\text{data}}$  also corresponds to the number of *shopping visits* at  $k$  (except for  $k = 1, n$ ), which are the number of times that customers visit  $k$  to purchase one or more items. Furthermore, the number  $C_s$  of journeys in the data satisfies  $C_s = O_1^{\text{data}} - D_1^{\text{data}}$  and  $C_s = D_n^{\text{data}} - O_n^{\text{data}}$ , as every journey starts at node 1 and ends at node  $n$ . (Note that  $D_1^{\text{data}}$  and  $O_n^{\text{data}}$  need not be equal to 0, as the entrance and till nodes can contain items, so customers can visit 1 or  $n$  to purchase something in the middle of their shopping journey.) We can therefore determine  $\{O_k^{\text{data}}\}_{k=1}^n$  and  $\{D_k^{\text{data}}\}_{k=1}^n$  from  $\{O_k^{\text{data}}\}_{k=1}^n$  and  $D_1^{\text{data}}$ . In practice, we estimate  $\{O_k^{\text{data}}\}_{k=1}^n$  and  $D_1^{\text{data}}$  from purchase data (see Appendix C.2). Therefore, we assume that we know  $\{O_k^{\text{data}}\}_{k=1}^n$  and  $\{D_k^{\text{data}}\}_{k=1}^n$ .

We use doubly-constrained versions of the mobility models (see Section 2.4.3), as we know  $\{O_k^{\text{data}}\}_{k=1}^n$  and  $\{D_k^{\text{data}}\}_{k=1}^n$ . As discussed in Section 2.4.3, the mobility flow  $\mathbf{T}^{\text{model}}$  in doubly-constrained models satisfies

$$O_k^{\text{data}} = O_k^{\text{model}}, \quad (5.1)$$

$$D_k^{\text{data}} = D_k^{\text{model}}, \quad (5.2)$$

where  $O_k^{\text{model}} = \sum_j T_{kj}^{\text{model}}$  and  $D_k^{\text{model}} = \sum_i T_{ik}^{\text{model}}$ . In other words,  $\mathbf{T}^{\text{model}}$  has the same row sums and column sums as  $\mathbf{T}^{\text{data}}$ . This, in turn, implies for each node  $k$  that both the number of trips that arrive at  $k$  and the number of trips that depart from  $k$  are equal to their empirical values. Therefore, for notational simplicity, we drop the superscripts on  $O_k$  and  $D_k$  for the remainder of this chapter. Because  $O_k = D_k$  for

$k = 2, \dots, n - 1$ , the number of people at each node (except for the entrance and till nodes) is also conserved in the model.

The mobility flow in a doubly-constrained model is

$$T_{ij}^{\text{model}} = (A_i O_i) \times (B_j D_j) \times f_{ij}, \quad (5.3)$$

where  $f_{ij}$  is the model-specific attraction value of  $j$  to  $i$  and  $A_i, B_j \geq 0$  are scaling factors (also called “balancing factors”) to ensure that Equations (5.1) and (5.2) are satisfied. Given  $O_k, D_k$ , and  $f_{ij}$ , we determine  $A_i$  and  $B_j$  using the iterative proportional-fitting procedure that we described in Section 2.4.3. We summarize the attraction values  $f_{ij}$  in the attraction matrix  $\mathbf{f}$  with entries  $f_{ij}$  if  $i \neq j$  and 0 otherwise. In practice, we found that the iterative proportional-fitting procedure does not always converge when  $\mathbf{f}$  contains a row of zeros or a column of zeros. We resolve this issue for our models by first calibrating the balancing factors  $A_i$  and  $B_j$  for an amended attraction matrix  $\tilde{\mathbf{f}}$ , which we obtain by setting all entries of all zero rows and columns of  $\mathbf{f}$  to the constant  $\langle \mathbf{f} \rangle$ , which is equal to the mean value of the entries of  $\mathbf{f}$ . In our models that we present in Sections 5.3.1–5.3.3, the  $k^{\text{th}}$  row is zero only if  $O_k = 0$  and the  $k^{\text{th}}$  column is a zero column only if<sup>3</sup>  $D_k = 0$ . Due to this fact, the values  $A_i$  and  $B_j$  calibrated on  $\tilde{\mathbf{f}}$  are also valid balancing factors for the original attraction matrix  $\mathbf{f}$ . That is,  $A_i$  and  $B_j$  satisfy the following equations:

$$O_i = \sum_j T_{ij}^{\text{model}} = \sum_j A_i O_i B_j D_j f_{ij}, \quad (5.4)$$

$$D_j = \sum_i T_{ij}^{\text{model}} = \sum_i A_i O_i B_j D_j f_{ij}. \quad (5.5)$$

(These equations correspond to Equations (2.73) and (2.74).) To see that  $A_i$  and  $B_j$  satisfy Equations (5.4) and (5.5), note that  $f_{ij} \neq \tilde{f}_{ij}$  only if the  $i^{\text{th}}$  row or  $j^{\text{th}}$  column of  $\mathbf{f}$  is a zero vector. Therefore, when  $f_{ij} \neq \tilde{f}_{ij}$ , either  $O_i = 0$  (if the  $i^{\text{th}}$  row is a zero vector) or  $D_j = 0$  (if the  $j^{\text{th}}$  column is a zero vector). Consequently,

$$O_i D_j f_{ij} = O_i D_j \tilde{f}_{ij}, \quad (5.6)$$

because both sides are equal to 0 when  $f_{ij} \neq \tilde{f}_{ij}$ . Therefore,

$$\sum_j A_i O_i B_j D_j f_{ij} = \sum_j A_i O_i B_j D_j \tilde{f}_{ij} = O_i, \quad (5.7)$$

$$\sum_i A_i O_i B_j D_j f_{ij} = \sum_i A_i O_i B_j D_j \tilde{f}_{ij} = D_j, \quad (5.8)$$

---

<sup>3</sup>Note that the converse is not true for all models. In other words, it is possible that  $O_k = 0$  and the  $k^{\text{th}}$  row is not equal to zero. It is also possible that  $D_k = 0$  and the  $k^{\text{th}}$  column is not equal to zero.

as required.

We interpret  $T_{ij}^{\text{model}}$  as the mean aggregate flow that arises from a continuous-time random-walk-type model at stationarity. Customers arrive at node 1 (i.e., the entrance) at a rate  $\lambda = C_s/\tau$ . In contrast to a standard random walk on networks that we used in Chapter 4, customers do not choose a random neighbour at each step. Instead, each customer at  $i$  chooses a random destination  $j$ , which need not be adjacent to  $i$ , with transition probability  $P_{ij} = A_i B_j D_j f_{ij}$ ; and it then takes a trip from  $i$  to the chosen destination  $j$ . That is, the customer traverses some path from  $i$  to the chosen destination  $j$ . (Each destination represents a zone where the customer purchases one or multiple items.) For simplicity, we assume that customers take a shortest path from node  $i$  to node  $j$ , where we choose this path uniformly at random from all shortest paths between these two nodes. (Other routing models are possible; one possibility is a standard random walk that starts at node  $i$  and reaches an absorbing state at node  $j$ .) We remove customers who finish a trip at node  $n$  at a constant rate  $\lambda$ ; this ensures that the mean number of customers in the system is constant. Therefore,  $T_{ij}^{\text{model}}$  gives the mean number of trips from zone  $i$  to zone  $j$  over a period  $\tau$ . We assume implicitly that there is no memory in customer mobility; the next destination of each customer depends only on the origin location (and not on previously visited locations).

We calculate the attraction values  $f_{ij}$  from  $d_{ij}$ ,  $O_i$ ,  $\{D_k\}_{k=1}^n$ , and information (such as the distance between two nodes) from a store network as inputs. We present each model in a *scale-invariant* form, such that the parameters are dimensionless and the transition probabilities  $P_{ij} = A_i B_j D_j f_{ij}$  are invariant under the scalings  $O_k \mapsto aO_k$  and  $D_k \mapsto aD_k$  for  $a > 0$  and for all  $k$ . Because  $O_k$  and  $D_k$  scale with the number  $C_s$  of journeys in the data set (and therefore with  $\tau$ ), scale invariance ensures that the model parameters and the transition probabilities  $P_{ij}$  are independent of  $C_s$  and  $\tau$ .<sup>4</sup>

In Table 5.2, we summarize the different choices of  $f_{ij}$  and the number of parameters for each of the models that we employ. We discuss these models in the following subsections.

---

<sup>4</sup>For sufficiently large  $\tau$ , it would be interesting to generalize our analysis to incorporate seasonal shopping variations, such as differences in behaviour during holidays.

Table 5.2. Summary of the five mobility models that we employ. The OD matrix of each model is given by Equation (5.3), with different functional forms for the attraction values  $f_{ij}$ . Note that  $N$  is the total number of trips.

Model	$f_{ij}$	Parameter	Parameter range	References
Gravity (power law)	$D_j (d_{ij}/l)^{-\gamma}$	$\gamma$	$[0, \infty)$	[16, 102, 113]
IO (Schneider's)	$\exp\left(-\frac{L}{N}S_{ij}\right) - \exp\left(-\frac{L}{N}(S_{ij} + D_j)\right)$	$L$	$[0, N]$	[87, 89]
Radiation	$\frac{O_i D_j}{(O_i + S_{ij})(O_i + D_j + S_{ij})}$	-	-	[70, 91, 92]
Extended radiation	$\frac{[(O_i + S_{ij} + D_j)^\alpha - (O_i + S_{ij})^\alpha][(O_i)^\alpha + N^\alpha]}{((O_i + S_{ij})^\alpha + N^\alpha)((O_i + S_{ij} + D_j)^\alpha + N^\alpha)}$	$\alpha$	$[0, \infty)$	[107]
Benchmark	1	-	-	-

### 5.3.1 Gravity models

We use the gravity model with power-law deterrence function, which has attraction values of

$$f_{ij} = f_{\text{GP}}(d_{ij}) = (d_{ij}/l)^{-\gamma} , \quad (5.9)$$

where  $l > 0$  is a spatial normalization factor (which we choose to be the mean zone length) and  $\gamma \geq 0$  is a dimensionless fitting parameter. In Appendix C.1, we also present the results for when we apply the gravity model with exponential deterrence function. In contrast to other studies on small spatial scales [59, 63, 92], we find that a power-law deterrence function gives a (slightly) better fit to our data than the exponential deterrence function. One may expect the exponential deterrence function to give the same performance as the power-law deterrence function in our problem, because the difference between the power-law function and the exponential function is large only when one considers trip distances that range across several orders of magnitude. In our problem, we consider trip distances that range from about 5 m to 100 m and thus only span at most two orders of magnitude. The superior performance of the power-law deterrence function of the gravity model (compared to the exponential deterrence function) is an interesting and unexplained feature of our problem.

### 5.3.2 Intervening-opportunities models

In the IO models, we define the opportunities at each node as follows. We use a more general notion of “opportunity” than in Section 2.4.2, as we now assume that each node  $k$  has two types of opportunities: origin opportunities and destination opportunities. We assume that the number of origin opportunities is equal to  $O_k$ , the number of trips that depart from  $k$ , and the number of destination opportunities is equal to  $D_k$ , the number of trips that arrive at  $k$ . We use the origin opportunities when we consider the flow originating from  $k$  and use the destination opportunities we consider flow arriving at  $k$ . However, as mentioned earlier in the section, the number of origin and destination opportunities coincide for all nodes except for the entrance and till nodes. We use two types of opportunities (instead of one) to ensure that the values  $f_{k1}$  and  $f_{nk}$  are small (or even zero) for all nodes  $k$ , so there are few (or zero) trips arriving at 1 and few trips departing from  $n$  in the IO models.

The intervening opportunities  $S_{ij}$  of an OD pair  $(i, j)$  consist of all destination opportunities  $D_k$  in nodes  $k$  that satisfy  $d_{ik} < d_{ij}$ . The number  $S_{ij}$  of intervening

opportunities of an OD pair  $(i, j)$  is then

$$S_{ij} = \sum_{\substack{k \neq i \\ d_{ik} < d_{ij}}} D_k. \quad (5.10)$$

Therefore, the destination opportunities in our problem amount to opportunities for customers to stop and purchase something. Note that  $S_{ij} \mapsto aS_{ij}$  when we scale  $D_k \mapsto aD_k$  for all  $k$ , so the number of opportunities scales linearly with the number  $C_s$  of shopping journeys.

We use three different IO models: Schneider's IO model, the radiation model, and the extended radiation model. We list the attraction values  $f_{ij}$  for each of these models in Table 5.2. The attraction values  $f_{ij}$  are the same as those that we presented in Section 2.4.2 except that  $m_i$  is replaced by  $O_i$ , the quantity  $m_j$  is replaced by  $D_j$ , and there are some small modifications to the formulas of Schneider's IO model and extended radiation model. The modifications for these two models ensure that their model parameters are dimensionless and that  $f_{ij}$  is scale-invariant. We highlight the modifications below.

The attraction values  $f_{ij}$  for Schneider's IO model are

$$f_{ij} = f_{\text{IO}}(D_j, S_{ij}) = e^{-\frac{L}{N}S_{ij}} - e^{-\frac{L}{N}(S_{ij}+D_j)} > 0, \quad (5.11)$$

where  $N = \sum_{i,j} T_{ij}^{\text{data}}$  is the total number of trips and  $L \in (0, N]$  is a dimensionless fitting parameter. This formulation differs from Equation (2.60), in that we use  $L/N$  in Equation (5.11) instead of  $L$ . By using  $L/N$ , we ensure that  $L$  is dimensionless. For the remainder of this chapter, we refer to Schneider's IO model by the IO model.

For the extended radiation model, the attraction values are

$$f_{ij} = f_{\text{ext}}(O_i, D_j, S_{ij}) = \frac{[(O_i + S_{ij} + D_j)^\alpha - (O_i + S_{ij})^\alpha] ((O_i)^\alpha + N^\alpha)}{((O_i + S_{ij})^\alpha + N^\alpha) [(O_i + S_{ij} + D_j)^\alpha + N^\alpha]}. \quad (5.12)$$

The difference to Equation (2.64) is that we replace the additive constant 1 in the second term of the product in both the numerator and the denominator by  $N^\alpha$ . This modification ensures that  $f_{ij} = f_{\text{ext}}(O_i, D_j, S_{ij})$  is scale-invariant.

### 5.3.3 Benchmark model

We compare the mobility models above to a simple benchmark model that ignores all spatial factors. In this benchmark model, the attraction values  $f_{ij}$  are all the same and equal to 1. That is,

$$f_{ij} = 1. \quad (5.13)$$

In such a benchmark model, the attraction value between two nodes does not depend on the layout of the store network; the mobility flow  $T_{ij}$  depends only on the values of  $\{O_k\}_{k=1}^n$  and  $\{D_k\}_{k=1}^n$ .

### 5.3.4 Goodness-of-fit measures

The main goodness-of-fit measure that we use is the common part of commuters (CPC) (see Equation (2.78)). Because our models are doubly-constrained, we can interpret the CPC score as the fraction of customers whose trip is assigned correctly by a model. We mainly use CPC, as the other measures from Section 2.4.4 often give similar results to CPC when comparing the performance of mobility models [59, 72]. Furthermore, CPC has an intuitive interpretation in our modelling context.

In addition to CPC, we also consider an application-specific goodness-of-fit measure  $\text{NRMSE}_v$ , which measures the normalized root-mean-square error (NRMSE) in the number of visits to each node. When we examine congestion in supermarkets in Chapter 6, we use measures of congestion that depend on the number of visits to each node, so a mobility model should have low values of  $\text{NRMSE}_v$  for it to be viable for our application to congestion. Furthermore, obtaining a good estimate for the number of visits to each node is useful for other applications. For example, a retailer may seek to estimate the foot traffic for each aisle to measure how many customers walk past a product without purchasing it to understand which items are unpopular due to low foot traffic. Given an OD matrix  $\mathbf{T}$ , we estimate the number of visits by assuming that each customer that takes an OD trip  $(i, j)$  traverses a shortest path from  $i$  to  $j$ , chosen uniformly at random among all shortest paths from  $i$  to  $j$  for each trip. Each customer who takes a trip from  $i$  to  $j$  visits each node along the chosen shortest path. The estimated number  $v_k$  of visits to each node  $k$  is the weighted sum of the number  $T_{ij}$  of trips with OD pairs  $(i, j)$  for all  $i$  and  $j$ , where the weight  $\omega_{ikj}$  is the fraction of shortest paths from  $i$  to  $j$  that traverse  $k$ . (We use the convention that the end nodes  $i$  and  $j$  are construed as traversed in a shortest path from  $i$  to  $j$ .) That is,

$$v_k = v_k(\mathbf{T}) = \sum_{i,j} \omega_{ikj} T_{ij}. \quad (5.14)$$

The number  $v_k$  of visits is closely related to geodesic node betweenness centrality [78], which we recover when  $T_{ij} = 1$  for all  $(i, j)$ . We can compute all  $v_k$  values in  $\mathcal{O}(nm)$  time using a straightforward adaptation of a fast algorithm for computing geodesic betweenness centrality [15].

To measure the model error in the estimated number of visits, we calculate the NRMSE in  $v_k(\mathbf{T})$  with the formula

$$\text{NRMSE}_v = \left( \frac{\sum_{k=1}^n (v_k(\mathbf{T}^{\text{data}}) - v_k(\mathbf{T}^{\text{model}}))^2}{n v_{\max}(\mathbf{T}^{\text{data}})^2} \right)^{\frac{1}{2}}, \quad (5.15)$$

where  $v_{\max}(\mathbf{T}) = \max_k [v_k(\mathbf{T})]$  is the number of visits to the most visited node.

### 5.3.5 Parameter calibration

Following the approach in [60], we calibrate the model parameters  $\gamma$ ,  $L$ , and  $\alpha$  of the gravity, IO, and extended radiation models (respectively) for each data set by maximizing the CPC score. We call a parameter value ‘optimal’ when it maximizes the CPC score for a given model and data set. We use the python function `minimize_scalar` from the `scipy.optimize` package (Version 1.14.3) [46] to find an optimal parameter value. The function `minimize_scalar` searches for an optimal parameter value over a finite interval  $I$  that we need to specify. We use  $I = [0, 5]$  for the gravity model,  $I = [0, 100]$  for the IO model, and  $I = [0, 10]$  for the extended radiation model. While the full parameter range for the three models is larger than  $I$  — in fact, it is unbounded for the gravity model and extended radiation model — we show in Section 5.4.3 that the CPC value decreases as the parameter value approaches the upper bound of the interval  $I$ . These results suggest that an optimal parameter value lies inside the interval  $I$ .

## 5.4 Results

### 5.4.1 Fit to data

We test the five models listed in Table 5.2 on each of the 17 stores. In our computations, the gravity model (with a power-law deterrence function) consistently achieves the best CPC score across the stores, with a mean of about 0.686 (see Figure 5.2a and Table 5.3). This value is comparable with reported CPC scores in previous studies on mobility systems with larger spatial scales [59, 72, 107]. (The highest reported CPC scores in these studies are between 0.67 and 0.72.) The performance of the extended radiation model, with a mean CPC score of 0.672, is almost as successful as the gravity model on average. The IO model consistently yields lower CPC scores than the gravity and extended radiation models. The gravity model also has the best (i.e., lowest) mean value of  $\text{NRMSE}_v$  across the 17 stores (see Table 5.3), closely followed by the IO model and then the extended radiation model. In terms of  $\text{NRMSE}_v$ , the relative



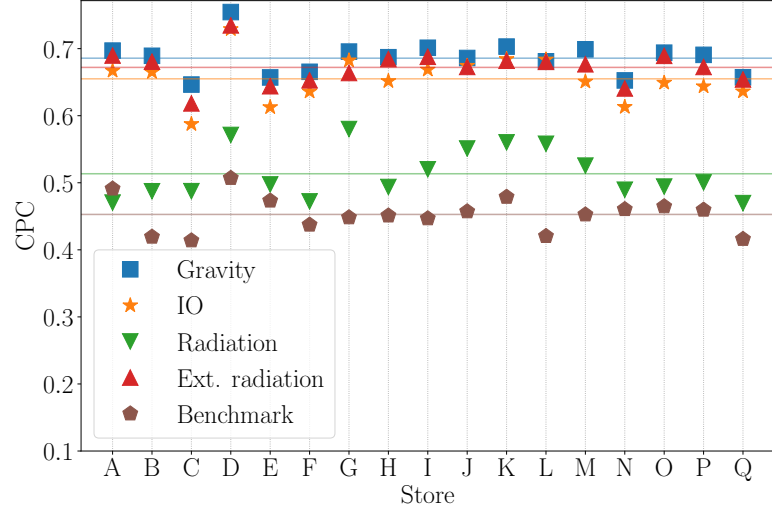
performance of these three models is store-dependent (see Figure 5.2b). In some stores, the gravity model has the lowest value of  $\text{NRMSE}_v$ ; in other stores, either the IO model or the extended radiation model attains the lowest value. The radiation and benchmark models perform worse than the other three models, achieving both lower CPC scores and higher values of  $\text{NRMSE}_v$  across all 17 stores. The poor performance of the radiation model is consistent with other studies of mobility systems on small spatial scales [60, 63, 70]. However, the radiation model performs better than the benchmark model, which is the worst of the five models.

Table 5.3. Mean CPC scores and  $\text{NRMSE}_v$  values from fitting the gravity, IO, radiation, extended radiation, and benchmark models to mobility-flow data from 17 supermarkets. We list the models in decreasing order of their mean CPC score. We highlight the best value in each column in bold.

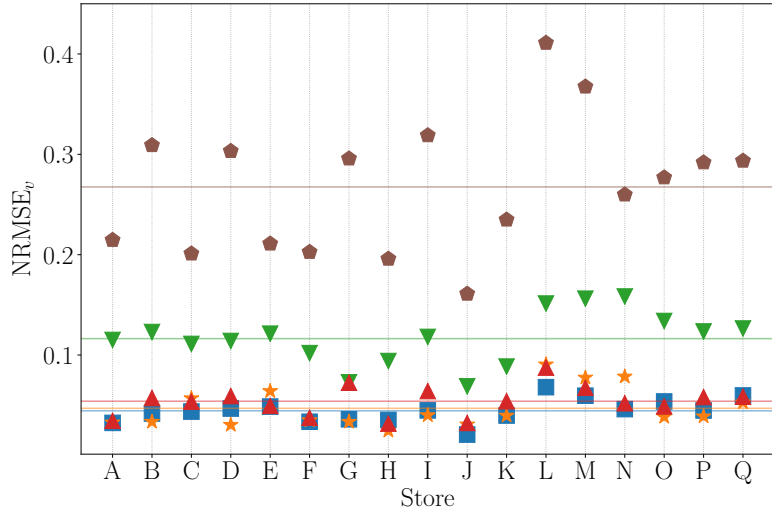
Model	Mean CPC	Mean $\text{NRMSE}_v$
Gravity	<b>0.686</b>	<b>0.045</b>
Ext. radiation	0.672	0.054
IO	0.655	0.047
Radiation	0.513	0.116
Benchmark	0.453	0.267

To further investigate the performance of the models, we examine the results of a single store (Store A) in more detail. Specifically, we examine the fitted OD matrix  $\mathbf{T}^{\text{model}}$ , the estimated number  $v_k(\mathbf{T}^{\text{model}})$  of visits, and the distance distribution of the OD trips for each of the mobility models. The results for this store are qualitatively similar to the results for the other stores.

In Figure 5.3, we compare the empirical number  $T_{ij}^{\text{data}}$  of trips with the estimated number  $T_{ij}^{\text{model}}$  of trips from each mobility model for each OD pair  $(i, j)$  of nodes in Store A. To evaluate the quality of  $T_{ij}^{\text{model}}$ , we create logarithmic bins from 1 to  $\max_{i,j}(T_{ij}^{\text{data}})$ . For each bin, we consider all OD pairs whose empirical number of trips lies within the bin; and we calculate the mean, median, and interquartile range for the estimated number  $T_{ij}^{\text{model}}$  of trips for these OD pairs  $(i, j)$ . See the black box plots in Figure 5.3. On average (both in terms of the mean and the median), the estimated numbers of trips from the gravity, IO, radiation, and extended radiation models are close to their empirical numbers, except for OD pairs with a large number of trips. For these OD pairs, the gravity, IO, and the extended radiation models underestimate the number of trips. The radiation model is effective at estimating the mean number of trips for most of the bins, but its overall performance is poor because of the large



(a) CPC scores



(b) NRMSE<sub>v</sub>

Figure 5.2. CPC scores and NRMSE<sub>v</sub> values when fitting the gravity, IO, radiation, and extended radiation models to mobility-flow data from 17 supermarkets. The solid lines show the mean value of CPC and NRMSE for each model.

variance in its estimates for each bin (see Figure 5.3c). The benchmark model performs poorly in all respects. It underestimates the number of trips for OD pairs with a large number of trips, and it overestimates the number of trips for OD pairs with a small number of trips. The poor performance is likely due to the benchmark model not taking distance between nodes into account. There tend to be more trips between nodes that are close together than between those that are far away from each other. The benchmark model treats both cases as the same, so it tends to underestimate the number of trips between nodes that are close together (which have a large number of trips between them) and overestimate the number of trips between nodes that are far away (which have a small number of trips between them).

In Figure 5.4, we compare the estimated number  $v_k(\mathbf{T}^{\text{model}})$  of visits that we compute from the OD matrix  $\mathbf{T}^{\text{model}}$  of the models with the number  $v_k(\mathbf{T}^{\text{data}})$  of visits that we estimate using the empirical OD matrix  $\mathbf{T}^{\text{data}}$  for Store A. We find that the gravity, IO, and extended radiation models are effective at estimating the number of visits for most nodes, except for some of the ones with a large number of visits. For these nodes, the three models overestimate the number of visits. The radiation model underestimates number of visits for most nodes (see Figure 5.4c), whereas the benchmark model overestimates the number of visits for most nodes.

In Figure 5.5, we compare the distribution of trip distances in our models with the empirical distribution. The gravity, IO, and extended radiation models have trip-distance distributions that qualitatively resemble the empirical distribution. Among the five models, the trip-distance distribution from the gravity model is closest to the empirical distribution. The IO model underestimates the number of long-distance trips (specifically, those above 60 m), and the extended radiation model overestimates the number of these long-distance trips. The trip-distance distributions of the radiation and benchmark models are qualitatively different from the empirical distribution (see Figure 5.5).

In summary, among the models that we examine, the gravity model best fits the empirical mobility-flow data. On average, it successfully explains about 69% of the OD trips in the data sets. It also is effective at estimating the number of visits to each node, with  $\text{NRMSE}_v$  values of about 0.045 (see Table 5.3). The extended radiation and IO models are close behind; on average, they successfully explain the data of about 65–67% of the OD trips. For the most part, these three models also yield trip-distance distributions that look similar to the empirical distribution. All three models also achieved a better fit than the benchmark model in all of the above investigations. The radiation model and the benchmark models do not fit the data well.

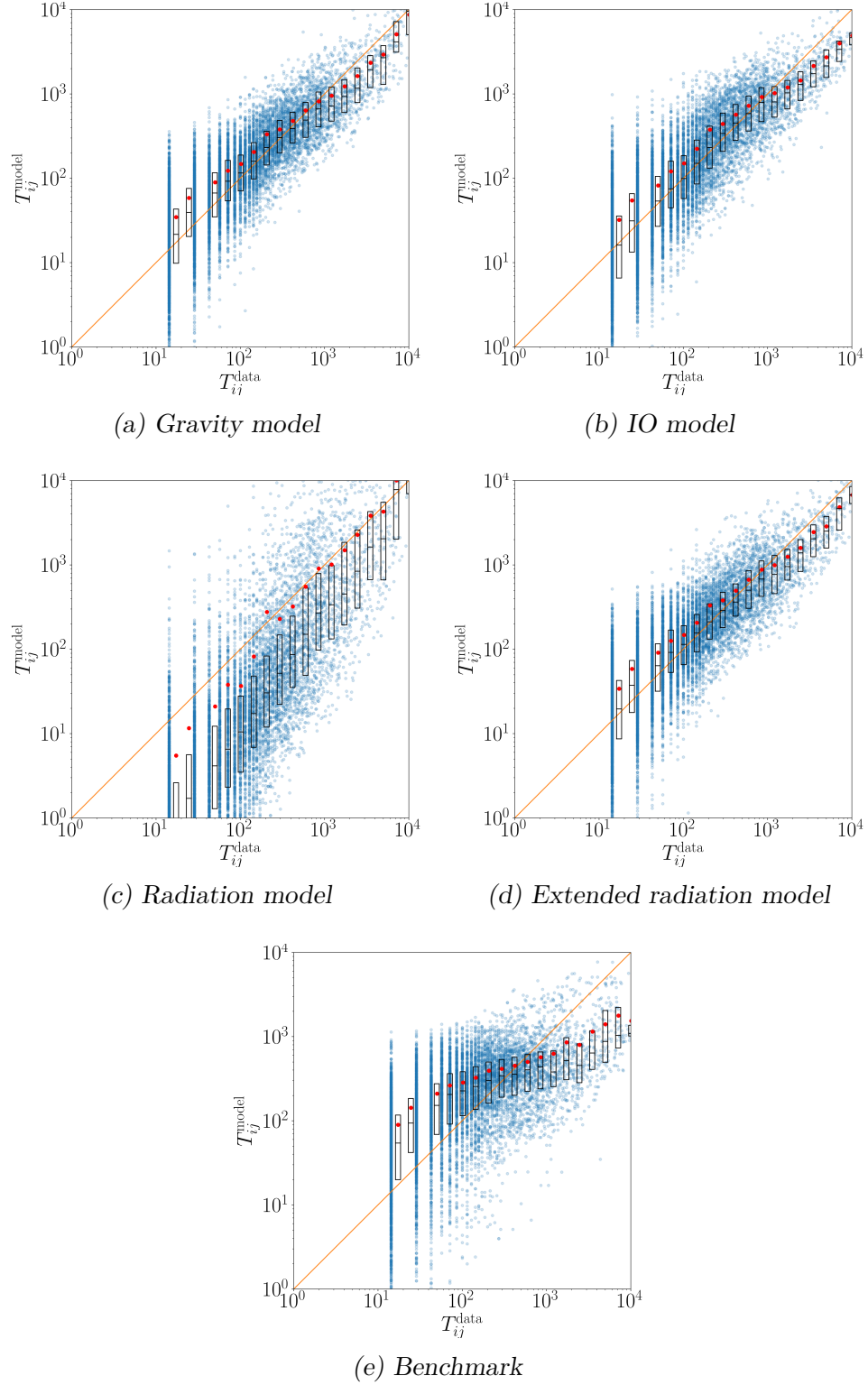


Figure 5.3. Comparison (blue dots) between the number of trips in the data ( $T_{ij}^{\text{data}}$ ) and the model estimation ( $T_{ij}^{\text{model}}$ ) for Store A. We also plot the mean number of trips that are estimated by the model (red dots) for each logarithmic bin of the data. The orange line is the identity line. Each box (in black) extends from the lower to the upper quartile values of the binned model estimate, and we draw a horizontal line at the median.

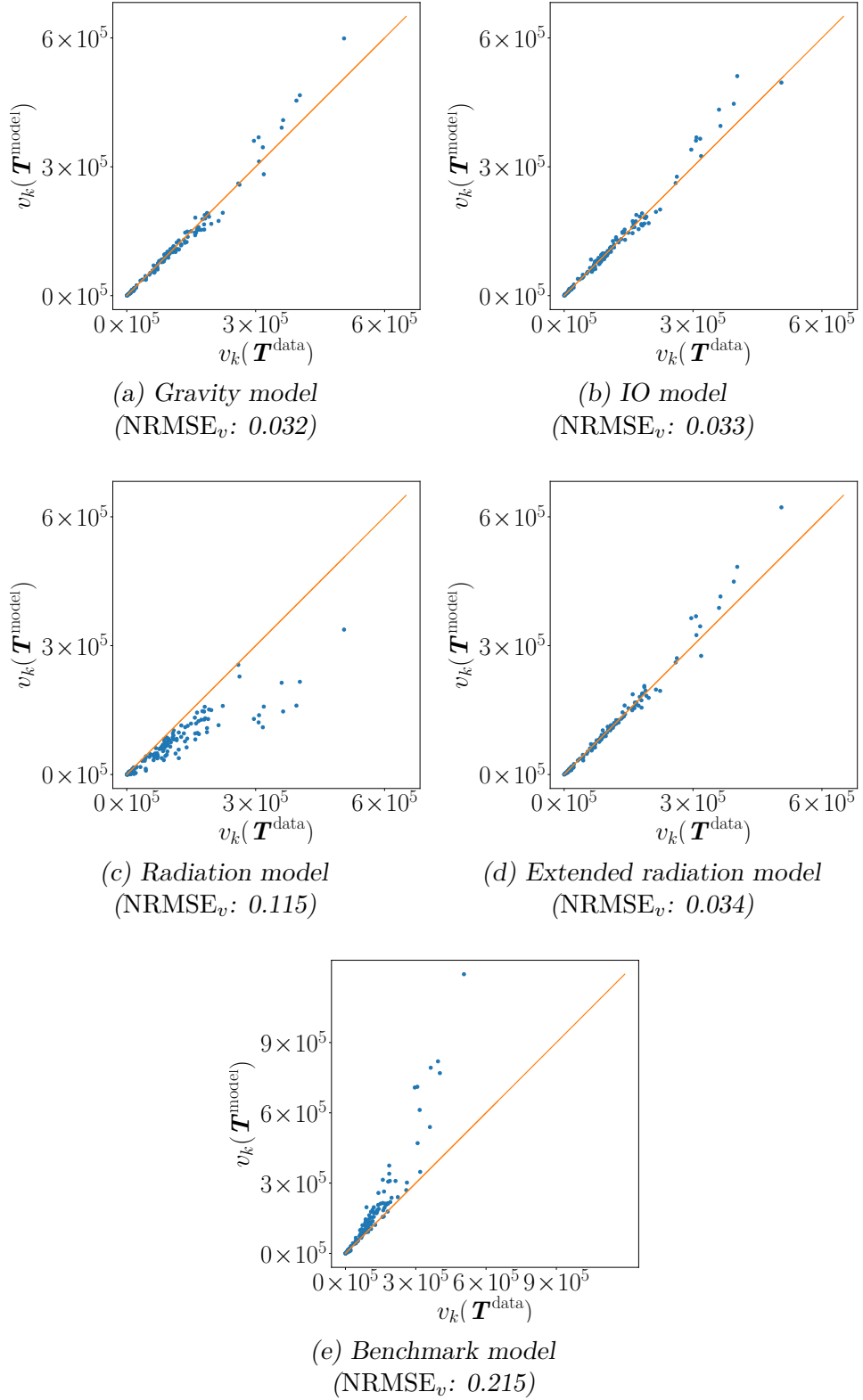


Figure 5.4. Comparison of the estimated number  $v_k$  of visits for each node  $k$  between the data and the mobility models for Store A. The orange line is the identity line. We see that the gravity, IO, and extended radiation models give good fits to the number of visits to each node.

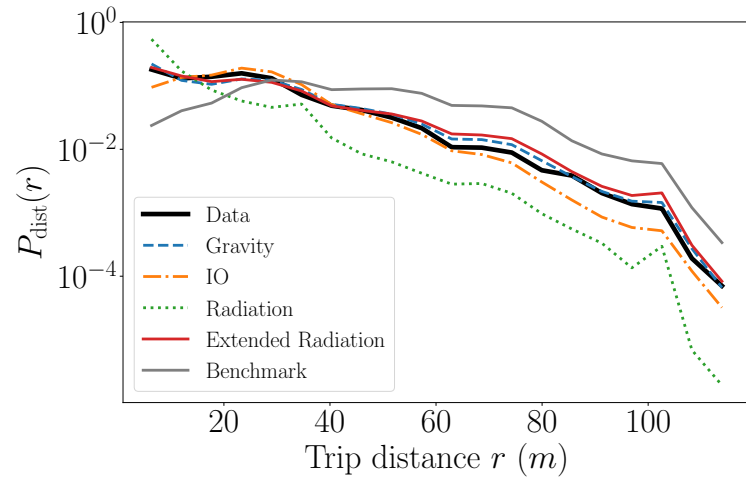


Figure 5.5. Probability density function (PDF) of the trip-distance distribution for Store A.

### 5.4.2 Sensitivity analysis: Size of the data set

We examine how the performance of the five mobility models depends on the number of shopping journeys (or, equivalently, the number of baskets) in the data set. We seek to understand the following question: How many shopping journeys are required for the models to fit well? We use the data set of Store A and calculate an OD matrix  $\mathbf{T}^{\text{data}}$  from the first  $J$  shopping journeys of the data set (ordered by the time of visit). We calculate these OD matrices for progressively larger values of  $J$ , fit the five models on these OD matrices, and measure the model performance by their CPC score. We plot the CPC scores as a function of the number  $J$  of shopping journeys in Figure 5.6. For all five models, the CPC score increases as we increase  $J$ , so the models fit the data progressively better with more data. The increase in the CPC score is initially steep, but then it flattens off. The CPC score reaches 95% of the maximum CPC score (which we achieve when using all 17814 shopping journeys in that data) after using 4000 journeys or more. (The number of journeys required to reach 95% of the maximum CPC score is similar in other stores.)

In summary, we observe that the performance of the models depends on the amount of data (specifically, on the number of shopping journeys in the data set). The performance (as measured using CPC) improves when we use more baskets, but with diminishing returns after about 4000 shopping journeys.

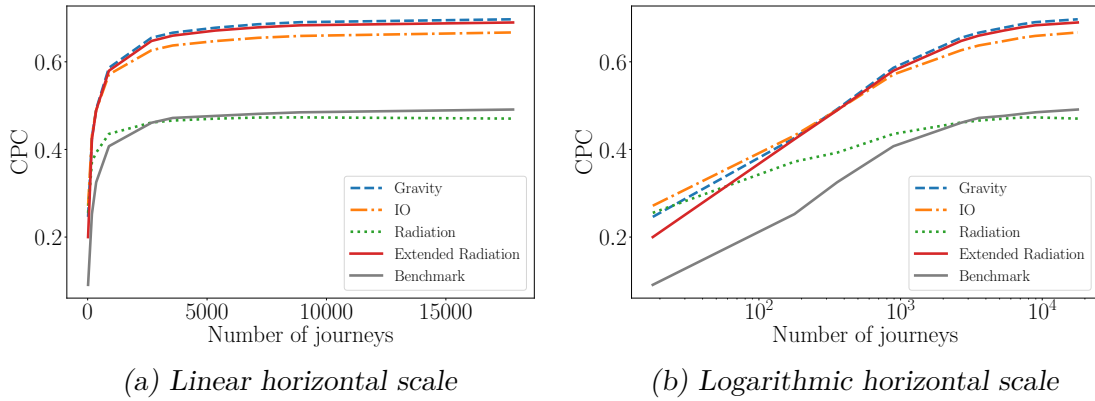


Figure 5.6. CPC dependence on the number of journeys. We plot the CPC of the five models as a function of the number of journeys in the data set (on which we fit the models) of Store A.

To understand this observation, suppose customers walk according to the random-walk-type model with transition matrix  $\mathbf{P}$  that we described in Section 5.3, starting at node 1 and ending at  $n$ . To obtain a good estimate for  $\mathbf{P}$  from a sample of random

walks (i.e., shopping journeys), we need a sufficiently large number of shopping journeys. Therefore, as we consider more shopping journeys, we obtain a better estimate of  $\mathbf{P}$ . When  $\mathbf{P}$  arises from a mobility model (e.g., a gravity model), we might expect the mobility model to fit better as our estimate of  $\mathbf{P}$  approaches the true transition matrix. To test whether this is what we are observing in Figure 5.6, we create  $10^6$  synthetic shopping journeys for Store A and apply the same procedure as above for empirical customers journeys.

We simulate each synthetic shopping journey as follows. Let  $\mathbf{P}$  be the empirical transition matrix, which we obtain from the empirical OD matrix  $\mathbf{T}^{\text{data}}$  by

$$P_{ij} = \begin{cases} T_{ij}^{\text{data}}/O_i, & \text{if } O_i > 0, \\ 0, & \text{otherwise.} \end{cases} \quad (5.16)$$

We generate a shopping journey by simulating a random walk on the fully connected network with  $n$  nodes. A walker starts at node 1 (i.e., the entrance node) and, at each step, it moves from its current position  $i$  to a random destination node  $j$  with probability  $P_{ij}$ . When a walker reaches node  $n$  (i.e., the till node), it finishes its journey with probability  $1 - D_n/O_n$ ; otherwise, it chooses its next destination  $j$  with probability  $P_{nj}$ . The nodes that a walker visits in this way form a shopping journey. Similar to a shopping journey in a real store, a synthetic shopping journey starts at 1 and ends at  $n$ . Furthermore, if we aggregate a sufficiently large number  $J$  of shopping journeys into an OD matrix  $\mathbf{T}$ , the matrix  $\mathbf{T}C_s/J$  (a normalized version of  $\mathbf{T}$ ), where  $C_s$  is the number of empirical journeys, converges to the empirical transition matrix  $\mathbf{T}^{\text{data}}$ .

Just as for the empirical shopping journeys, we aggregate the first  $J$  synthetic shopping journeys (for some fixed  $J$ ) into an OD matrix  $\mathbf{T}$ , fit the five mobility models to  $\mathbf{T}$ , and record the CPC score for each model. We do this for progressively larger  $J$  and plot the CPC scores as a function of  $J$  in Figure 5.7. We compare the CPC curve with the one that we obtain when using empirical journeys and find a strong agreement between them (see Figure 5.7). These results therefore give further indication that the models give a good fit when we use at least 4000 shopping journeys for stores of similar size as Store A. The precise number of journeys required depends on the size of the transition matrix  $\mathbf{P}$  and therefore on the number of nodes in the store network. The networks that we consider are of similar size, and 4000 journeys appear to be a good estimate to the minimum number of journeys required for the models to fit well.



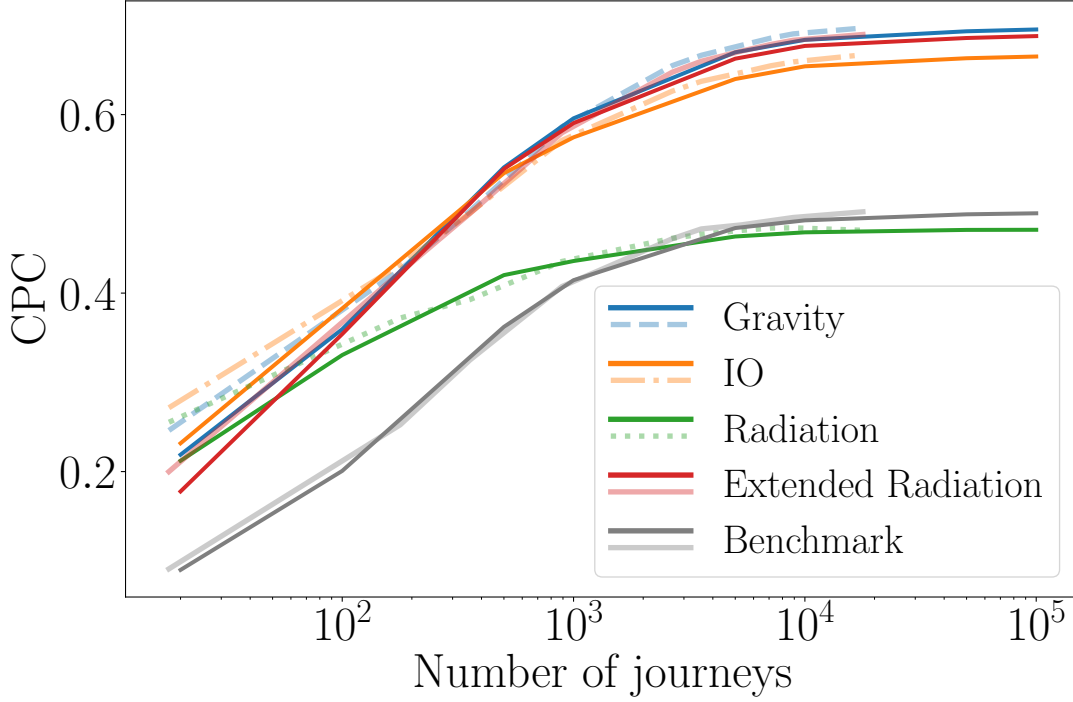


Figure 5.7. CPC versus the number of journeys, where we use synthetic shopping journeys for Store A. We plot (as solid curves) the CPC of the five models as a function of the number of synthetic journeys on which we fit the models. The transparent curves are the CPC curves from Figure 5.6.

### 5.4.3 Sensitivity analysis: Parameter dependence of models

We explore how the performance of the gravity, IO, and extended radiation models depends on their respective model parameter values. For each model, let  $p_{\text{opt}}$  to be the optimal parameter value for Store A. We calculate the CPC scores for parameter values between 0 and  $10p_{\text{opt}}$  (see Figure 5.8). For each of the models, we observe progressively smaller CPC scores for parameter values that are progressively farther away from  $p_{\text{opt}}$ , so model performance depends on the parameter value. The decrease in CPC score with distance from  $p_{\text{opt}}$  is steepest for the gravity model, second-steepest for the IO model, and shallowest for the extended radiation model. Interestingly, the CPC score for the extended radiation model plateaus as  $\alpha \downarrow 0$  at a value close to the maximum CPC score. This suggests that a parameter-free special case of the extended radiation model, which we obtain by setting  $\alpha = 0$ , may perform well. (We do not explore this special case in this thesis.)

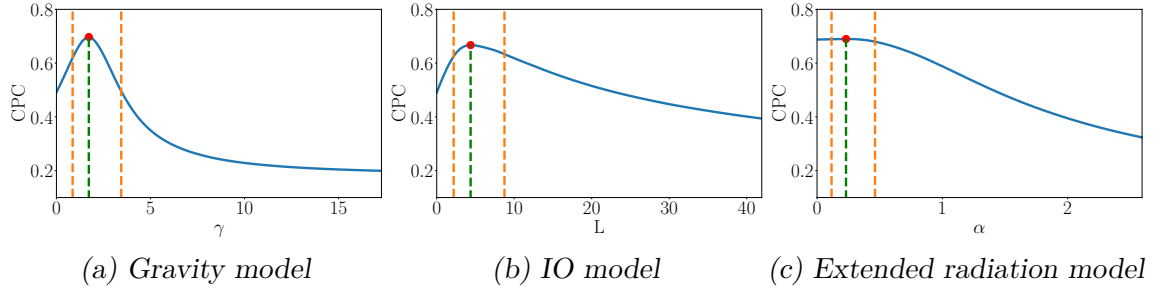


Figure 5.8. CPC dependence on the model parameter for the gravity, IO, and extended radiation models in Store A. We highlight the maximum CPC score with a red dot. The orange vertical lines are at  $0.5p_{\text{opt}}$  and  $2p_{\text{opt}}$ .

#### 5.4.4 Evaluation: Model performance on estimating trips in unseen data

We conduct two series of tests to analyse the performance of the gravity, IO, and extended radiation models in estimating mobility flow for a time period or a store for which we do not know the optimal parameter value. As we showed in Section 5.4.3, the performance of each of these models depends on the value of its associated parameter. In each test, we estimate a mobility flow using the optimal parameter value from a different time period of the same store (in our first series of tests) or from a different store for the same time period (in our second series of tests). We compare for each model the achieved CPC value ( $\text{CPC}_a$ ) with the maximum CPC value ( $\text{CPC}_{\text{max}}$ ), which we obtain when using the optimal parameter value, by computing their ratio

$$R = \frac{\text{CPC}_a}{\text{CPC}_{\text{max}}} \in [0, 1]. \quad (5.17)$$

A value of  $R$  that is close to 1 indicates that the estimated mobility flow using a parameter value that is optimal for a different time period or for a different store fits the empirical flow just as successfully as using the optimal parameter value. In other words, these tests allow us to investigate whether the optimal parameter values of a model differ significantly (in terms their model performance measured in the CPC score) between different time periods of the same store or between different stores.

In the first series of tests, we split the 91-day data set of each store into two parts. The first part is the mobility-flow data from the first 60 days; for this subset of the data, we find the optimal parameter value for each model. The second part is the mobility-flow data of the remaining 31 days; we estimate the mobility flow during this period using a mobility model with the empirical values of  $O_k$  and  $D_k$  (i.e., the number of OD trips that, respectively, start and end in zone  $k$ ) from this period and

the optimal parameter value from the initial 60-day time period. We perform one test for each store, so there are 17 tests per model. The mean value of  $R$  is about 0.97–0.98 for each of the three models (see Table 5.4). Therefore, the model parameter values do not change much across different time periods of the same store. This also suggests that to estimate mobility flow of a store during some (sufficiently long) time period, we only need to know the values of  $O_k$  and  $D_k$  (which we can estimate from purchase data) during that time period and the optimal parameter value for a different (sufficiently long) time period of the same store.

*Table 5.4. Mean, median, and minimum values of  $R$  (i.e., the ratio of our achieved CPC value to the maximum one) for the 17 stores in our first series of tests.*

Model	Mean $R$	Median $R$	Minimum $R$
Gravity	0.975	0.975	0.959
IO	0.978	0.977	0.963
Ext. rad.	0.975	0.973	0.960

In each test in our second series of tests, we estimate the 91-day mobility flow of one store using a mobility model with the optimal parameter value of another store for the same time period. We perform one test for each possible ordered pair of distinct stores, so there are  $17 \times 16 = 272$  tests in total for each model. The mean value of  $R$  is above 0.99 for each of the three models (see Table 5.5); this suggests that the differences in model parameter values across stores are small and have minimal effect on the performance of the models. We therefore conclude that we can estimate the mobility flow of one store using the optimal parameter value from another store. As the 17 stores all have different store layouts, these results also suggest that if we change the layout of a store, we expect the optimal model parameter values to not change appreciably.

*Table 5.5. Mean, median, and minimum values of  $R$  (i.e., the ratio of our achieved CPC value to the maximum one) for the 272 tests in our second series of tests.*

Model	Mean $R$	Median $R$	Minimum $R$
Gravity	0.996	0.998	0.976
IO	0.994	0.996	0.926
Ext. rad.	0.999	1.000	0.980

## 5.5 Incorporating zone–zone correlations

In a supermarket, there are complementary items, which are frequently bought together (e.g., strawberries and cream). One can measure the ‘complementarity’ of two items by their product–product correlation, which is the proportion of baskets that include both items. In the models in Section 5.3, the attraction values  $f_{ij}$  depend only on the popularity (measured in shopping visits) of each zone; they do not take into account the product–product correlation. In this section, we incorporate zone–zone correlations (as a proxy for product–product correlations<sup>5</sup>) into the attraction values  $f_{ij}$  for the five mobility models at an expense of an additional calibration parameter and examine whether this improves the fit to the data for each model. We measure the zone–zone correlation  $r_{ij}$  between zones  $i$  and  $j$  (for  $i \neq j$ ) by

$$r_{ij} = \frac{b_{ij}}{b_i} \times \left( \frac{b_j}{C_s} \right)^{-1} = \frac{b_{ij} C_s}{b_i b_j}, \quad (5.18)$$

where  $b_k$  is the number of shopping journeys that visit  $k$ , the quantity  $b_{ij}$  is the number of shopping journeys that visit both  $i$  and  $j$ , and the  $C_s$  is the total number of shopping journeys. Note that  $r_{ij} = r_{ji}$  for all  $i$  and  $j$ . The first term  $b_{ij}/b_i$  in Equation (5.18) is also called the *confidence* of the association  $i \rightarrow j$  [61] and measures the proportion of shopping journeys that visit  $i$  and that also visit  $j$ . The second term  $b_j/C_s$  in Equation (5.18) is the proportion of shopping journeys that visit  $j$ . The quantity  $r_{ij}$  is the ratio between these two terms and measures how much more likely a randomly chosen shopping journey that visited  $i$  has also visited  $j$  compared to a randomly chosen shopping journey. (We choose uniformly at random.) For example, a value of  $r_{ij} = 2$  indicates that a shopping journey that visited  $i$  is twice as likely to have visited  $j$  than a randomly chosen shopping journey. When  $r_{ij} > 1$ , a shopping journey that visited  $i$  is more likely to have visited  $j$  as well (than a general shopping journey); similarly, when  $r_{ij} < 1$ , a shopping journey that visited  $i$  is less likely to have visited  $j$  as well (than a general shopping journey). For each of the five mobility models in Section 5.3, we formulate an extended model, which we call a *correlation-biased* model, with attraction values  $\bar{f}_{ij}$  given by

$$\bar{f}_{ij} = (1 - t) \frac{f_{ij}}{\sum_{k,l} f_{kl}} + t \frac{r_{ij}}{\sum_{k,l} r_{kl}}, \quad (5.19)$$

---

<sup>5</sup>We cannot compute the product–product correlations, because our data consists of shopping journeys that are on a zone level. We do not know which items customers bought in their shopping journeys.

where  $f_{ij}$  are the attraction values of the original model and  $t \in [0, 1]$  is an additional calibration parameter. Note that we normalize  $f_{ij}$  and  $r_{ij}$  to ensure they are on the same order of magnitude.  $\bar{f}_{ij}$  is a convex combination of the normalized quantities of  $r_{ij}$  and  $f_{ij}$ . When  $t = 1$ , we have  $\bar{f}_{ij} = f_{ij} / \sum_{k,l} f_{kl}$ , so we recover the original model, as  $\mathbf{T}^{\text{model}}$  is invariant with respect to scaling of  $f_{ij}$ . Therefore, a correlation-biased model performs at least as good as the original model. When  $t = 0$ , we have  $\bar{f}_{ij} = r_{ij} / \sum_{k,l} r_{kl}$ , so we disregard the original attraction values  $f_{ij}$ . We calibrate  $t$  and any additional model parameters by maximizing the CPC score using the python function `minimize` from the `scipy.optimize` package (Version 1.14.3) [46]. The function `minimize` requires an initial value of all model parameters as an input. We set the initial value of  $t$  to 0 and any model parameters to  $p_{\text{opt}}$ , the optimal parameter value in the original model for a given store.

We fit the correlation-biased models for the gravity, IO, radiation, extended radiation, and benchmark models to all 17 stores and report the difference  $\delta_{\text{CPC}}$  in CPC score between the original model and the correlation-biased model in Table 5.6. Incorporating zone–zone correlations yields a very small increase in CPC for the IO and benchmark models and a moderate increase for the radiation model. However, the performance of the gravity and extended radiation models, which gave the best fit to the data in the original form, is not improved by incorporating zone–zone correlations.

Table 5.6. Difference  $\delta_{\text{CPC}}$  in CPC score between the correlation-biased models (that incorporate zone–zone correlations) and the original models for the gravity, IO, radiation, and extended radiation models. For each model, we show the mean, minimum, and maximum value of  $\delta_{\text{CPC}}$  across 17 supermarkets.

Model	Min $\delta_{\text{CPC}}$	Mean $\delta_{\text{CPC}}$	Max $\delta_{\text{CPC}}$
Gravity	0.00	0.00	0.01
IO	0.00	0.01	0.03
Radiation	0.05	0.10	0.14
Ext. radiation	0.00	0.00	0.01
Benchmark	0.01	0.02	0.02

## 5.6 Summary and Discussion

We employed several population-level mobility models to investigate customer mobility flow between zones in supermarkets, whose spatial scales are much smaller than in previous uses of these models. We estimated origin–destination (OD) matrices, which describe empirical mobility flow, for 17 supermarkets from anonymized, ordered

customer-basket data (where we defined a customer OD trip as either a journey between consecutive purchases, a journey from the entrance to the first purchase, or a journey from the last purchase to the tills). We fit the mobility models to these empirical OD matrices, analysed their fit to them, performed a sensitivity analysis of the models with respect to the size of the data set and their model parameters, and examined how the model parameters change across different time periods and different stores.

Among the five models that we studied, the gravity model (with a power-law deterrence function) gave the best fit to the empirical mobility flow (it successfully estimated about 69% of the flow on average), and the extended radiation and Schneider’s intervening-opportunities (IO) models were almost as successful. This illustrates that one can successfully use population-level mobility models for applications on spatial scales of tens to hundreds of metres. All three models also performed significantly better than a simple benchmark model, which suggests that the three models capture aspects of the mobility that a simple benchmark model does not capture.

In our investigation, we estimated the number  $v_k$  of visits to each node  $k$  from mobility flow by assuming that each customer traverses a shortest path, and we found that our estimations from the OD matrices from the gravity, IO, and extended radiation models agree well with the total number of visits that we estimated from empirical OD matrices. Additionally, the gravity, IO, and extended radiation models yield trips with similar distance distributions to the empirical distribution. However, consistent with other studies on small spatial scales (which generally have been in intra-urban settings) [60, 63, 70], the basic radiation model was not successful at reproducing features of the data.

We also examined how the performance of the models depends on the number of shopping journeys. We observed that the performance increases steeply until the size of the data set exceeds 4000 shopping journeys. When the size of the data set increases beyond 4000 shopping journeys, the performance increases only slightly. This suggests that the models achieve a good fit to the empirical mobility flow when we have at least around 4000 shopping journeys for the stores that we examined. We anticipate that this number increases with the size of the OD matrix and therefore with the size of a store. Further analysis is required to understand precisely how this number depends on the size of a store.

The gravity, IO, and extended radiation models each have one parameter, and their performance depends on the value of that parameter. In our investigation, we found that it is sufficient to use the optimal model parameters that we calibrated on

a single store to give good estimates of the mobility flows of all other stores. The only additional information that we needed for the other stores is the number of trips from and to each node; one can estimate these quantities from the purchase data of these stores. For a given store, we were also successful at using the models to estimate the mobility flow of a time period using a parameter value that was calibrated on data from another time period of the same store.

Our success at translating optimal parameter values across both stores and time periods suggests that one can estimate the mobility flow of customers for all stores from purchase data alone (without needing ordered customer-basket data). Additionally, if we have a model (which we call a *sales model* in Chapter 6) for estimating the number of customers that purchase items at each node in new store layouts, we can combine it with mobility models to estimate the mobility flow in new store layouts. This approach provides a potentially valuable testbed for experimentation by supermarket companies before trying out new store layouts. If a retailer is interested in reducing congestion or increasing foot traffic along a certain aisle in a store, they can use this approach to explore different feasible store layouts and select the best store layout with respect to their objective function. We can also systematically explore feasible store layouts using an optimization algorithm (e.g., the one presented in Chapter 6). More generally, such an approach is not restricted to supermarkets, as it can be used to estimate mobility flow of pedestrians in a building or other space, where their routes are not prescribed.<sup>6</sup> Examples for where our approach is potentially applicable include exhibitions, markets, and museums.<sup>7</sup>

Finally, based on the intuition that customers tend to buy items that are complementary to each other, we proposed an extension to our models that incorporates correlations between zones. These zone–zone correlations  $r_{ij}$  measures how much more likely a shopping journey that visits  $i$  (to buy some items) also visits  $j$  compared to a general shopping journey. In these extensions, which we call correlation-biased models, we introduce an extra term to the attraction values; this extra term includes  $r_{ij}$  and an additional calibration parameter. We thereby introduce a bias of flow between nodes with large zone–zone correlations, and the calibration parameter controls the magnitude of the bias. We formulated a correlation-biased model for each of the mobility models that we examined. We found that these correlation-biased models do not yield a better fit to the data for the gravity and extended radiation model

---

<sup>6</sup>When pedestrians take prescribed routes (e.g., Ikea stores), the mobility flow is known (up to a scaling factor) and we do not need models to estimate it.

<sup>7</sup>In some museums and exhibitions, the route that visitors take is prescribed.

(the two models that fit data the best) and only gave a very small improvement in the fit for the IO model. There are a number of possible reasons for this observation. Firstly, zone–zone correlations may not capture the notion of complementary items sufficiently well. In Store A, each customer purchases on average 30 items and visits on average 14 different zones where they purchase items. The majority of items in each baskets are likely not complementary, but every pair of items that are in different zones contribute to the zone–zone correlations  $r_{ij}$ . Secondly, customers may not take a direct trip between zones that contain complementary items in many cases, resulting in small number of trips between complementary items. Thirdly, we may not want to use symmetric correlations (i.e., where  $r_{ij} = r_{ji}$ ), as for certain complementary items (that are often unplanned purchases), one item of the pair is usually bought first. For example, people may tend to buy strawberries first and then think about buying cream, whereas fewer people might buy cream first and then buy strawberries. In future work, it would be interesting to test different measures of zone–zone correlations that take some of the above factors into account and see whether they improve the model.

Our work is not without limitations, as we made a number of choices and assumptions. Our results may change, if we make different choices or if the assumptions do not hold in reality.

In our investigation, we inferred empirical mobility flow from anonymized, ordered basket data of the mobility of a relatively small sample of customers (approximately 7%) from 17 supermarkets. Naturally, this sample also has certain biases, as our data consists mainly of baskets from regular customers. It is likely that these customers possess better knowledge than other customers of the stores in which they shop (given that they do so regularly), so their mobility patterns may not be representative of all customers of a given store.

We have also neglected temporal information and seasonality effects in our data by aggregating the mobility flow over  $\tau = 91$  days. We expect mobility flow to be different at different times of the day (and on weekdays versus weekends) and at different times of a year (e.g., during certain holidays). We also expect different zones of a store to be the most congested ones at different times. Given sufficient data (e.g., at least 4000 baskets for each segment for one of our examples), one can apply mobility models to data that is segmented by time of day or the day of a year and then compare the parameter values from independent fitting to data in different time periods.

Another consideration is the choice of space discretization and spatial resolution, and it is necessary to examine how such choices affect qualitative results of the mobility



models. In our work, we divided each store into zones of approximately similar size, with zone lengths of about 7 m. If one can create a fully automated way of creating store networks with prescribed zone lengths, it would be interesting to see how the performance of our models depends on the size of the zones.

## Chapter 6

# Integrating human-mobility models with queueing networks

In this chapter, we integrate human-mobility models with queueing networks to estimate congestion, measured by the total mean queue size  $Q$ , in supermarkets; and we use a simulated-annealing (SA) algorithm to identify store layouts with low congestion. Our approach has four components:

- (1) a congestion model, based on queueing networks, that estimates congestion from an origin–destination (OD) matrix  $\mathbf{T}$  (representing the mobility flow);
- (2) a sales model that estimates the number of trips that arrive at and depart from each node  $k$  under a new store layout;
- (3) a mobility model (specifically, a gravity model) that determines the dependence of the OD matrix  $\mathbf{T}$  on the store layout; and
- (4) an optimization algorithm (specifically, an SA algorithm) that finds store layouts with less congestion than the original store layout.

We describe these components in detail in Sections 6.1–6.4.

### 6.1 Congestion model

Our congestion model is a multi-class queueing network with  $n(n - 1)$  groups of customers; there is one (customer) group for each origin–destination (OD) pair  $(i, j)$ . Each node  $k$  is a single-server queue with exponential service rate  $\mu_k$ . There are three inputs: The first input is an origin–destination (OD) matrix  $\mathbf{T}$  with entries  $T_{ij}$ , which we calculate using one of the doubly-constrained mobility models in Section 5.3; the

entry  $T_{ij}$  records the number of (estimated) OD trips from  $i$  to  $j$  over some time period  $\tau$ . Each trip represents a segment of a customer's journey either between consecutive purchases in different zones, between the entrance and the first purchase, or between the last purchase and the tills (see Section 5.1). As we use a doubly-constrained model, we have  $O_k := \sum_j T_{kj} = \sum_i T_{ik} = D_k$  for all  $k$  except  $k = 1$  (entrance) and  $k = n$  (tills). As explained in Section 5.3, the quantities  $O_k$  and  $D_k$  are equal to the number of shopping visits to each node  $k$ . The other two inputs to our congestion model are the store network  $G$  with its associated distance matrix  $\mathbf{\Lambda}$  and the service rates  $\mu_k$  for each node  $k$ , which we can estimate from the mean customer dwell time at  $k$  (described later in this section).

In our congestion model, new customers arrive from outside the system at each node  $i$  (not just the entrance) of a store network (representing a supermarket) according to a Poisson process with rate  $O_i/\tau$ . After entering the system, each customer chooses a random destination  $j$  with probability  $P_{ij}$ .<sup>1</sup> Customers traverse the network by taking a shortest path from node  $i$  to node  $j$ , chosen uniformly at random from all shortest paths from  $i$  to  $j$ . We say that these customers take a trip (or an OD trip) from  $i$  to  $j$ . Therefore, customers who take a trip from  $i$  to  $j$  arrive at the system at rate  $P_{ij}O_i/\tau$ . We group customers by the OD trip that they take, so there are  $n(n-1)$  groups in total. Customers queue at each node that they visit on their shortest path from  $i$  to  $j$  (including  $i$  and  $j$ ) to be served for both traversal and shopping. At each node  $k$ , there are both customers who traverse the node  $k$  (to go their destination  $j \neq k$ ) and those who shop at  $k$  (i.e., those who start or end their trip at  $k$ ). There is a single service rate  $\mu_k$  at a node, so the node serves both types of customers equally. After a customer arrives at their destination node  $j$  and is served there, we remove it from the network. The quantity  $T_{ij}$  is then the mean number of customers who take a trip from  $i$  to  $j$  during a time period of length  $\tau$ .

The arrival and departure processes of customers in our model differ from the corresponding processes in the model in Section 5.3 (which we call the ‘‘Chapter 5 model’’). In the Chapter 5 model, new customers arrive at the entrance and take trips to random destinations based on a transition matrix  $\mathbf{P}$  with entries

$$P_{ij} = \begin{cases} T_{ij}/O_i, & \text{if } O_i > 0, \\ 0, & \text{otherwise.} \end{cases} \quad (6.1)$$

---

<sup>1</sup>We can also use an equivalent formulation in which there are  $n-1$  types of customers who arrive at  $i$  according to a Poisson process. There is one type of customer for each possible destination  $j \neq i$ . The rate of arrival for customers arriving at  $i$  with destination  $j$  is  $P_{ij}O_i/\tau$ .

Therefore, there is only a single group of customers (i.e., walkers) who move according to the transition matrix  $\mathbf{P}$  in the Chapter 5 model, whereas there are  $n(n-1)$  different groups of customers in present model (which we henceforth call the “Chapter 6 model”). In the Chapter 6 model, customers of each group start at a different node in a network and leave after taking exactly one trip to a specified destination. By contrast, customers always start at the entrance node and leave at the till node after taking one or more trips in Chapter 5 model. The Chapter 6 model is clearly less realistic than the Chapter 5 model, because in reality customers do not appear and disappear inside a store. However, the independence of the queue sizes  $X_k$  of each node  $k$  is not guaranteed in the Chapter 5 model.<sup>2</sup> In other words,  $X_k$  may not satisfy Equation (2.36) in the Chapter 5 model. The Chapter 6 model, however, does satisfy Equation (2.36), as it is a multi-class queueing network model, which is why we use the Chapter 6 model. We can construe the Chapter 6 model as an approximation of the Chapter 5 model in which we assume that the queue sizes are independent of each other.

The Chapter 6 model is a multi-class queueing network with  $n(n-1)$  groups of customers; each group is associated with an OD pair  $(i, j)$ . Customers in the group associated with the OD pair  $(i, j)$  enter the system at node  $i$  with an arrival rate  $\lambda_{0i}^{(i,j)} = P_{ij}O_i/\tau = T_{ij}/\tau$ . They walk according a transition matrix  $\mathbf{P}^{(i,j)}$  (which is unrelated<sup>3</sup> to  $\mathbf{P}$ ), and we show in Appendix D.1 that its entries are

$$P_{kl}^{(i,j)} = \begin{cases} \frac{s_{kl}}{\sum_{l'} s_{kl'}}, & \text{if } \sum_{l'} s_{kl'} > 0, \\ 0, & \text{otherwise,} \end{cases} \quad (6.2)$$

---

<sup>2</sup>As described in Section 2.3.1, the queue sizes are independent if all customers take a random walk on the store network with the same transition matrix  $\mathbf{P}$  or if customers belong to one of  $K$  groups (for some integer  $K \geq 1$ ) with each group having its own transition matrix. For the latter case (i.e., when we have  $K$  groups), we also require that customers from each group arrive at the network independently of the customers from other groups. Note that the Chapter 5 model is not an example of the former case, even though every customer has the same transition matrix  $\mathbf{P}$ , because it is not a random walk on the store network; customers take random shortest-path trips to any node (other than to their current position). We can view the Chapter 5 model as a model in which each customer belongs to one of  $K = n(n-1)$  groups; each group corresponds to an OD trip  $(i, j)$  that the customers in that group take. However, customers of each group do not arrive independently from another: once an  $(i, j)$ -customer (i.e., a customer who belongs to the group associated with the OD pair  $(i, j)$ ) finishes its trip, it chooses another destination  $k$ , so it becomes a  $(j, k)$ -customer for some  $k$ . Therefore, the arrival process of  $(i, j)$ -customers is not independent from the arrival process of  $(j, k)$ -customers, and the queue sizes are thus not independent.

<sup>3</sup>The matrix  $\mathbf{P}$  determines the final destination of walkers; its entries  $P_{ij}$  records the probability of a walker who enters the system at  $i$  to choose a destination  $j$ . The matrix  $\mathbf{P}^{(i,j)}$  determines the path of a customer who enters the system at  $i$  and has chosen the destination  $j$  takes; in our case, the path is a random shortest-path from  $i$  to  $j$ .

where  $s_{kl}$  is equal to the number of shortest paths from  $i$  to  $j$  that traverse the directed edge  $(k, l)$  if  $(k, l) \in E$  and is equal to 0 otherwise. We show in Appendix D.2 that the arrival rate  $\lambda_k^{(i,j)}$  of customers from group  $(i, j)$  to node  $k$  is equal to the arrival rate  $\lambda_{0i}^{(i,j)}$  from outside the system multiplied by the fraction  $\omega_{ikj}$  of shortest paths from  $i$  to  $j$  that traverse  $k$ . That is,

$$\lambda_k^{(i,j)} = \omega_{ikj} \lambda_{0i}^{(i,j)} = \frac{\omega_{ikj} T_{ij}}{\tau}. \quad (6.3)$$

The total arrival rate  $\lambda_k$  to node  $k$  is

$$\lambda_k = \sum_{i,j} \lambda_k^{(i,j)} = \sum_{i,j} \frac{\omega_{ikj} T_{ij}}{\tau} = \frac{v_k}{\tau}, \quad (6.4)$$

where  $v_k$  is the number of visits to node  $k$ . We estimate  $v_k$  from  $\mathbf{T}$  using Equation (5.14). (That is why we also examined the error in  $v_k$  of our mobility models in Chapter 5; we showed that the gravity model that we used gives a good fit in  $v_k$ .)

We assume that the service rates  $\mu_k$  satisfy

$$\mu_k > \lambda_k, \quad \text{for all } k, \quad (6.5)$$

so there exists a stationary state for the queue sizes. If we have information about the mean customer dwell time  $W_k$  at each node  $k$ , we can infer the service rates  $\mu_k$  using

$$\mu_k = \frac{1}{w_k} + \lambda_k, \quad (6.6)$$

as we explained in Section 4.5 (Equation (6.6) corresponds to Equation (4.63)).

Because we do not have empirical data for the service rate, we assume for simplicity that the service rates are homogeneous (i.e.,  $\mu_k = \mu > 0$  for all  $k$ ). Equation (6.5) then becomes  $\mu > \lambda_{\max}$ , where  $\lambda_{\max} = \max_k \lambda_k$ .

As in Chapter 4, we use the total mean queue size  $Q$  to measure congestion. The total mean queue size  $Q$  is equal to the mean number of customers in a store. A store layout that minimizes  $Q$  also minimizes the mean trip time by Little's Law, as we discussed in Chapter 4. The value of  $Q$  and the store layout that minimizes it both depend on the value of  $\mu$ . To identify store networks in the small- $\mu$  regime (when  $\mu$  is sufficiently small) and in the large- $\mu$  regime (when  $\mu$  is sufficiently large), we also perform simulations that attempt to minimize  $\lambda_{\max} = \max_k \lambda_k$  and  $\lambda_{\text{total}} = \sum_k \lambda_k$ . As we explained in Section 4.3.2, store layouts that minimize  $Q$  for a given value of  $\mu$  also minimize  $\lambda_{\max}$  if  $\mu$  is sufficiently small and minimize  $\lambda_{\text{total}}$  if  $\mu$  is sufficiently large. The measures  $Q$ ,  $\lambda_{\max}$ ,  $\lambda_{\text{total}}$  are correlated with each other, as  $Q$  is a sum that is dominated by the terms from nodes with large values of  $\lambda_i$ , so store layouts with smaller values of  $Q$  often also have smaller values of  $\lambda_{\max}$  and  $\lambda_{\text{total}}$ .

## 6.2 Sales model

We need a hypothesis for how  $O_k$  and  $D_k$  (i.e., the number of shopping visits, if  $k \neq 1, n$ ) change when we change the location of a node  $k$ . We assume that  $O_k$  and  $D_k$  depend only on the items inside a zone and not on the zone’s location. Therefore, when we change the location of a node  $k$ , we assume that the node has the same values of  $O_k$  and  $D_k$  in the new location. Put another way, we assume that the number of shopping visits at node  $k$  (i.e., the number of times that customers visit  $k$  to purchase items) for  $k \neq 1, n$  is independent of its location. We call this model the *location-independent sales model*. For nodes with many essential items, such as bread and milk, this assumption seems justifiable, as customers buy such items regardless of their location in a store. However, we anticipate that this assumption breaks down for nodes that include mostly items that are either less essential or purchased with less (or no) planning. We view the problem on an aggregate (flow) level, and we do not make any explicit hypothesis on how the shopping journeys of customers change in a new store layout, except that the total number of shopping visits to each node stays the same.

## 6.3 Mobility model

We focus on the doubly-constrained gravity model with power-law deterrence function as our mobility model to estimate changes in the OD matrix  $\mathbf{T}^{\text{model}}$  when changing a store’s layout, because this gravity model provides the best fit to the data among the models that we tested, both in terms of the CPC score and in the estimated number  $v_k$  of visits (see Figure 5.4c and Table 5.3). We assume that we can swap the locations of nodes (which corresponds to swapping the contents of their shelves), but that we cannot change the store network topology or edge distances in any other way. In particular, we do not consider adding or removing nodes or edges, as such changes are often costly or impractical. To ensure that similar items stay with one another in the same aisle, we add a further constraint (which we call the *aisle constraint*) that we can only swap an aisle (which consists of a set of nodes) with another aisle with the same number of nodes. However, we do allow the permutation of nodes within the same aisle. For comparison, we also report our results when we relax the aisle constraint, where we allow the permutation of individual nodes (except for the entrance and till node, which are fixed). We highlight the nodes that belong to an aisle in Figure 6.1a by their colour. Nodes of the same colour are in the same aisle, and grey nodes do not

belong in any aisle. The mobility model (i.e., the gravity model) takes three inputs to calculate the OD matrix  $\mathbf{T}^{\text{model}}$  of a new store layout: (1) the quantities  $O_k$  and  $D_k$  (calculated using the location-independent sales model described in Section 6.2); (2) the permuted store network  $G$ ; and (3) the value of model parameter  $\gamma$ , which we take to be the optimal model parameter found in Section 5.3.

## 6.4 Optimization algorithm

We use SA algorithms [53] to find permuted layouts of a store with smaller values of one of the three objective functions ( $\lambda_{\max}$ ,  $Q$ , or  $\lambda_{\text{total}}$ ). In our SA algorithms, we make a change to the store layout (i.e., swapping aisles or swapping nodes) at each step and then calculate the difference  $\Delta E$  in the objective-function value. If  $\Delta E < 0$  (i.e., the change reduces the objective function), we always accept the change. If  $\Delta E \geq 0$ , we accept this change with probability  $\exp(-\Delta E/T_C)$ , where  $T_C$  is the computational temperature, and otherwise we reject it. The initial computational temperature is set to 200 when minimizing  $\lambda_{\max}$ , 20 when minimizing  $Q$ , and 2000 when minimizing  $\lambda_{\text{total}}$ . We use a cooling schedule in which the temperature is reduced by 0.18% at each step with 5000 steps in total. Therefore, as the algorithm progresses, it is less likely to accept changes that increase the objective function.

We consider two different SA algorithms for each objective function. The first SA algorithm respects the aisle constraint and it swaps at each step two aisles, which we choose uniformly at random from all pairs of aisles with the same number of nodes and whose centroids are less than 25 m apart. After swapping the two aisles, we permute the nodes within each aisle, where we choose the permutation uniformly at random from all possible permutations. Our second SA algorithm is for the case where we relax the aisle constraint. In this SA algorithm, instead of swapping aisles, we choose an edge uniformly at random from the set of edges that are not incident to the entrance and till nodes<sup>4</sup> and swap its incident nodes.

## 6.5 Results

We optimize a store’s layout (specifically, the layout of Store A) with the two SA algorithms (i.e., one that swaps aisles, and one that swaps nodes) for the three different objective functions  $\lambda_{\max}$ ,  $Q$ , and  $\lambda_{\text{total}}$ . The total mean queue size  $Q$  requires us to specify a service rate  $\mu$  and we choose  $\mu$  as follows. The maximum arrival rate  $\lambda_{\max}$

---

<sup>4</sup>We do this to ensure that we do not change the position of the entrance and till nodes.

in the original store layout of Store A is 6575, so we require  $\mu > 6575$ . We choose  $\mu = 7500$ , which appears to be a plausible service rate (in the sense that  $\mu$  has a sensible order of magnitude). In this case, each node serves incoming customers at a rate of 7500 customers per day, which amounts to 12.5 customers per minute in a store that is open for 10 hours. For example, in a store that is open for 10 hours, if  $\mu = 7500$ , the most popular node has a mean dwell time of about 38 seconds (which we calculate using Equation (4.61)). A mean dwell time of 38 seconds at the most popular node and a service rate of 12.5 customers per minute seem like plausible numbers for a store. We perform each optimization 20 times for each SA algorithm and each objective function and report our results in Table 6.1.

*Table 6.1. Minimum and mean values of objective functions of the final store layouts from 20 runs of the two SA algorithms (one with the aisle constraint and one without the aisle constraint) for optimizing Store A for all three objective functions. The first three rows show the minimum final value across 20 runs for each objective function. The last three rows show the mean final value across 20 runs for each objective function. We also show the relative reduction in brackets.*

	Obj. function	Original value	With aisle constraint	No aisle constraint
Min. value	$\lambda_{\max}$	6575	5009 (−23.8%)	5040 (−23.3%)
	$Q$ (with $\mu = 7500$ )	38.10	29.10 (−23.6%)	27.21 (−28.6%)
	$\lambda_{\text{total}}$	159745	151592 (−5.1%)	139957 (−12.4%)
Mean value	$\lambda_{\max}$	6575	5042 (−23.3%)	5150 (−21.7%)
	$Q$ (with $\mu = 7500$ )	38.10	29.28 (−23.1%)	28.02 (−26.5%)
	$\lambda_{\text{total}}$	159745	152279 (−4.7%)	141944 (−11.1%)

For all three objective functions, the two SA algorithms (i.e., an SA algorithm with aisle constraint and an SA algorithm without aisle constraint) produce store layouts with objective-function values that are significantly smaller than their values in the original store layout.

The SA algorithm with the aisle constraint achieves a relative reductions in  $\lambda_{\max}$  and  $Q$  of around 23%. When minimizing  $\lambda_{\text{total}}$ , the relative reduction in  $\lambda_{\text{total}}$  is significantly smaller and is around 5%. It is not surprising that the relative reduction in the objective function is smaller when minimizing  $\lambda_{\text{total}}$  compared to when we minimize  $\lambda_{\max}$  and  $Q$ , because nodes with the highest arrival rates in the original network give a much larger (relative) contribution to  $Q$  (with  $\mu = 7500$ ) and  $\lambda_{\max}$  than to  $\lambda_{\text{total}}$ . By definition, the maximum arrival rate depends only on the arrival rate of one node (the one with the highest arrival rate). When minimizing  $Q$  with  $\mu = 7500$ , we find that the sum of the mean queue sizes of the three nodes with the highest



arrival rates (i.e., the three most congested nodes) in the original store layout make up 39% of  $Q$ . By contrast, the sum of the arrival rates of these three nodes make up only 10% of  $\lambda_{\text{total}}$ . Therefore, store layouts in which these previously most congested nodes have significantly lower arrival rates (while the arrival rates of the other nodes do not increase much) tend to also have significantly lower values of  $\lambda_{\text{max}}$  and  $Q$  for  $\mu = 7500$  (compared to the original store layout); however, these store layouts do not necessarily have much lower values of  $\lambda_{\text{total}}$ . These most congested nodes contribute (in relative terms) less to  $\lambda_{\text{total}}$  than to  $\lambda_{\text{max}}$  or  $Q$  when  $\mu = 7500$ . Therefore, to achieve a relative reduction in  $\lambda_{\text{total}}$  of similar magnitude as for  $Q$  and  $\lambda_{\text{max}}$ , we need to reduce the arrival rates of a larger number of nodes than when minimizing  $Q$  or  $\lambda_{\text{max}}$ . However, this is potentially a very difficult task, so our observation of a lower reduction in  $\lambda_{\text{total}}$  than in  $Q$  and  $\lambda_{\text{max}}$  is consistent with expectations.

When we relax the aisle constraint, we allow many more store layouts, so we expect a larger reduction in the objective function without the aisle constraint than with it. To quantify how ‘restrictive’ the aisle constraint is for each objective function, we calculate the ratio  $R_{\text{AC}}/R_{\text{noAC}}$  between the relative reduction (across 20 runs for each SA algorithm) with the aisle constraint,  $R_{\text{AC}}$ , and the maximum relative reduction without the aisle constraint,  $R_{\text{noAC}}$ . A small value of  $R_{\text{AC}}/R_{\text{noAC}}$  means that there is a significantly larger reduction when we relax the aisle constraint than when we impose the aisle constraint, so the aisle constraint is more restrictive for a given objective function. We expect that  $R_{\text{AC}}/R_{\text{noAC}} \leq 1$  for all three objective functions. This is the case for  $Q$  and  $\lambda_{\text{total}}$ , where  $R_{\text{AC}}/R_{\text{noAC}}$  is equal to  $0.236/0.286 \approx 0.83$  when minimizing  $Q$  and to  $0.051/0.124 \approx 0.41$  when minimizing  $\lambda_{\text{total}}$ . The aisle constraint is more restrictive for  $\lambda_{\text{total}}$  than for  $Q$  — the relative reduction in the objective function is more than halved by imposing the aisle constraint. When minimizing  $Q$  (with  $\mu = 7500$ ), the reduction with the aisle constraint is only 17% lower than the reduction without the aisle constraint. Interestingly, when minimizing  $\lambda_{\text{max}}$ , we find that  $R_{\text{AC}}/R_{\text{noAC}} = 23.8/23.3 \approx 1.02$ , a value that is slightly larger than 1. This suggests that the SA algorithm is not very efficient at exploring the state space of store layouts, as it is unable to find the best store layouts that we obtained when we ran the SA algorithm with the aisle constraint; this store layout has smaller values of  $\lambda_{\text{max}}$  than the best store layout that the SA algorithm without the aisle constraint found. Note that the SA algorithm is a heuristic algorithm, so it is not guaranteed to find a global minimum (and it likely does not, like in this case).

We now examine the store layouts with the lowest values of each objective function and each SA algorithm in more detail. We measure the popularity of node  $k$  by

$O_k + D_k$  (i.e., the sum of the numbers of trips that start and end at  $k$ ). For each node except the entrance and till nodes,  $O_k + D_k$  is equal to twice the number of shopping visits to that node. When we examine the layouts with the smallest values of  $\lambda_{\max}$  and  $Q$  that the SA optimization with aisle constraint found, popular nodes have been moved from the centre of Store A towards the left and top of the store (see Figures 6.1b and 6.1c). By contrast, in the store layout with the smallest  $\lambda_{\text{total}}$  value that we obtained, many popular nodes remain in the centre of Store A (see Figure 6.1d). However, our optimization moves some of them to the bottom-left part of the store, which previously was not a popular area. In the store layouts that we obtain from the SA algorithm without the aisle constraint, we find that many popular nodes in the centre of the store have been moved to the top-right corner of the store and to the upper and right perimeter of the store (see Figure 6.2). In the store layout with the smallest  $Q$  with  $\mu = 7500$ , some popular nodes have also been moved to the left part of the store, which was not a popular area in the original store layout.

In the original store layout, the nodes in the centre of the store have the highest arrival rates (see Figure 6.3a), so these nodes are the most congested (according to our notion of congestion) and give a large contribution to  $Q$  and  $\lambda_{\text{total}}$ . The node with the maximum arrival rate  $\lambda_{\max}$  is also in the centre. Moving popular nodes from the centre to the perimeter appears to significantly reduce the arrival rates of the nodes in the centre (see Figures 6.3b–6.3d), and it thereby gives reductions in  $\lambda_{\max}$ ,  $Q$ , and  $\lambda_{\text{total}}$ .

Our approach also allows us to examine the proportion  $\rho_{\text{traversal},k}$  of arrivals at each node  $k$  that correspond to traversal visits. For each node  $k$ , the number  $v_k$  of total visits (over a time period  $\tau$ ) consists of  $O_k + D_k$  visits for shopping<sup>5</sup> and  $v_k - O_k - D_k$  traversal visits. Therefore, the proportion of visits that are traversals is

$$\rho_{\text{traversal},k} = \frac{v_k - O_k - D_k}{v_k}. \quad (6.7)$$

The majority of the nodes in the center walkway and on the perimeter in both the original network and in the optimized networks (with the SA algorithm with the aisle constraint) receive more traversal visits than visits for shopping (see Figure 6.4). These zones connect popular shopping aisles with one another, so people mainly visit these zones to move to a different aisle.

---

<sup>5</sup>Note here the difference between visits for shopping and the previously defined shopping visits. In our congestion model, the origin and destination node in each OD trip count as a visit for shopping. Each such visit for shopping count as one visit towards  $v_k$ . The previously defined shopping visits (which is equal to  $O_k$  for  $k \neq 1, n$ ) is the empirical number of times that customers visited  $k$  to purchase some items. Each shopping visit counts as two visits for shopping; one visit for a trip ending at  $k$  and one visit for a trip starting at  $k$ .

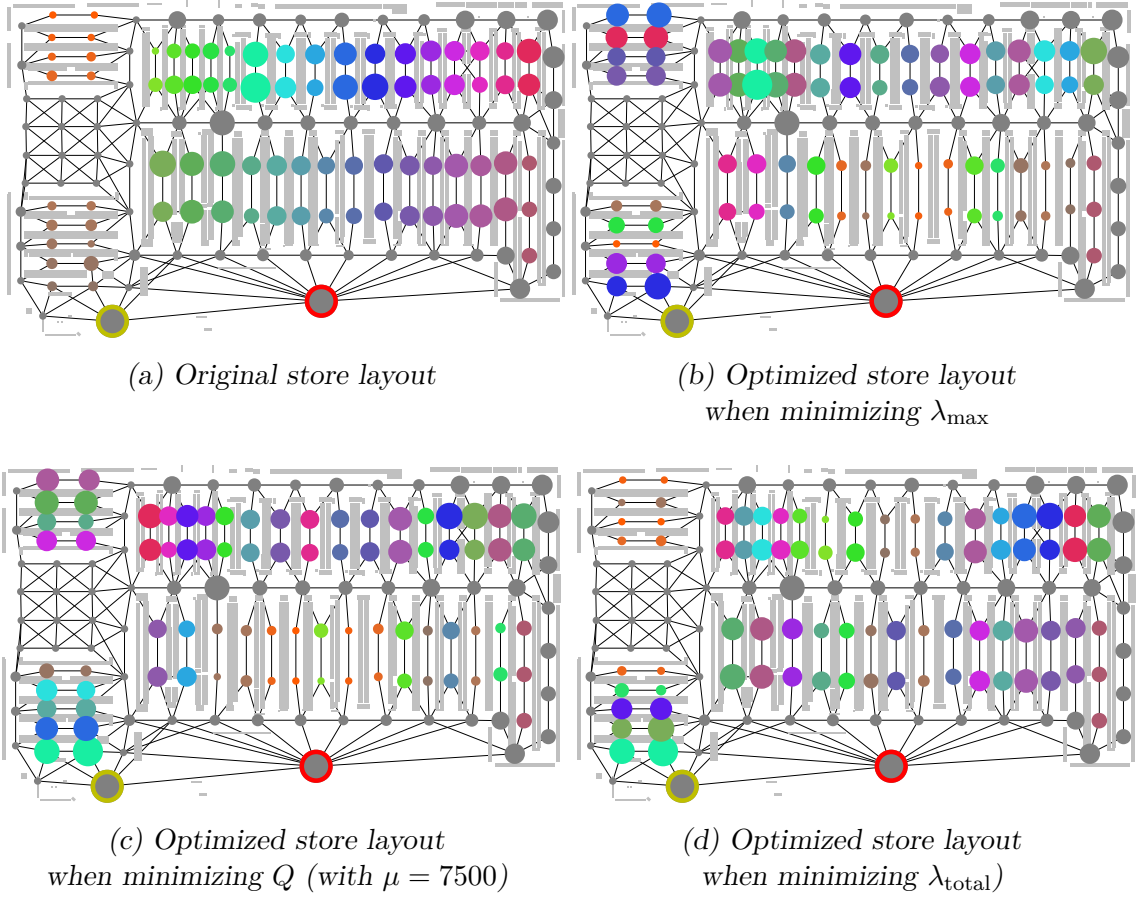


Figure 6.1. Location of popular nodes before and after optimization using an SA algorithm with the aisle constraint. Nodes of the same colour belong to the same aisle. Gray nodes do not belong to any aisles. The size of each node  $k$  is proportional to  $O_k + D_k$  (i.e., to the sum of the numbers of trips that start and end at zone  $k$ ). We circle the entrance and till nodes in yellow and red, respectively.

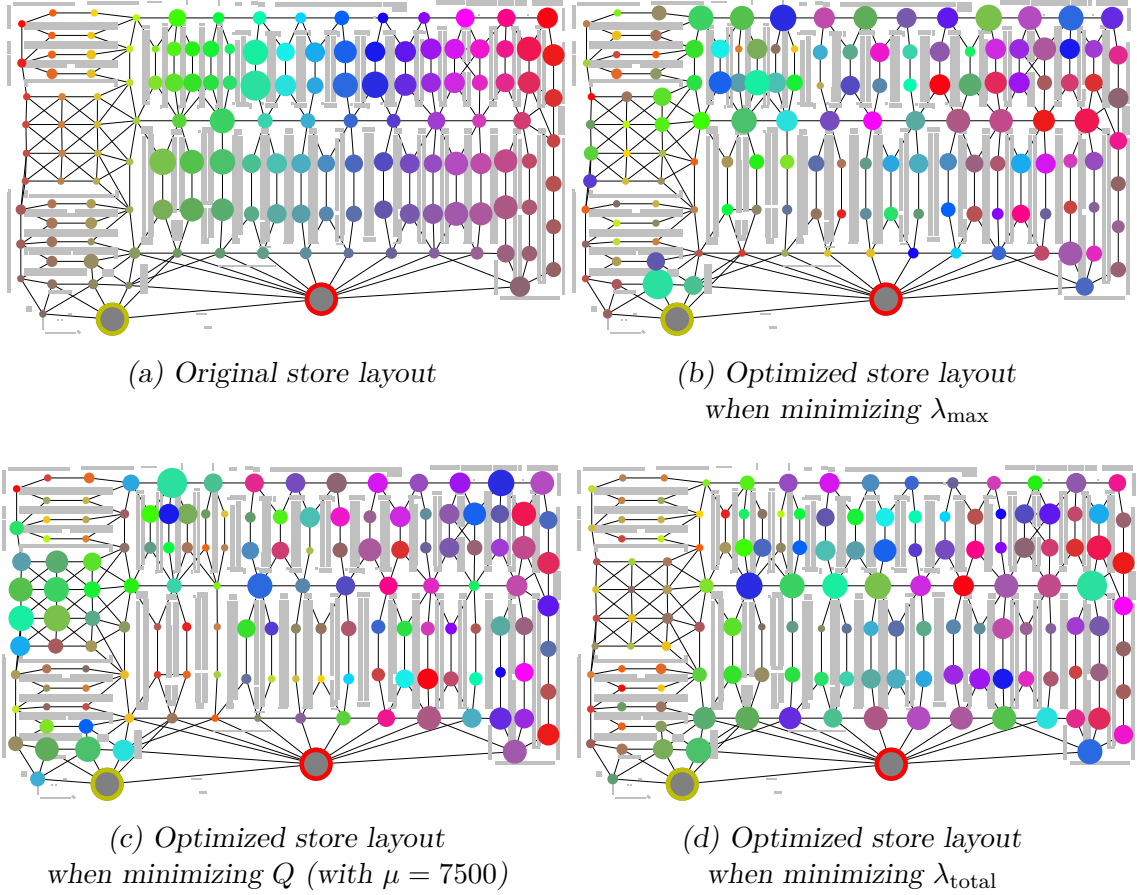


Figure 6.2. Location of popular nodes before and after optimization using an SA algorithm without the aisle constraint. We colour nodes by their position in the original store layout such that two nodes that are close together in the original store layout have similar colours. The size of each node  $k$  is proportional to  $O_k + D_k$  (i.e., to the sum of the numbers of trips that start and end at zone  $k$ ). We circle the entrance and till nodes in yellow and red, respectively.

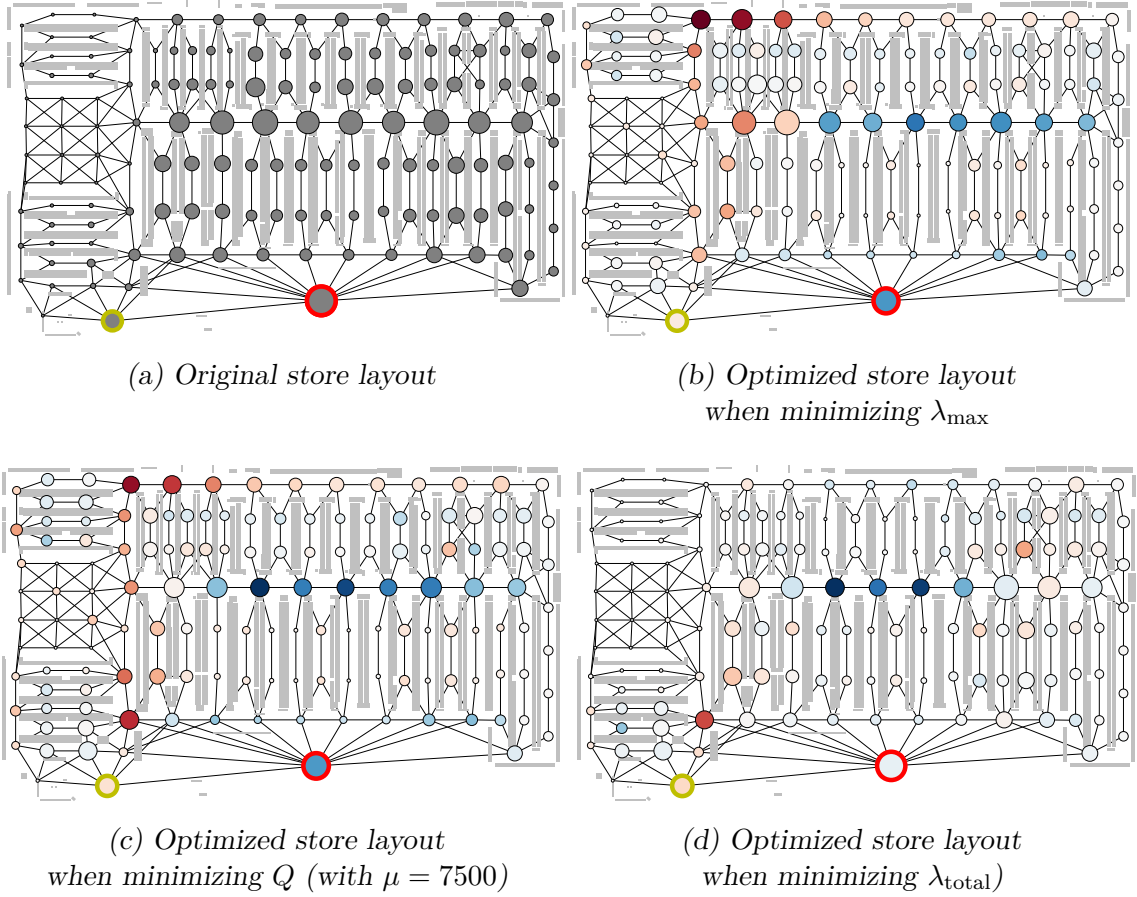


Figure 6.3. Arrival rates of nodes before and after optimization using an SA algorithm with the aisle constraint. The size of each node  $k$  is proportional to its arrival rate  $\lambda_k$ . We circle the entrance and till nodes in yellow and red, respectively. In panels (b)–(d), we colour nodes according to the difference  $\Delta\lambda_k$  between its arrival rate in the optimized store layout and the arrival rate in the original store layout. We colour a node blue if  $\Delta\lambda_k < 0$ ; these nodes have lower values of  $\lambda_k$  in the optimized store layout than in the original layout and therefore are less congested. We colour a node red if  $\Delta\lambda_k > 0$ . The darker the colour of a node, the larger the value of  $|\Delta\lambda_k|$  is. In the optimized store layouts, the nodes in the centre of the store, which have the highest arrival rates in the original layout, have significant lower values of  $\lambda_k$  than in the original layout.

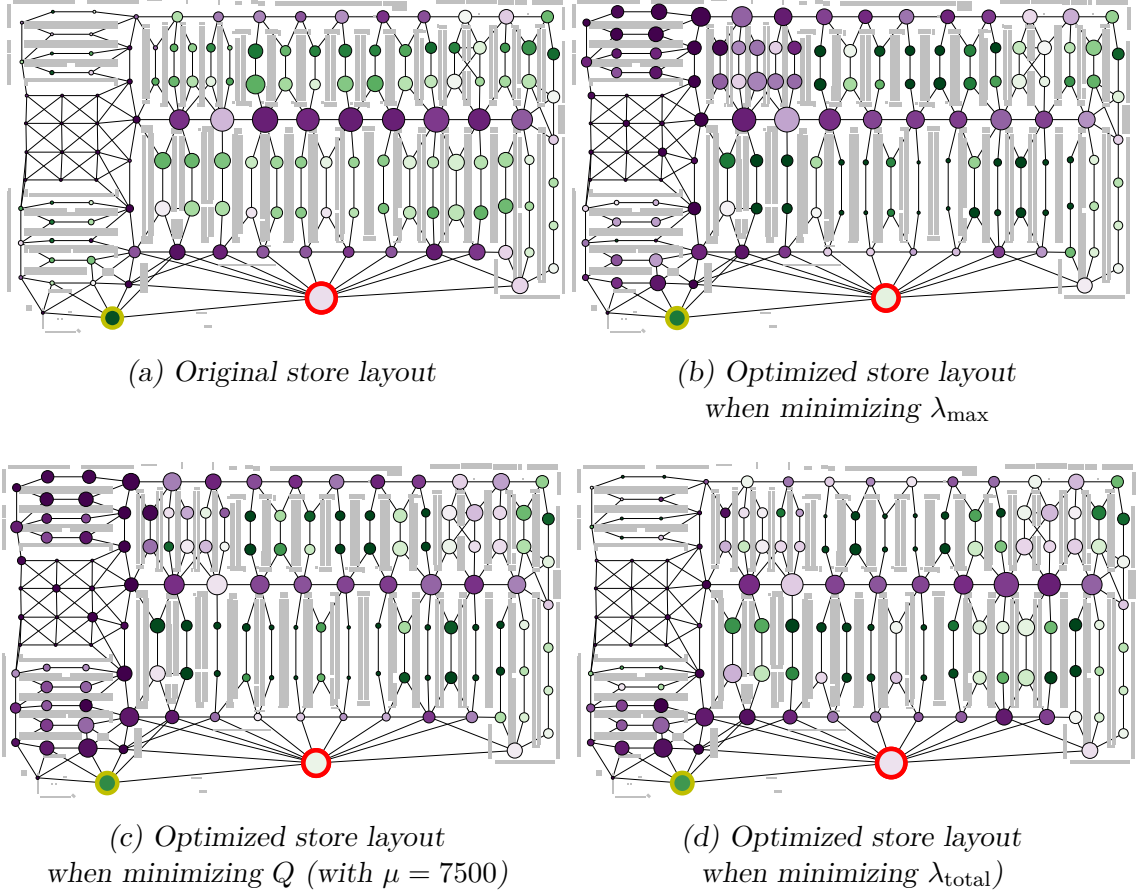


Figure 6.4. Proportion  $\rho_{\text{traversal},k}$  of arrivals that are traversal visits in the original store layout of Store A and in the optimized store layouts (optimized using an SA algorithm with the aisle constraint). The size of each node  $k$  is proportional to its arrival rate  $\lambda_k$ . We circle the entrance and till nodes in yellow and red, respectively. We colour nodes according to the value of  $\rho_{\text{traversal},k}$ . We colour a node  $k$  purple if  $\rho_{\text{traversal},k} > 0.5$ ; these nodes have more traversal visits than visits for shopping. We colour a node  $k$  green if  $\rho_{\text{traversal},k} < 0.5$ ; these nodes have less traversal visits than visits for shopping. The darker the colour of a node, the further away the value is from 0.5.

## 6.6 Summary and Discussion

In summary, we proposed a framework for modelling and reducing congestion in supermarkets. The framework consists of four components. The first component is a sales model that estimates the number of shopping visits (i.e., number of times that customers visit  $k$  to purchase items) when we change the location of a node. We assume, for simplicity, that the number of shopping visits of a node  $k$  does not depend on the location of  $k$ ; we call this model the location-independent sales model. The second component is a mobility model that takes a store layout and the number of shopping visits to each node  $k$  as an input and estimates the flow in this store layout. We chose the doubly-constrained gravity model (with power-law deterrence function), as it gave the best fit to the mobility flow data in Chapter 5. The third component is a congestion model that estimates congestion from the (estimated) mobility flow  $\mathbf{T}$  of a store layout. For this, we used a multi-class queueing network model with single-server queues that have homogeneous service rate  $\mu$  and assumed that there is a constant flow of customers who traverse a shortest path between any two nodes  $i$  to  $j$ . Each customer queues at every node on their shortest path. The flow rate from  $i$  to  $j$  is proportional to  $T_{ij}$ , and we measured congestion by the total mean queue size  $Q$ . The fourth component is an optimization algorithm that finds store layouts with less congestion. We used an SA algorithm for this and explored two scenarios. In the first scenario, aisles can be permuted (but individual store zones cannot be permuted, except within the same aisle). In the second scenario, we relaxed the aisle constraint, so all nodes except the entrance and till node can be permuted. For both settings, we used the gravity model to estimate how mobility flow changes from permutations of the store layout. We summarize our modelling framework in Figure 6.5.

We applied SA algorithms to identify store layouts with lower  $Q$  with  $\mu = 7500$ , a service rate that we deemed to be plausible. Furthermore, we used SA algorithms where we set the objective function to  $\lambda_{\max}$  and  $\lambda_{\text{total}}$ ; these two scenarios are proxies for minimizing  $Q$  in the small- $\mu$  regime and in the large- $\mu$  regime (see Section 4.3.2). Our SA algorithm identified store layouts with significantly lower values of  $Q$ ,  $\lambda_{\max}$ , and  $\lambda_{\text{total}}$  than the original store layout for both settings. In these layouts, many popular nodes (as measured by the number of trips from and to the node) move from the centre of a store to the perimeter of the store. Such a move alleviates an area of major congestion in the centre of the store.

Our framework does not rely on the specific choices of the four components. We can use any sales model as our first component, any population-level mobility model

as our second component, any congestion model that estimates congestion from the mobility flow as our third component, and any optimization algorithm that can explore the state space of store layouts as our fourth component. To improve our estimates of congestion and to identify better store layouts, one can replace our sales, mobility, or congestion models with more accurate models or replace our optimization algorithm with a better algorithm. In our location-independent sales model, the number of customer shopping visits at node  $k$  is assumed to be fixed and does not depend on the location of  $k$ . However, the location of a zone that includes items that are typically bought in an unplanned way likely affects the number of shopping visits to that zone. One can incorporate a more accurate sales model in our framework to improve estimates of the mobility flow from different store layouts and therefore the estimate of congestion. In our congestion model, we assumed shortest-path routing, but some researchers reported that customers deviate from shortest paths between purchases [42] (see Section 2.5.2). It is therefore important to improve understanding of the routes that customers take between purchases. One possible approach is to use anonymized customer trajectory data to develop and calibrate a stochastic routing model (e.g., using a variant of a random walk, perhaps with probabilities affected by heterogeneous fitness values for different zones of a store). One can incorporate such a routing model into our framework to better estimate the number  $v_k$  of visits to each zone  $k$ . Additionally, more empirical research is necessary to gain a detailed mechanistic understanding of the causes of congestion in supermarkets. We modelled congestion as queues in a zone and we can incorporate more realistic types of queues (e.g., with variable service rates or with customers who do not enter a queue if it is too long). One can infer service rates using a method that is analogous to what we described in Section 6.1, provided one possesses data on customer dwell time (or can somehow infer such times) for each zone of a store. In our current model, nodes equally serve customers who traverse it and customers who shop at it. It would be interesting to extend our analysis to queues which have two different service rates for these two types of customers. Furthermore, by using a better optimization algorithm, it should be possible to identify store layouts with even less (estimated) congestion than the ones that we found. On a theoretical level, it would be interesting to identify features of optimal queueing networks (i.e., those that minimize  $Q$ ) in which the mobility flow is characterized by a mobility model (e.g., a gravity model). It is also possible to include more constraints in our optimization, e.g., restricting to store layouts in which aisles that contain frozen food stay close to each other.



Our modelling framework is not restricted to supermarkets or small spatial scales. It can be applied to any system in which one seeks to reduce congestion that can be estimated from the mobility flow of people (as pedestrians or in cars). We expect that our methodology is implementable in practice in systems in which one can modify the underlying spatial structure. Many such applications have small spatial scales, as rewiring a small system is often a lot less costly than rewiring a large one. For example, when considering commuting flow, we cannot change the locations of countries or buildings. However, we can apply our tools for modelling mobility flow and congestion in a museum and use our optimization procedure to suggest better locations for the exhibits. Other examples include poster sessions in academic conferences, food stations in buffet restaurants, and stalls at markets and exhibitions. Applying our approach to these settings (and perhaps other settings on larger spatial scales) will help reveal which of our findings are specific to mobility flow in supermarkets and which ones apply more generally to human mobility on small spatial scales (or on any spatial scale).

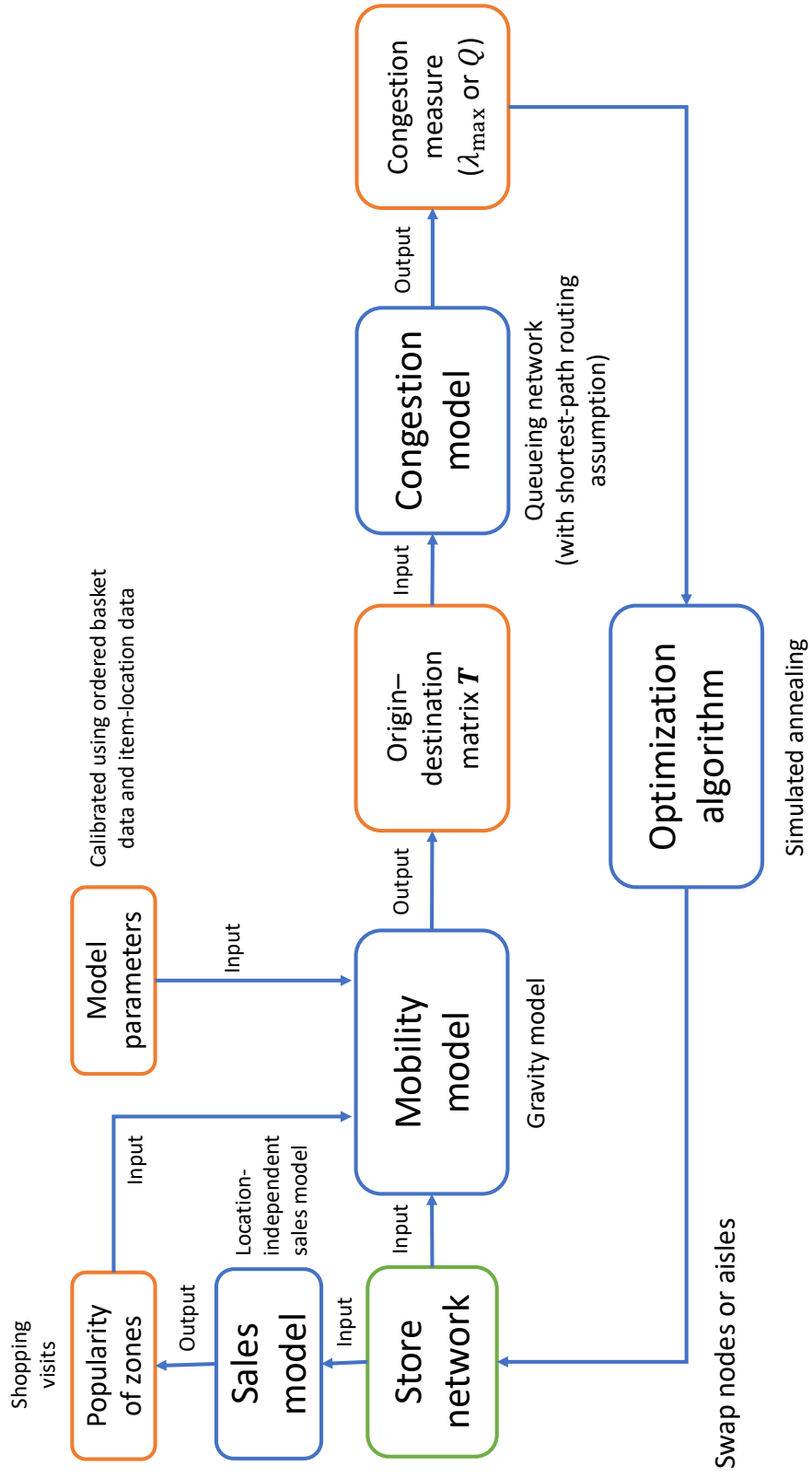


Figure 6.5. Schematic of our modelling framework.

# Chapter 7

## Conclusions and future work

In this thesis, we considered the problem of modelling and analyzing customer mobility and congestion in supermarkets. We were motivated by the problem of modelling how customers move in a supermarket and finding an optimal supermarket layout that minimizes congestion. We developed not only models of customer movement but also models that can estimate congestion based on mobility patterns in new store layouts. We used these models to find store layouts with less congestion. Previous models for customer mobility have provided no evidence that they fit to empirical data [41, 106], have yielded poor fits to empirical data [27], or are prohibitively expensive to simulate [14].

We represented a store as a spatial network (which we called a “store network”) in which the nodes are zones of the store and edges connect adjacent zones. We describe how we constructed the store networks in Chapter 3. We assumed that each zone acts as a queue, which is an abstract representation for congestion; and we measured congestion by the total mean queue size  $Q$ , a popular measure for congestion in queueing networks. Chapters 4–6 are the research chapters of this thesis. Broadly speaking, our main contributions in these chapters are (1) the development and analysis of models for customer mobility and congestion and (2) the analysis of how an underlying store layout affects both mobility and congestion in supermarkets. We summarize our specific contributions in the following paragraphs; see also the summary sections in Chapters 4–6.

In Chapter 4, we considered queueing networks with a single source (representing the entrance) and a single sink (representing the tills) and examined how network topology affects congestion (as measured by the total mean queue size  $Q$ ). We assumed that customers are unbiased random walkers and considered the system at stationarity. We identified network topologies that minimize  $Q$  given certain conditions and provided numerical evidence for the optimality of networks under more general conditions. We

also developed greedy algorithms that add and delete edges from a network to reduce  $Q$ . These algorithms are able to significantly reduce  $Q$  when applied to both random networks (from various ensembles) and to a store network (i.e., a network that we constructed from a real supermarket). In the context of supermarkets, the greedy algorithms do not obtain useful store networks (i.e., store layouts), as in these store networks, customers are directed as quickly as possible to the tills. Although these layouts have low congestion, customers do not have the chance to visit many zones to purchase items. Our random-walk model ignores customers' shopping intentions, which we conclude are an essential feature to include in a mobility model.

In Chapter 5, we used population-level human-mobility models to fit and estimate the mobility flow of customers between zones in supermarkets. We measured the empirical mobility flow from anonymized customer-basket data, where the flow of customers from an origin zone  $i$  to an destination zone  $j$  is approximately equal to the number of times that customers purchased items in zone  $i$  immediately followed by a purchase in zone  $j$ . We therefore considered shopping intentions in these models to overcome the aforementioned shortcoming of the unbiased random-walk model from Chapter 4. We found that some of the mobility models (including a gravity model) gave good fits to the empirical flow data in all 17 Tesco stores that we considered. The models estimate mobility flows using the popularity (estimated from sales data) of the nodes, the layout of the store (i.e., the store network), and a model parameter (that we calibrated using the empirical mobility flow). On average, the models can account for 65–70% of the empirical mobility flow. Therefore, we successfully applied population-level mobility models on small spatial scales. To the best of our knowledge, this is the first successful application of such models on these spatial scales. Interestingly, the model parameters do not change markedly between different stores and different time periods of the same store. This suggests that our models can be used to estimate mobility flow in stores using only sales data and store network. From mobility flow, one can estimate the flow of customers (i.e., foot traffic) through any zone, which then can be used to estimate congestion (by combining our mobility model with a congestion model, as we discussed in Chapter 6) or to estimate the number of customers who pass a shelf without purchasing anything from it. We also explored an extension to these models that gives a bias to flows between zones in which items are bought together frequently; however, these models did not provide a better fit to the data.

In Chapter 6, we proposed a framework for modelling and reducing congestion in supermarkets. In this framework, we integrated mobility models with queueing networks to estimate congestion in a supermarket by assuming that every node is a

queue with a customer-arrival rate estimated by a mobility model. We introduced a simulated-annealing (SA) algorithm that swaps aisles to find store layouts with less congestion. Our SA algorithm identified store layouts with significantly smaller values of  $Q$ . In these layouts, many popular nodes (i.e., nodes that are visited frequently for item purchases) that were originally in the centre of the store are now in the perimeter. In these rearrangements, congestion in the centre of the store (which is a major congestion point in the original layout) is significantly reduced.

Our models and our results from applying them to supermarket data open several new directions for future study. From a technical standpoint, several of these directions are concerned with validating or overcoming the assumptions that we have made throughout the thesis (e.g., stationarity). In particular, several components of the framework in Figure 6.5 can be improved by capturing more realistic features of customer mobility or congestion. We discuss some of these directions below. As we also discuss below, our work also opens new directions in the study of mobility, especially on small spatial and temporal scales.

In Chapter 6, we used a location-independent sales model that assumes that the number of shopping visits to a zone do not change with the location. This is a simplifying assumption that does not hold for many zones in real supermarkets (especially in large supermarkets, in which customers typically only visit a small proportion of the store [93]). An important direction is to develop a model that can accurately forecast how the number of shopping visits to a zone depends on the location in the store.

A further component of the framework in Figure 6.5 is our model of how customers travel between purchases. We assumed shortest-path routing, but a more realistic model is necessary to accurately ascertain how customers traverse a store between purchases. A key challenge is to identify which purchases are planned and which are unplanned, as customers are likely to traverse stores differently in these two situations. Furthermore, we assumed that the route and purchases that a customer takes does not depend on congestion. This too has not been verified empirically, and more research is needed to assess whether this assumption stands up to empirical scrutiny. (We expect that some customers change their route or even drop items from their shopping lists in response to congestion.)

To pursue the above directions, it would be useful for retailers to collect data sets of customers with their planned purchases (i.e., shopping lists), their journeys inside stores, and their actual purchases during a store visit. These data sets can be used to investigate how customers move when they buy a planned item versus when they buy

an unplanned item (and how their routes change with congestion). Additionally, these data sets help researchers develop models for forecasting how the number of shopping visits at each node depends on the location in the store.

Although reducing congestion is one of the main motivations of this thesis, we used a model of congestion (a queueing network) that we have not verified to be accurate empirically. Queues are a useful, abstract model for congestion, but it is important to develop more accurate and empirically verified models for congestion. One can use location data of customers in a supermarket to assist in the development of better congestion models. One can measure the time that customers spend in a zone and record how this depends on the number of customers that are present. A key challenge will be to separate the delay in traversing a zone due to congestion from the delay due to purchasing or considering to purchase an item. Applying computer-vision techniques may be helpful to overcome this challenge. For example, Liang *et al.* [62] developed a model that estimates future paths and actions of pedestrians from camera data using convolutional neural networks. It may be possible to adapt such technology to infer the intention behind customer's action. For example, one may be able to infer whether a customer waits or changes their shopping path due to congestion or due to shopping intentions (e.g., going to the next item in their shopping list). If a queueing framework fits empirical data sufficiently well, one can extend our analysis from Chapter 4 to work with more realistic queues. Throughout our work, we assumed that our systems were at stationarity to allow the mathematical theory to be tractable. Congestion in supermarkets is very unlikely to be a stationary system, and there is therefore a need to extend queueing theory to non-stationary systems. We expect that this will be an interesting challenge for both mathematics and complex systems.

We used population-level mobility models, which do not offer any description of the trajectories that customers take. We only described how aggregate flow depends on a store network and the individual zone popularities (measured, in a rough way, using the number of times that customers bought items in a zone). It will be useful to develop individual-level models for customer mobility that capture the key features that underlie customers' individual mobility.

Our framework is not restricted to modelling customers in a supermarket only, but can be used in any application where one seeks to model the mobility of humans (or animals) on small spatial scales. Applying our ideas and approaches to mobility in places such as museums, fairs, or rabbit burrows will help reveal which findings are specific to supermarkets and which findings apply more generally to small spatial scales.

Advances in the above directions will allow retailers and researchers to gain insight into how customers move in supermarkets and help lead to more efficient and less congested stores. More generally, taking insights of our work and using them to examine other challenging problems in human (and animal) mobility will help further understanding of mobility of humans (and animals) on small spatial scales and the understanding of complex systems in general.

# Appendices



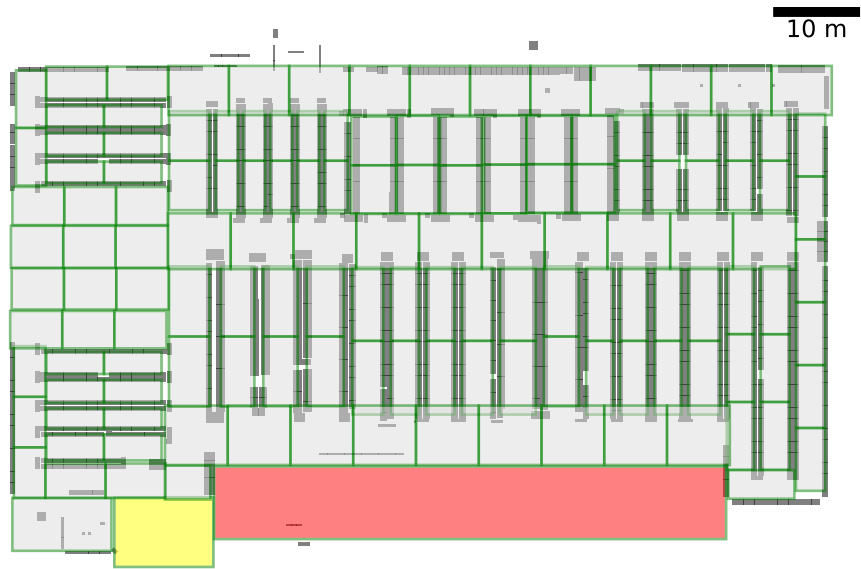
# Appendix A

## Store zoning, store networks, and store network properties

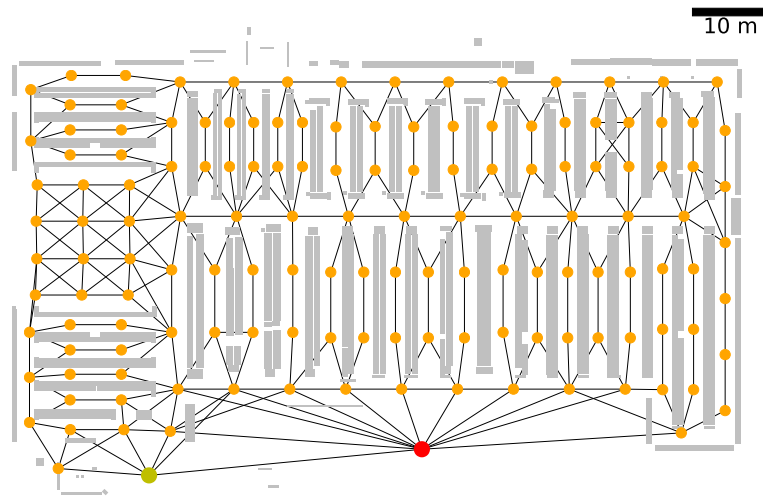
We show statistics of the 17 stores (Stores A–Q) in Table [A.1](#). We also show the store zoning and store networks for Stores A–Q in Figures [A.1–A.17](#).

Table A.1. Statistics of the store networks and store zoning of 17 stores (Stores A–Q).

Store ID	Num- ber of nodes	Num- ber of edges	Mean degree	Shortest path length from entrance to tills (in m)	Mean shortest path length (in m)	Shortest path length from entrance to tills (in number of traversed edges)	Mean shortest path length (in number of traversed edges)	Mean zone length (excl. entrance and tills)
Store A	144	272	3.78 m	35.96 m	49.31 m	1	5.81	6.99 m
Store B	162	299	3.69 m	45.39 m	54.37 m	2	6.37	6.90 m
Store C	129	240	3.72 m	31.72 m	46.61 m	1	5.73	6.60 m
Store D	89	192	4.31 m	37.00 m	36.66 m	1	4.45	7.42 m
Store E	73	136	3.73 m	46.18 m	33.81 m	5	4.52	6.42 m
Store F	179	384	4.29 m	43.68 m	55.23 m	2	6.46	6.86 m
Store G	77	169	4.39 m	25.02 m	35.46 m	1	4.13	7.65 m
Store H	159	265	3.33 m	30.38 m	48.79 m	1	6.28	6.43 m
Store I	154	261	3.39 m	59.98 m	55.94 m	5	6.71	7.29 m
Store J	73	133	3.64 m	26.05 m	39.30 m	1	4.42	6.95 m
Store K	61	128	4.20 m	18.24 m	31.89 m	1	3.76	6.91 m
Store L	110	191	3.47 m	34.16 m	45.23 m	3	5.89	6.59 m
Store M	133	256	3.85 m	57.18 m	47.80 m	6	6.13	7.43 m
Store N	89	163	3.66 m	32.30 m	33.65 m	5	5.12	6.42 m
Store O	146	271	3.71 m	33.25 m	45.77 m	2	6.26	6.49 m
Store P	138	255	3.70 m	43.56 m	50.25 m	3	6.34	7.27 m
Store Q	197	401	4.07 m	36.70 m	56.45 m	1	7.20	6.82 m
Mean	124.29	236.24	3.82 m	37.46 m	45.09 m	2.41	5.62	6.91 m

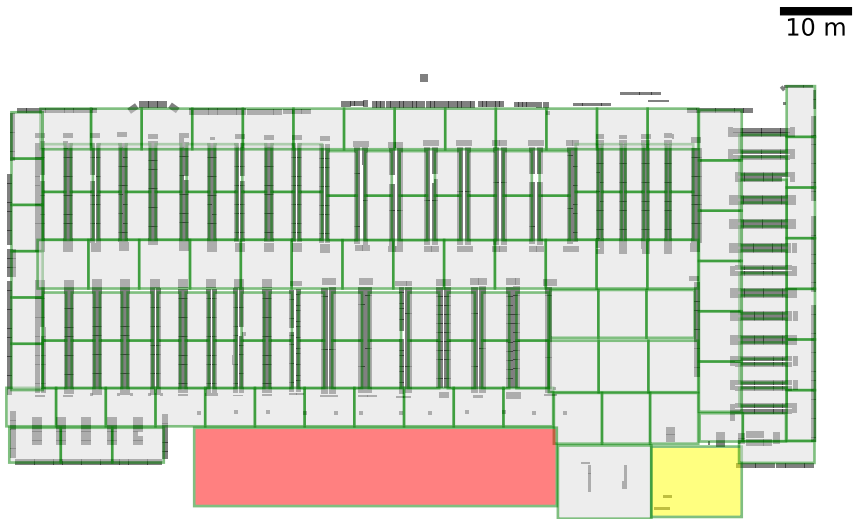


(a) Store zoning

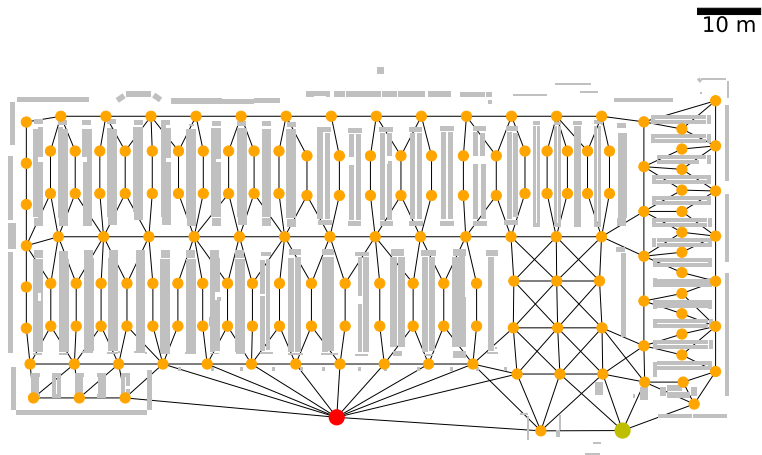


(b) Store network

Figure A.1. Store zoning and store network of Store A.

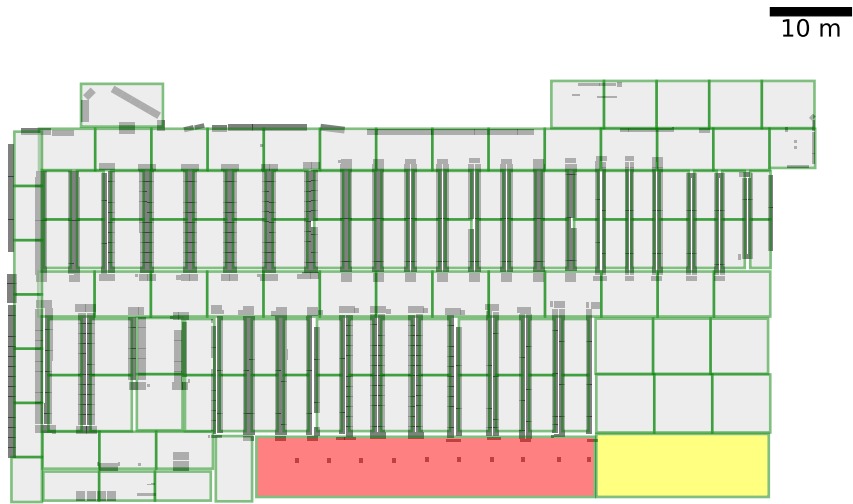


(a) Store zoning

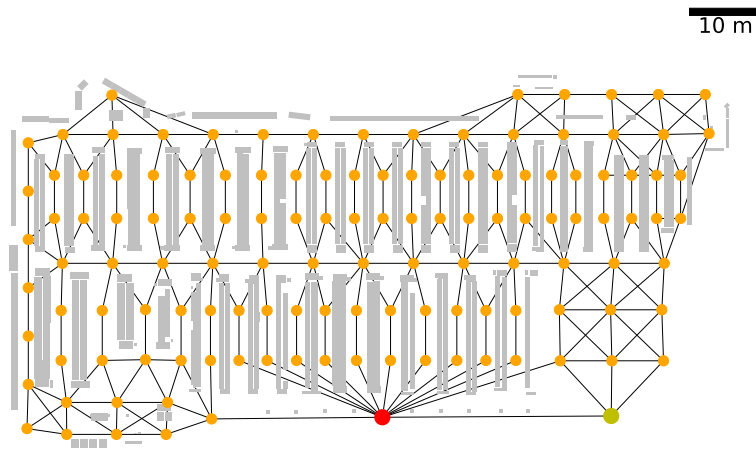


(b) Store network

Figure A.2. Store zoning and store network of Store B.

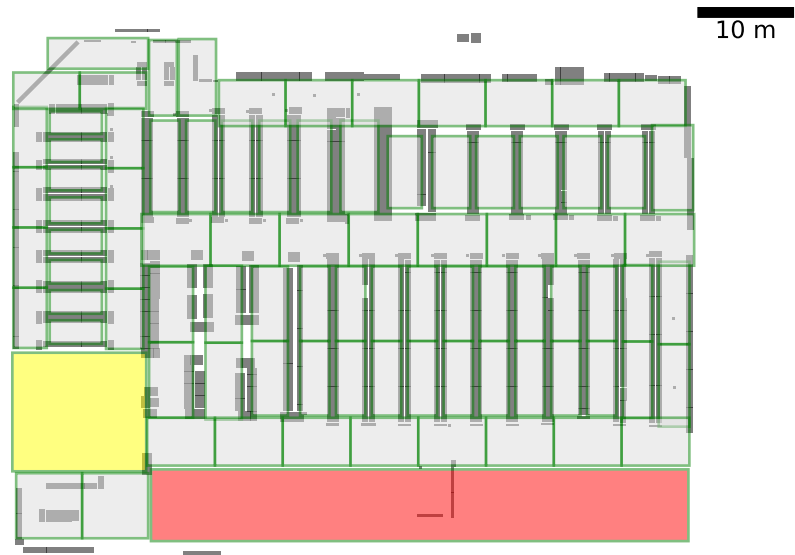


(a) Store zoning

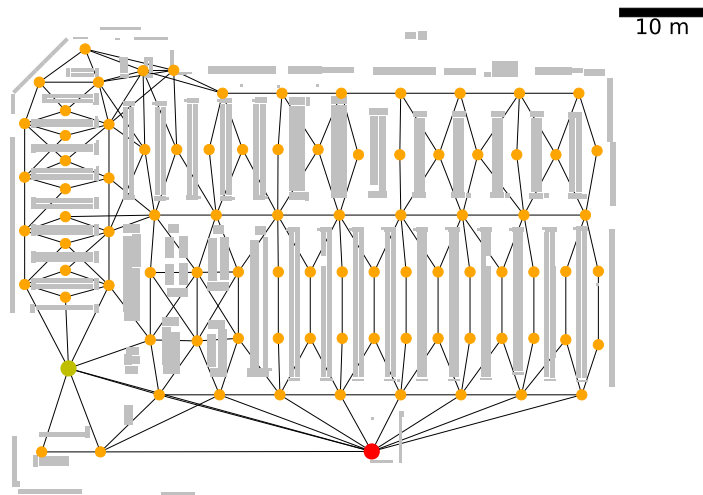


(b) Store network

Figure A.3. Store zoning and store network of Store C.

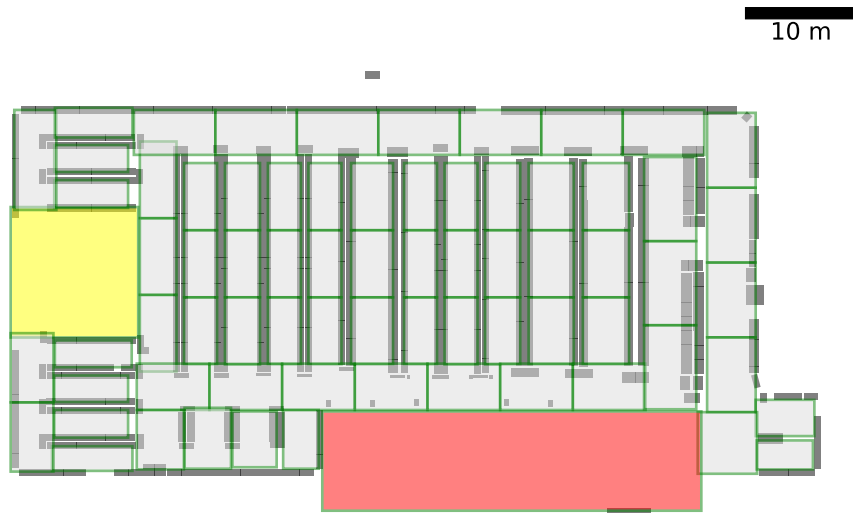


(a) Store zoning

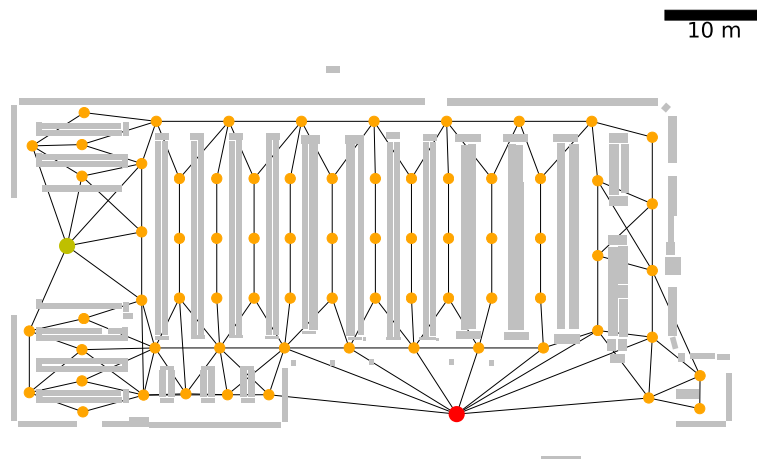


(b) Store network

Figure A.4. Store zoning and store network of Store D.

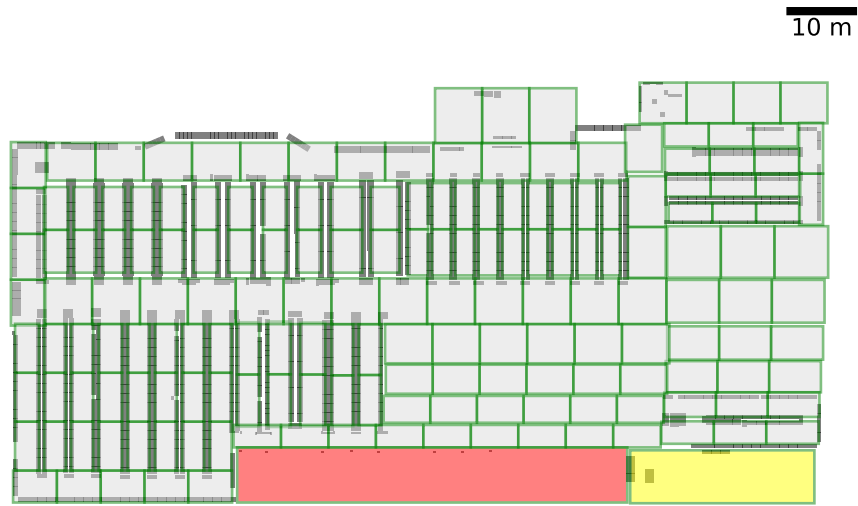


(a) Store zoning

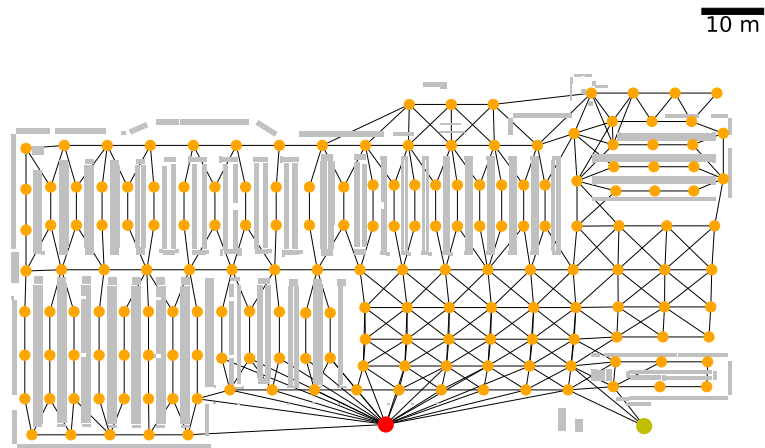


(b) Store network

Figure A.5. Store zoning and store network of Store E.



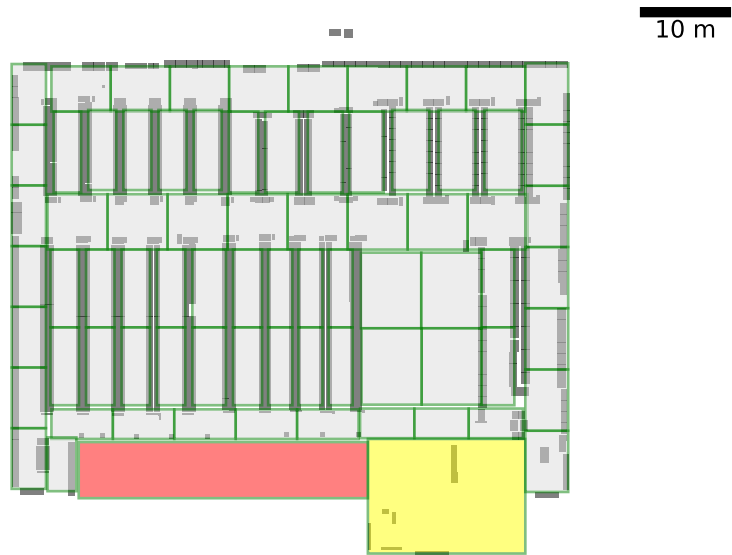
(a) Store zoning



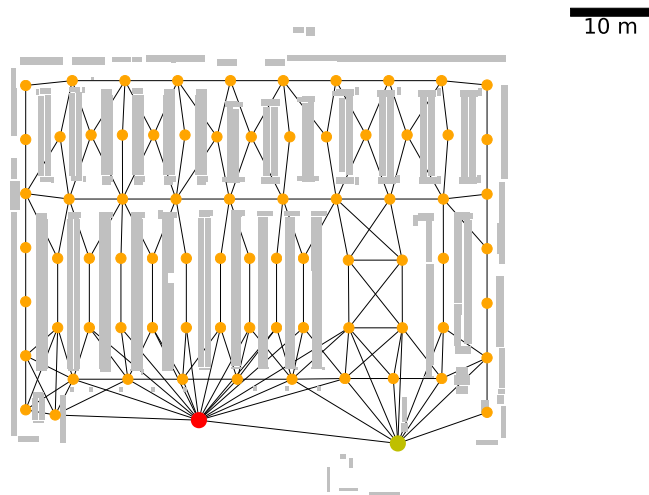
(b) Store network

Figure A.6. Store zoning and store network of Store F.



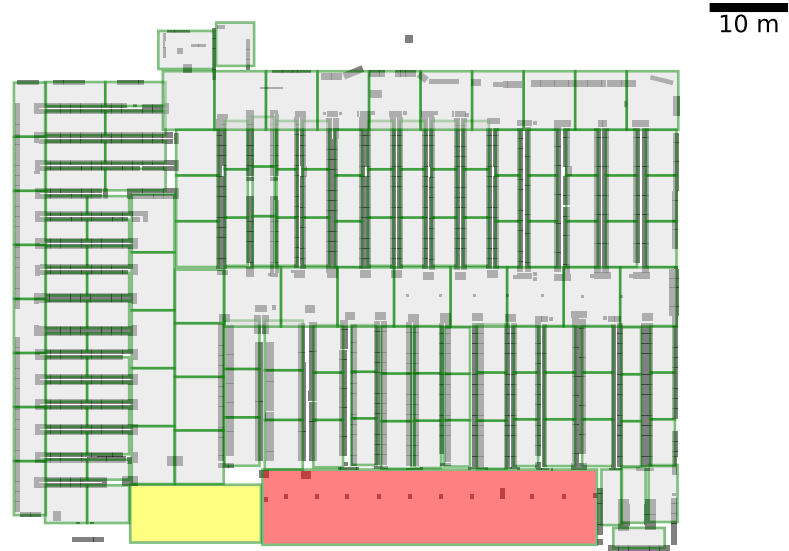


(a) Store zoning

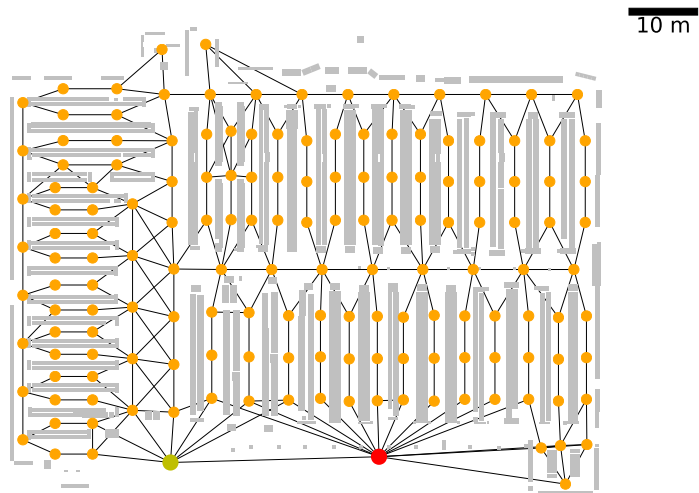


(b) Store network

Figure A.7. Store zoning and store network of Store G.

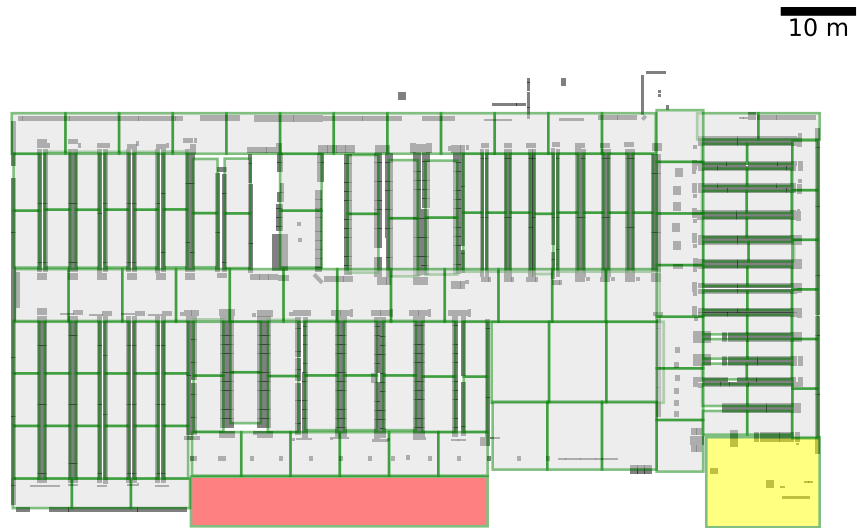


(a) Store zoning

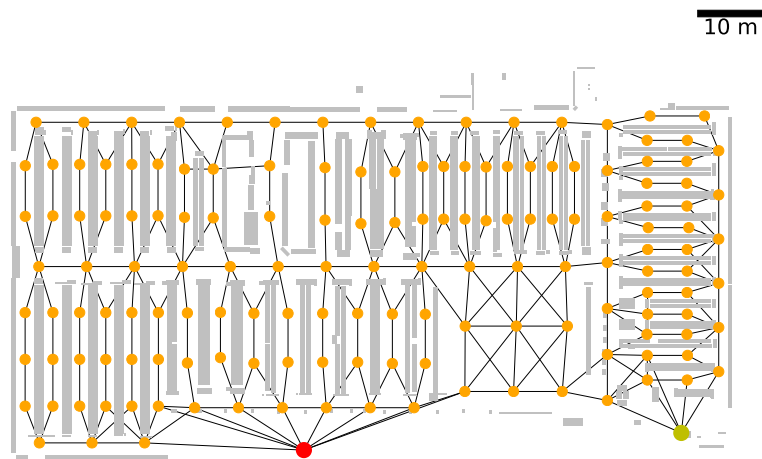


(b) Store network

Figure A.8. Store zoning and store network of Store H.

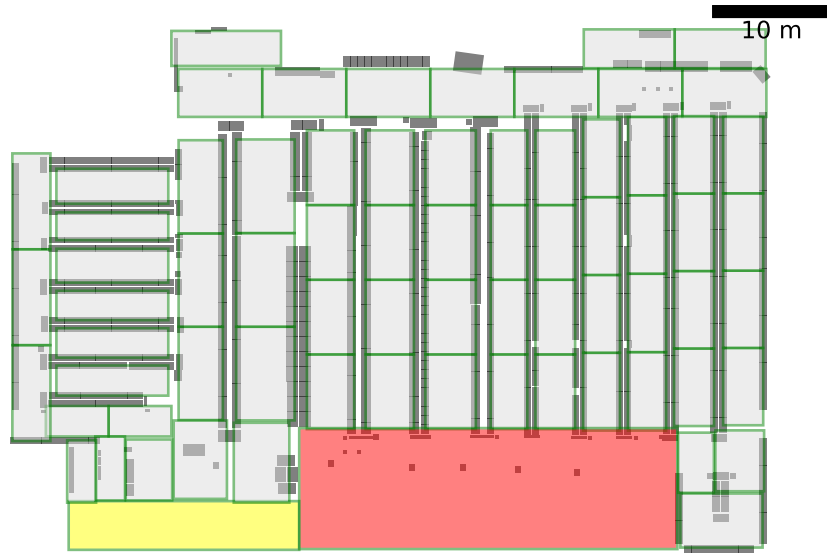


(a) Store zoning

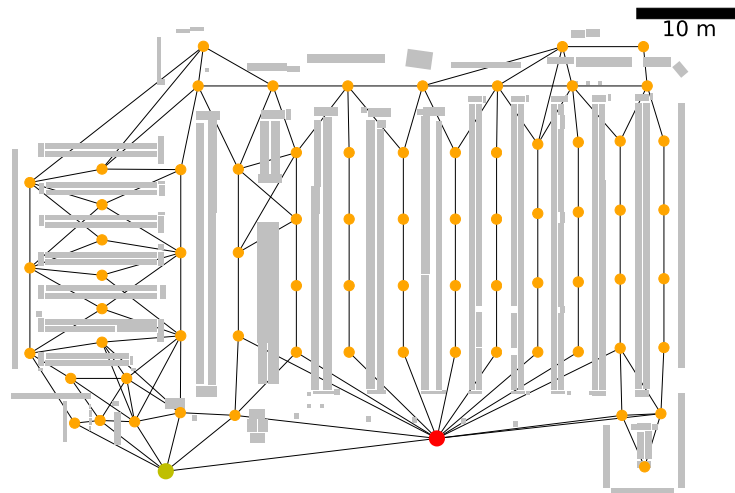


(b) Store network

Figure A.9. Store zoning and store network of Store I.

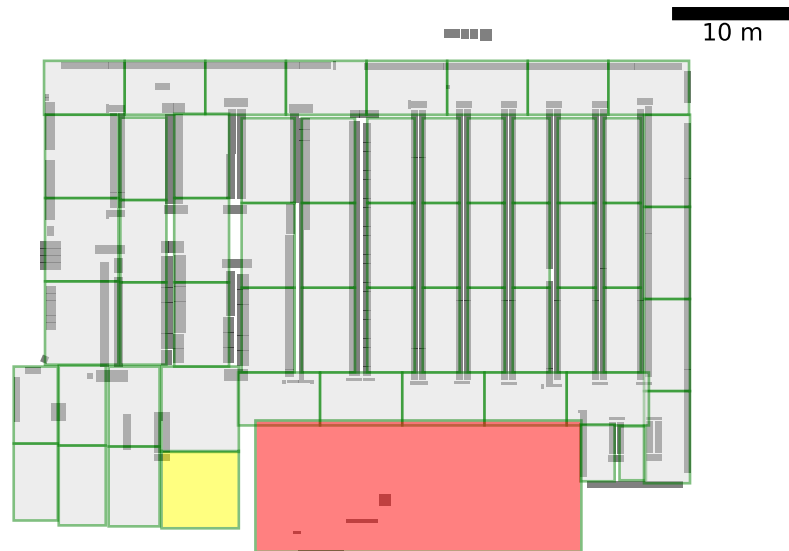


(a) Store zoning

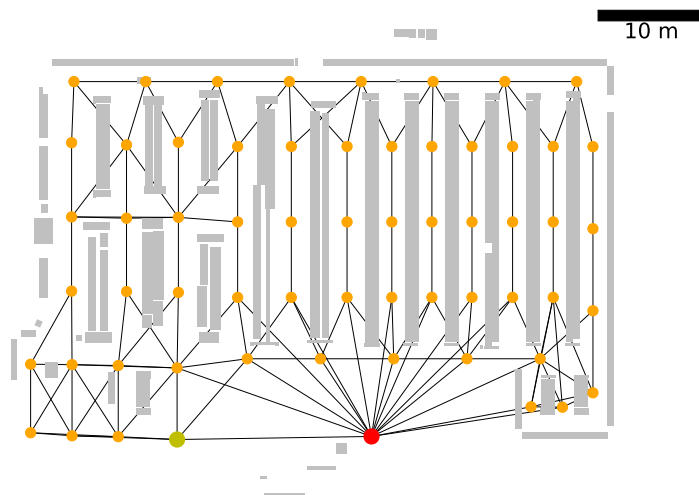


(b) Store network

Figure A.10. Store zoning and store network of Store J.



(a) Store zoning

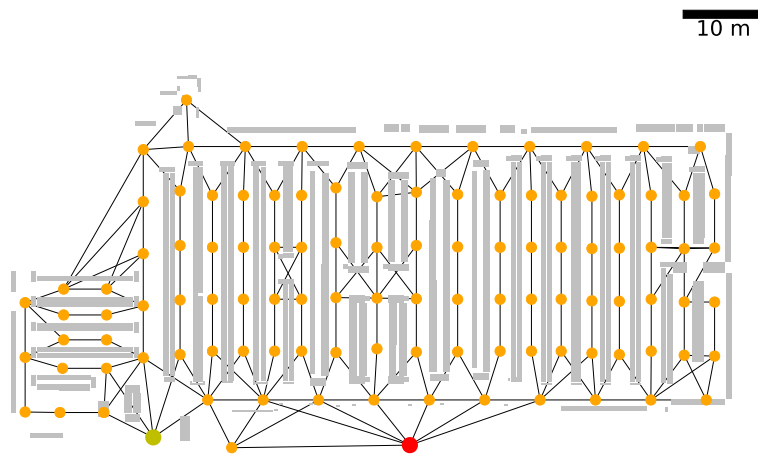


(b) Store network

Figure A.11. Store zoning and store network of Store K.

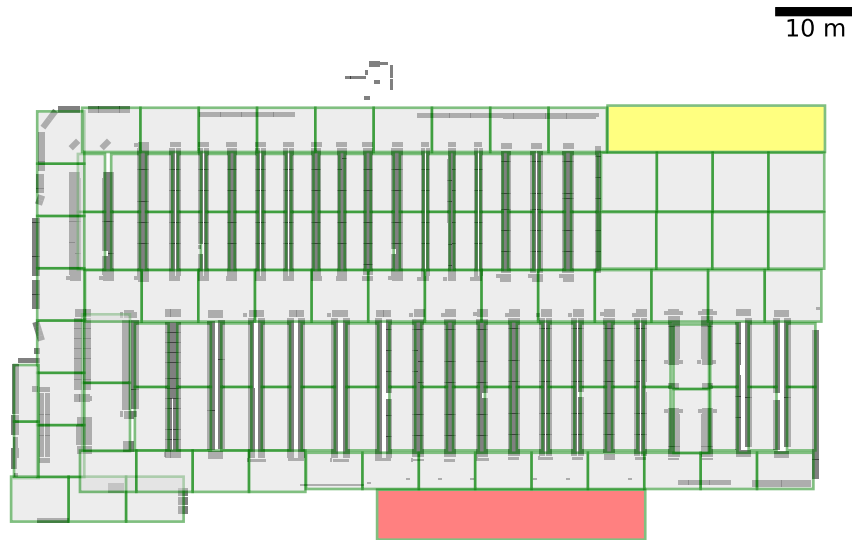


(a) Store zoning

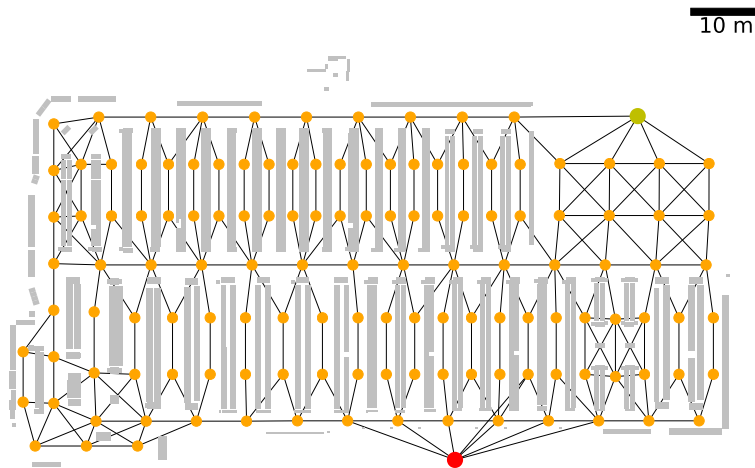


(b) Store network

Figure A.12. Store zoning and store network of Store L.

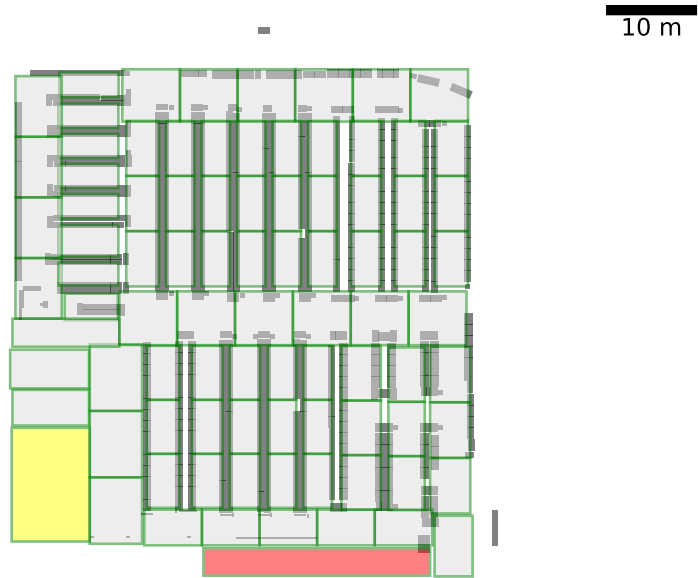


(a) Store zoning

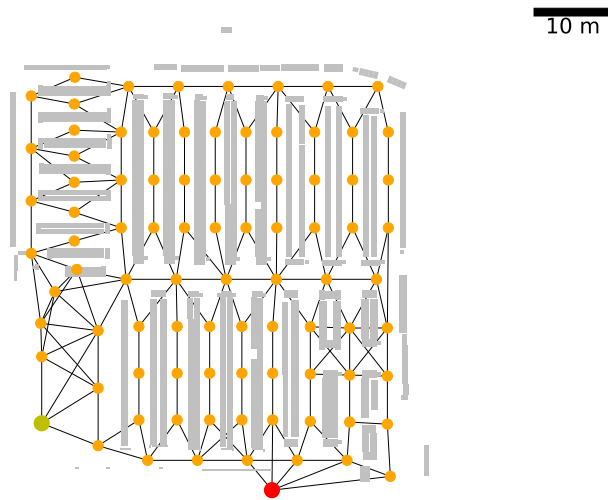


(b) Store network

Figure A.13. Store zoning and store network of Store M.



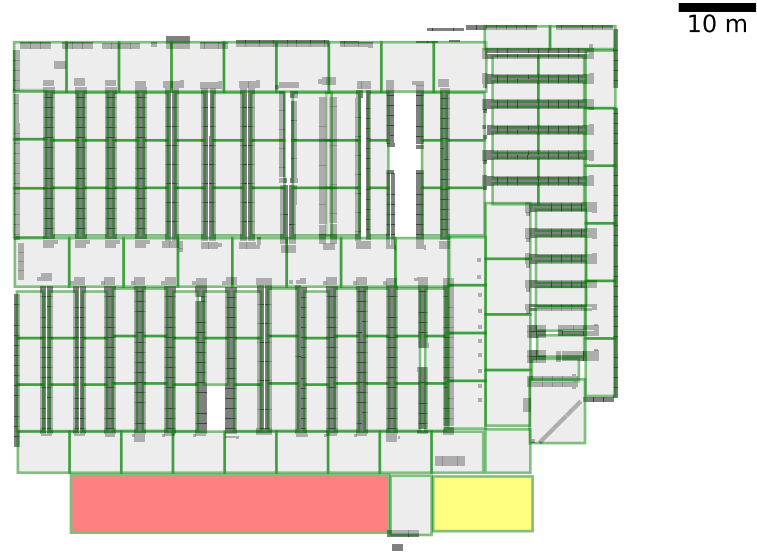
(a) Store zoning



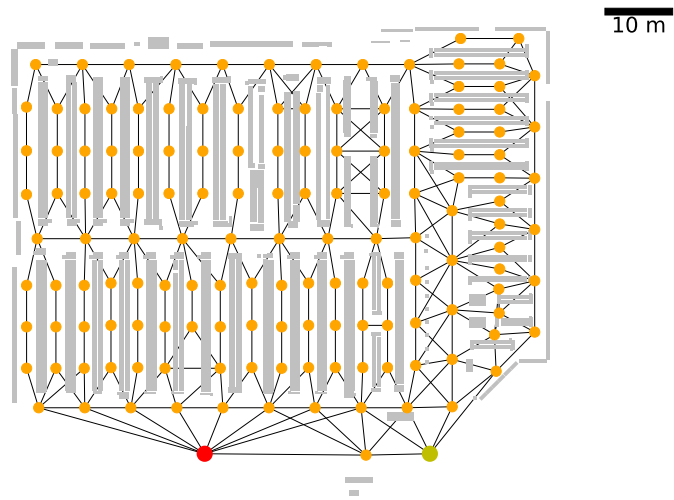
(b) Store network

Figure A.14. Store zoning and store network of Store N.



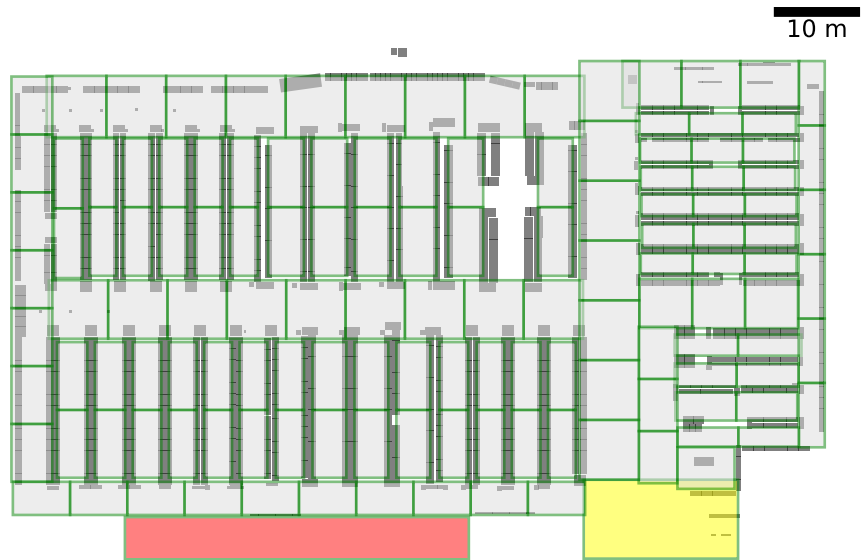


(a) Store zoning

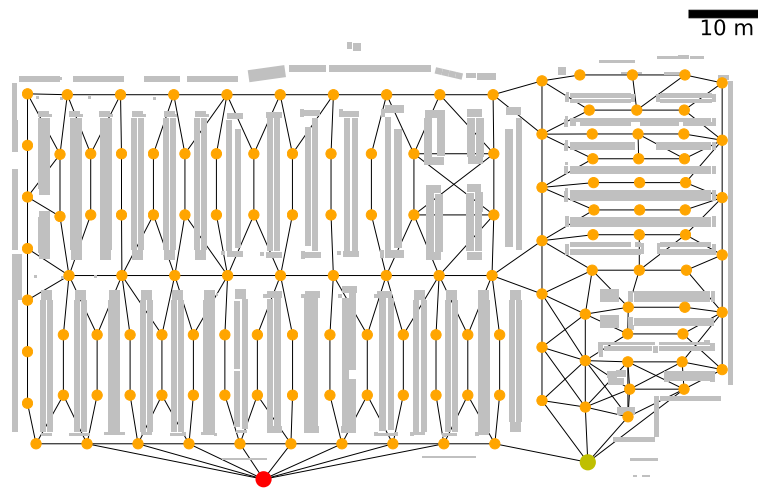


(b) Store network

Figure A.15. Store zoning and store network of Store O.

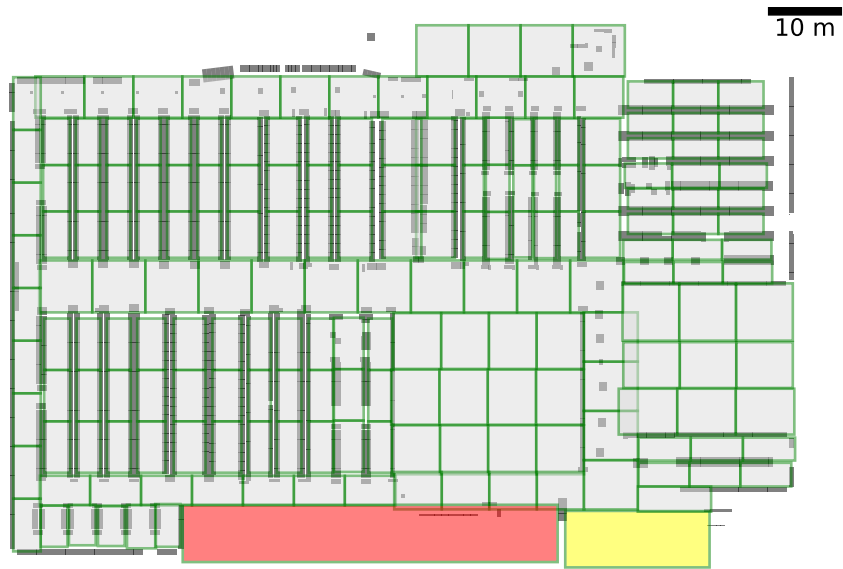


(a) Store zoning

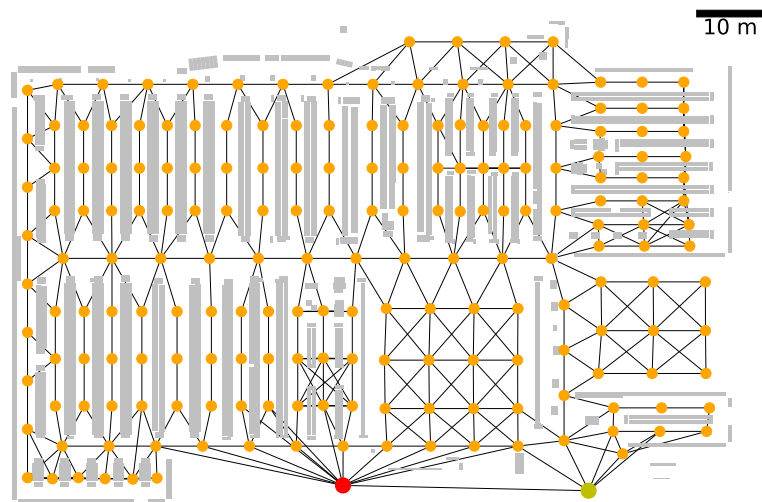


(b) Store network

Figure A.16. Store zoning and store network of Store P.



(a) Store zoning



(b) Store network

Figure A.17. Store zoning and store network of Store Q.

# Appendix B

## Appendix to Chapter 4

### B.1 Simulated-annealing algorithm for finding optimal network topologies

#### B.1.1 Description of simulated-annealing algorithm

We use a simulated-annealing (SA) algorithm to find networks (directed or undirected) with a minimum objective-function value. We use objective functions  $f(\lambda_1, \dots, \lambda_n)$  (specifically,  $\sum_{i=2}^{n-1} \lambda_i$ ,  $\lambda_{\max}$ , and  $\lambda_{\text{total}}$ ) that depend only on the arrival rates  $\lambda_i$  (with  $i = 1, \dots, n$ ) of nodes of a network  $G$ .

We initialize the SA algorithm with a network  $G$  that is a fully connected network, and we set the initial computational temperature to  $T_c = 1$ . At each iteration of the SA algorithm, we perform the following actions on  $G$ :

Step 1: Choose an ordered pair of nodes  $(i, j)$  of the current graph  $G = (V, E)$  uniformly at random from all possible pairs of nodes.

Step 2: Perform one of the following actions (depending whether  $(i, j) \in E$  or  $(i, j) \notin E$ ):

- (a) If  $(i, j) \notin E$ , we add the directed edge  $(i, j)$  to  $G$ . If  $G$  is undirected and  $j \neq n$ , we also add the reciprocal edge  $(j, i)$  to  $G$ .
- (b) If  $(i, j) \in E$ , we remove  $(i, j)$  from  $G$ . If  $G$  is undirected and  $j \neq n$ , we also remove the reciprocal edge  $(j, i)$  from  $G$ . If the resulting graph does not satisfy the reachability conditions (see Section 2.3.1), we undo the removal of the edge(s) and repeat Steps 1 and 2 until we select a pair of nodes  $(i, j)$  with either  $(i, j) \in E$  or the graph that we obtain from removing  $(i, j)$  from  $G$  satisfies the reachability conditions.

Step 3: Compute the change  $\Delta f$  in the objective function between the new graph and the old graph.

Step 4: If  $\Delta f < 0$  (i.e., the change reduces the value of the objective function), we accept this change. If  $\Delta f \geq 0$ , we accept this change with probability  $\exp(-\Delta f/T_c)$ , where  $T_c$  is the computational temperature.

Step 5: Reduce the computational temperature  $T_c$  by  $8.3 \times 10^{-6}T_c$ .

We run  $10^5$  iterations in total. As the algorithm progresses, the computational temperature decreases and the algorithm accepts progressively fewer changes that increase the objective function.

We use the SA algorithm to try and find the following optimal networks:

1. A directed network on  $n$  nodes with minimal  $\sum_{i=2}^{n-1} \lambda_i$ .
2. An undirected network on  $n$  nodes with minimal  $\lambda_{\max} = \max_{i=1}^n \lambda_i$ .
3. An undirected network on  $n$  nodes with minimal  $\lambda_{\text{total}} = \sum_{i=1}^n \lambda_i$ .

## B.1.2 Results

### Directed networks on $n$ nodes with minimal $\sum_{i=2}^{n-1} \lambda_i$

We conjecture that for any directed network  $G \in \mathcal{C}_n$  (with  $n \geq 3$ ), the arrival rates  $\lambda_i$  of the nodes of  $G$  satisfy  $\sum_{i=2}^{n-1} \lambda_i \geq 1 - 1/(n-1)$  (see Equation (4.52)). We have verified this statement for small  $n$  (specifically, for  $n = 3, \dots, 7$ ).

We use the SA algorithm to find networks of larger sizes with minimal  $\sum_{i=2}^{n-1} \lambda_i$ . For each  $n$ , we run the SA algorithm 20 times and record the value of  $\sum_{i=2}^{n-1} \lambda_i$  of the minimal network from each run. We find for  $n = 20, 50$ , and 100 that the SA algorithm fails to find any networks with  $\sum_{i=2}^{n-1} \lambda_i$  that are smaller than  $1 - 1/(n-1)$ , the conjectured minimum (see Figure B.1). We also find that the values of  $\sum_{i=2}^{n-1} \lambda_i$  are close to the conjectured minimum value.

### Undirected networks on $n$ nodes minimizing $\lambda_{\max}$ or $\lambda_{\text{total}}$

We conjecture that, for any network size  $n \geq 3$ , the network  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  (depicted in Figure 4.10) uniquely minimizes  $\lambda_{\text{total}}$  over  $\mathcal{U}_n$ . If the conjecture holds, then for any  $n \geq 5$ , there are no  $Q$ -optimal networks over the space  $\mathcal{U}_n$  of undirected networks that satisfy the reachability conditions (see Section 4.1.4). We further conjecture that for any network size  $n \geq 3$ , the  $G_{\mathcal{U}_n}^{\lambda_{\max}}$  (depicted in Figure 4.11) minimizes  $\lambda_{\max}$ .

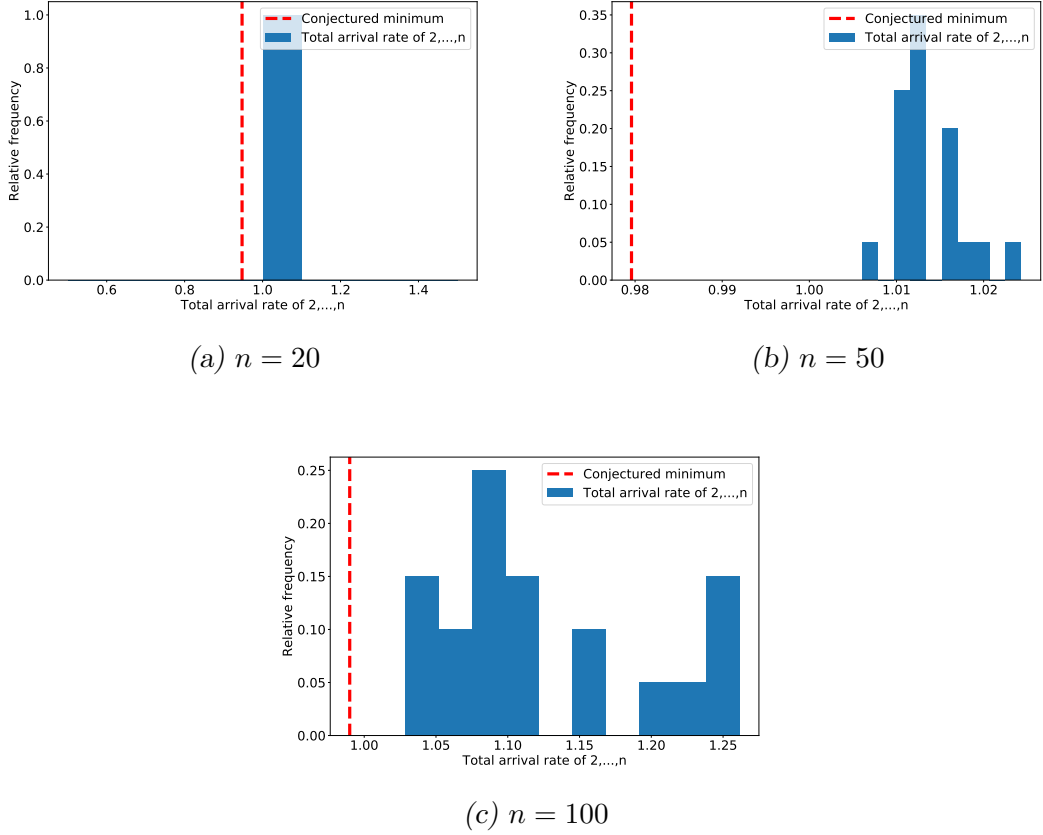
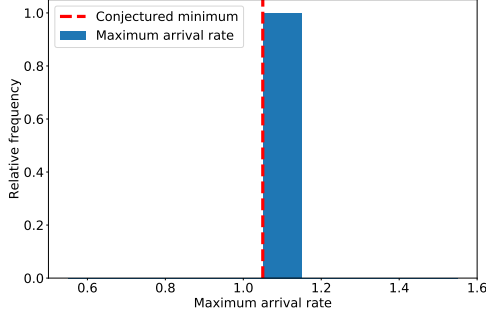
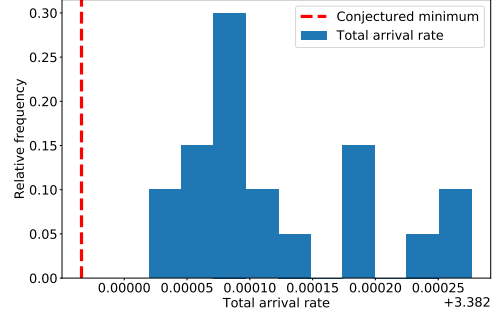


Figure B.1. Histograms of the minimum values of  $\sum_{i=2}^{n-1} \lambda_i$  that we find using an SA algorithm. We indicate the conjectured minimum value with the dashed red line. Note that the horizontal scales are different between the figures.

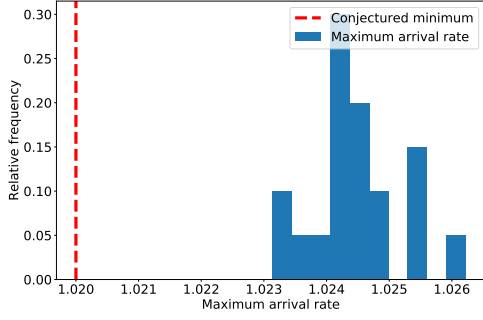
For small  $n$  (specifically, for  $n = 3, \dots, 7$ ), we have verified both conjectures by exhaustive enumeration. For larger  $n$ , we use the SA algorithm that we described in Appendix B.1.1 to find undirected networks with small values of  $\lambda_{\max}$  or  $\lambda_{\text{total}}$ . For each  $n$ , we run the SA algorithm 20 times and record the minimum objective function value from each run. For either objective function, we find that our SA algorithm yields networks with objective function values close to (but above) our conjectured minimum value for  $n = 20, 50$ , and 100 (see Figure B.2). These results therefore support both conjectures.



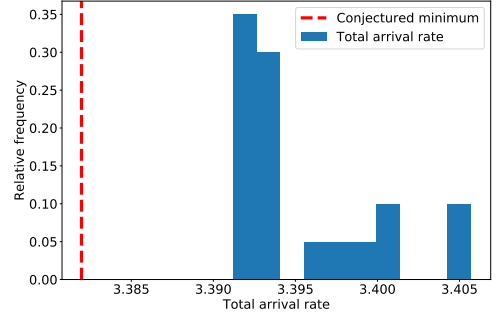
(a)  $n = 20$



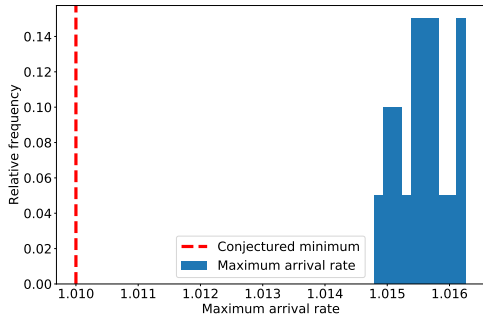
(b)  $n = 20$



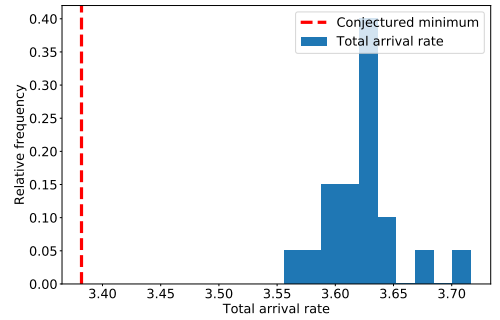
(c)  $n = 50$



(d)  $n = 50$



(e)  $n = 100$



(f)  $n = 100$

Figure B.2. Histograms of the minimum values of (left figures)  $\lambda_{\max}$  and (right figures)  $\lambda_{\text{total}}$  that we obtain using an SA algorithm. We indicate the conjectured minimum value with the dashed red line. Note that the horizontal scales are different between the figures.

## B.2 Maximum arrival rates of $G_{\mathcal{U}_n}^{\lambda_{\max}}$ and $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$

We show that  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  (depicted in Figure 4.10) does not minimize the maximum arrival rate  $\lambda_{\max}$  (depicted in Figure 4.11) over  $\mathcal{U}_n$  for  $n \geq 5$ . We do this by showing that the maximum arrival rate of  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  is larger than the maximum arrival rate of  $G_{\mathcal{U}_n}^{\lambda_{\max}}$ .

We first examine the arrival rates  $\lambda_k$  of  $G_{\mathcal{U}_n}^{\lambda_{\max}}$ . They satisfy the following traffic equations (4.5):

$$\begin{aligned}\lambda_1 &= \frac{1}{n-1}\lambda_2 + 1, \\ \lambda_2 &= \frac{1}{2}\lambda_1 + \sum_{k=3}^{n-1} \frac{1}{2}\lambda_k, \\ \lambda_k &= \frac{1}{n-1}\lambda_2, \quad \text{for } k = 3, \dots, n-1, \\ \lambda_n &= \frac{1}{2}\lambda_1 + \frac{1}{n-1}\lambda_2 + \sum_{k=3}^{n-1} \frac{1}{2}\lambda_k.\end{aligned}\tag{B.1}$$

The following values of  $\lambda_k$  satisfy Equation (B.1) and are therefore the arrival rates of  $G_{\mathcal{U}_n}^{\lambda_{\max}}$  for  $n \geq 5$ :

$$\begin{aligned}\lambda_1 &= 1 + \frac{1}{n}, \\ \lambda_2 &= 1 - \frac{1}{n}, \\ \lambda_k &= \frac{1}{n}, \quad \text{for } k = 3, \dots, n-1, \\ \lambda_n &= 1.\end{aligned}\tag{B.2}$$

It follows that  $G_{\mathcal{U}_n}^{\lambda_{\max}}$  has a maximum arrival rate  $\lambda_{\max} = \lambda_1 = 1 + 1/n$ .

We now examine the arrival rates  $\lambda_k$  of  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$ . They satisfy the following traffic equations (4.5):

$$\begin{aligned}\lambda_1 &= \frac{1}{3}\lambda_2 + 1, \\ \lambda_2 &= \frac{1}{2}\lambda_1 + \frac{1}{3}\lambda_3, \\ \lambda_k &= \frac{1}{3}\lambda_{k-1} + \frac{1}{3}\lambda_{k+1}, \quad \text{for } k = 3, \dots, n-3, \\ \lambda_{n-2} &= \frac{1}{3}\lambda_{n-3} + \frac{1}{2}\lambda_{n-1}, \\ \lambda_{n-1} &= \frac{1}{3}\lambda_{n-2}, \\ \lambda_n &= \frac{1}{2}\lambda_1 + \sum_{k=2}^{n-2} \frac{1}{3}\lambda_k + \frac{1}{2}\lambda_{n-1}.\end{aligned}\tag{B.3}$$



We establish a lower bound for  $\lambda_1$  (and therefore a lower bound for  $\lambda_{\max}$ ) by substituting the second equation of Equation (B.3) into the first equation of Equation (B.3):

$$\lambda_1 = \frac{1}{3} \left( \frac{1}{2} \lambda_1 + \frac{1}{3} \lambda_3 \right) + 1. \quad (\text{B.4})$$

Rearranging Equation (B.4) yields

$$\begin{aligned} \frac{5}{6} \lambda_1 &= \frac{1}{9} \lambda_3 + 1 \\ &> 1, \end{aligned} \quad (\text{B.5})$$

because  $\lambda_3 > 0$ , as node 3 is reachable by walkers. Consequently,  $\lambda_1 > 6/5 = 1 + 1/5$ , so  $\lambda_{\max} > 1 + 1/5 \geq 1 + 1/n$  for all  $n \geq 5$ . This shows that  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  has a smaller maximum arrival rate than  $G_{\mathcal{U}_n}^{\lambda_{\max}}$  for  $n \geq 5$  and cannot minimize  $\lambda_{\max}$  over  $\mathcal{U}_n$  for  $n \geq 5$ .

### B.3 $Q$ -optimal networks over $\mathcal{U}_n$ for $n = 3, 4$

We show that  $G_{\mathcal{U}_n}^{\lambda_{\max}}$  is the unique  $Q$ -optimal network (which we defined in Section 4.3) over  $\mathcal{U}_n$  for  $n = 3$  and  $n = 4$ . (Note that  $G_{\mathcal{U}_n}^{\lambda_{\max}}$  is identical to  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  for  $n = 3$  and  $n = 4$ .)

For  $n = 3$ , there are only 3 different networks (see Figure B.3). Among these three networks, the network  $G_{\mathcal{U}_3}^{\lambda_{\max}}$  has the smallest value of arrival rates  $\lambda_k$  for all nodes  $k = 1, 2, 3$ . (We show the arrival rates in the captions of Figures B.3a–B.3c.) Because  $Q = \sum_k \lambda_k / (\mu - \lambda_k)$  is an increasing function in  $\lambda_k$  (when we keep the other arrival rates fixed), the network  $G_{\mathcal{U}_3}^{\lambda_{\max}}$  minimizes  $Q$  for all values of  $\mu$ , so it is  $Q$ -optimal over  $\mathcal{U}_3$ .

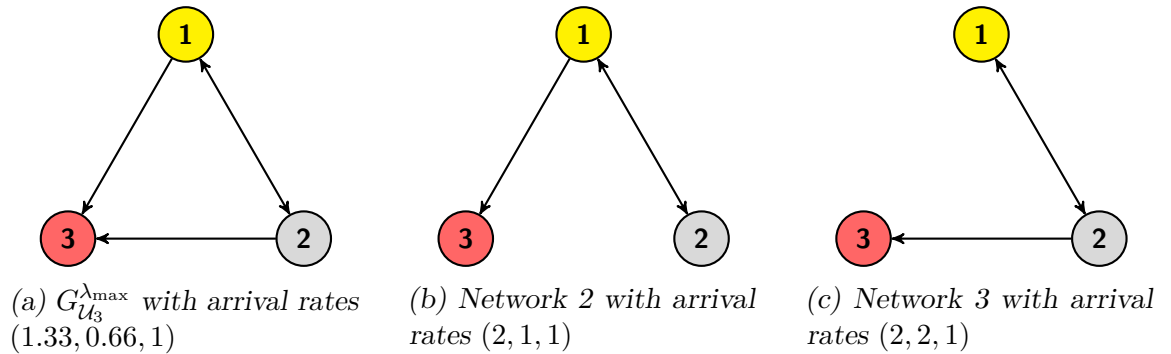


Figure B.3. All three networks in  $\mathcal{U}_3$  with their arrival rates  $(\lambda_1, \lambda_2, \lambda_3)$  (to 2 decimal places).

For  $n = 4$ , there are 28 different networks in  $\mathcal{U}_4$ . Among them, there are 11 pairs  $(G_1, G_2)$  of distinct networks such that  $G_1$  is the same as  $G_2$  except that the labels

of nodes 2 and 3 are swapped. In other words, we obtain  $G_2$  from  $G_1$  by swapping the labels of nodes 2 and 3 in  $G_1$ . For each pair  $(G_1, G_2)$ , the network  $G_2$  has the same arrival rate as  $G_1$ , except that the arrival rates of nodes 2 and 3 are swapped. It follows that the total mean queue sizes of  $G_1$  and  $G_2$  are identical for any value of  $\mu$ . We can therefore ignore one network of each pair and thus consider 17 different networks. We display these 17 networks in Figure B.4 and show their arrival rates in Table B.1. For each node  $k = 1, \dots, n$ , the arrival rate  $\lambda_k$  is smaller in  $G_{\mathcal{U}_4}^{\lambda_{\max}}$  than in all other networks except for networks 2 and 6. (Networks 2 and 6 have smaller values of  $\lambda_3$  than  $G_{\mathcal{U}_4}^{\lambda_{\max}}$ .) Using the same argument as for  $n = 3$ , the network  $G_{\mathcal{U}_4}^{\lambda_{\max}}$  has smaller values of  $Q$  than the other networks, except for networks 2 and 6, for all values of  $\mu$ .

We now show that  $G_{\mathcal{U}_4}^{\lambda_{\max}}$  also has smaller or equal values of  $Q$  than networks 2 and 6 for all values of  $\mu$ . Let  $\lambda_k^{(1)}, \lambda_k^{(2)}, \lambda_k^{(3)}$  be the arrival rate of node  $k$  in the network  $G_{\mathcal{U}_4}^{\lambda_{\max}}$ , network 3, and network 6, respectively. They take the respective values

$$\begin{aligned} (\lambda_1^{(1)}, \lambda_2^{(1)}, \lambda_3^{(1)}, \lambda_4^{(1)}) &= (1.25, 0.25, 0.75, 1), \\ (\lambda_1^{(2)}, \lambda_2^{(2)}, \lambda_3^{(2)}, \lambda_4^{(2)}) &= (1.5, 0.5, 0.5, 1), \\ (\lambda_1^{(3)}, \lambda_2^{(3)}, \lambda_3^{(3)}, \lambda_4^{(3)}) &= (2, 2/3, 2/3, 1). \end{aligned} \tag{B.6}$$

For a given service rate  $\mu$ , let  $f(x) = x/(\mu - x)$ . The function  $f$  calculates the mean queue size of a node with arrival rate  $x \in [0, \mu)$ . To show that  $G_{\mathcal{U}_4}^{\lambda_{\max}}$  has smaller values of  $Q$  than networks 2 and 6, we need to show that

$$f(\lambda_1^{(1)}) + f(\lambda_2^{(1)}) + f(\lambda_3^{(1)}) + f(\lambda_4^{(1)}) \leq f(\lambda_1^{(l)}) + f(\lambda_2^{(l)}) + f(\lambda_3^{(l)}) + f(\lambda_4^{(l)}), \tag{B.7}$$

for  $l = 2, 3$  for all sufficiently large values of  $\mu$  (to ensure that a stationary state exists). Specifically, we need to show Equation (B.7) for all values of  $\mu$  such that  $\mu > \lambda_k^l$  for all  $k$  and  $l$ . (When  $l = 2$ , we require  $\mu > 1.5$ ; when  $l = 3$ , we require  $\mu > 2$ .) Because  $\lambda_2^{(1)} \leq \lambda_2^{(l)}$  and  $\lambda_4^{(1)} = \lambda_4^{(l)}$  for  $l = 2, 3$  and  $f$  is increasing, it suffices to show that

$$f(\lambda_1^{(1)}) + f(\lambda_3^{(1)}) \leq f(\lambda_1^{(l)}) + f(\lambda_3^{(l)}), \tag{B.8}$$

for  $l = 2, 3$  and for all values of  $\mu$  such that  $\mu > \lambda_k^l$  for all  $k$  and  $l$ .

We first show Equation (B.8) for  $l = 2$ . Note that  $f$  is convex on  $[0, \mu)$ , so it satisfies

$$f(tx + (1 - t)y) \leq tf(x) + (1 - t)f(y), \tag{B.9}$$

for all  $t \in [0, 1]$  and  $x, y \in [0, \mu)$ . Setting  $t = 0.25$  and  $t = 0.75$  in Equation (B.9) yields

$$\begin{aligned} f(0.25x + 0.75y) &\leq 0.25f(x) + 0.75f(y), \\ f(0.75x + 0.25y) &\leq 0.75f(x) + 0.25f(y). \end{aligned} \tag{B.10}$$

Summing both sides of the inequalities in (B.10), we have

$$f(0.75x + 0.25y) + f(0.75x + 0.25y) \leq f(x) + f(y). \quad (\text{B.11})$$

Using  $x = \lambda_1^{(2)} = 1.5$  and  $y = \lambda_3^{(2)} = 0.5$ , we have

$$f(0.25 \times 1.5 + 0.75 \times 0.5) + f(0.75 \times 1.5 + 0.25 \times 0.5) \leq f(1.5) + f(0.5). \quad (\text{B.12})$$

That is,

$$f(1.25) + f(0.75) \leq f(1.5) + f(0.5), \quad (\text{B.13})$$

thus showing Equation (B.8) with  $(\lambda_1^{(1)}, \lambda_3^{(1)}) = (1.25, 0.75)$  and  $(\lambda_1^{(2)}, \lambda_3^{(2)}) = (1.5, 0.5)$ , as required.

For  $l = 3$ , we sum the inequalities in (B.9) with  $t = 0.875$  and  $t = 0.125$  to obtain

$$f(0.875x + 0.125y) + f(0.125x + 0.875y) \leq f(x) + f(y). \quad (\text{B.14})$$

Substituting  $x = 4/3$  and  $y = \lambda_3^{(3)} = 2/3$  in Equation (B.14), we have

$$f(1.25) + f(0.75) \leq f(4/3) + f(2/3). \quad (\text{B.15})$$

Therefore,

$$\begin{aligned} f(\lambda_1^{(1)}) + f(\lambda_3^{(1)}) &\leq f(4/3) + f(\lambda_3^{(3)}) \\ &\leq f(\lambda_1^{(3)}) + f(\lambda_3^{(3)}), \end{aligned} \quad (\text{B.16})$$

where the last inequality holds because  $f$  is increasing and  $\lambda_1^{(3)} = 2 \geq 4/3$ . Consequently,  $G_{\mathcal{U}_4}^{\lambda_{\max}}$  has smaller values of  $Q$  than networks 2–17, so it is  $Q$ -optimal over  $\mathcal{U}_4$ .

Table B.1. Arrival rates (to 2 decimal places) of 17 networks in  $\mathcal{U}_4$ .

	$\lambda_1$	$\lambda_2$	$\lambda_3$	$\lambda_4$
$G_{\mathcal{U}_4}^{\lambda_{\max}}$	1.25	0.25	0.75	1.00
Network 2	1.50	0.50	0.50	1.00
Network 3	1.33	0.33	1.00	1.00
Network 4	1.50	0.75	0.75	1.00
Network 5	1.50	0.50	1.00	1.00
Network 6	2.00	0.67	0.67	1.00
Network 7	2.00	1.00	1.00	1.00
Network 8	1.88	1.00	1.12	1.00
Network 9	1.67	0.67	2.00	1.00
Network 10	3.00	1.00	1.00	1.00
Network 11	2.00	1.00	2.00	1.00
Network 12	2.00	1.50	1.50	1.00
Network 13	2.00	1.00	3.00	1.00
Network 14	3.00	2.00	2.00	1.00
Network 15	4.00	2.00	2.00	1.00
Network 16	3.00	2.00	4.00	1.00
Network 17	3.33	2.67	3.00	1.00

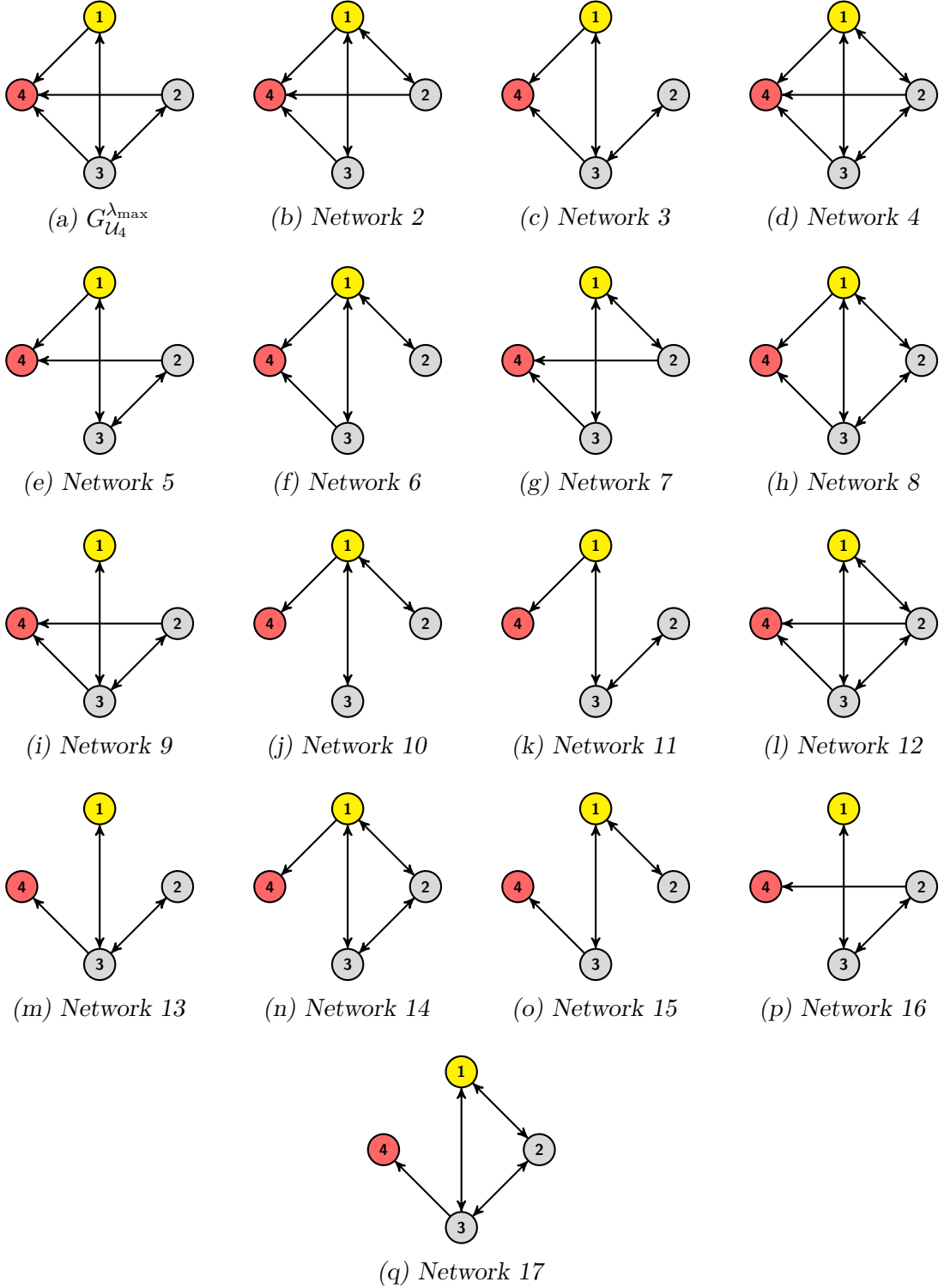


Figure B.4. 17 networks in  $\mathcal{U}_4$ . There are 28 different networks in total in  $\mathcal{U}_4$ . Among them, there are 11 pairs of networks in which each pair have the same arrival rates except that the arrival rate of node 2 and 3 are swapped. We display one network from each pair and the other 6 networks that cannot be paired up.

## B.4 Computation time of greedy algorithms

We compare the computation time of our greedy algorithms (see Table 4.1) between the efficient implementation (see Algorithm 3) and the naive implementation (Algorithm 2). We run each algorithm on Barabási–Albert graphs with parameter  $m_{BA} = 3$  for different network sizes  $n = 20$ –1000. For each network size  $n$ , we generate 10 graphs, run each edge-perturbation algorithm on it with  $T = \lceil m/2 \rceil$  and  $K = \lceil m/50 \rceil$ , and record the mean computation time in Table B.2. (We stop any computations that take longer than 10000 seconds and record the mean computation time as “> 10000”.) We run the computation on a standard desktop computer with four processing cores (Intel Core i5 7500T at 2.7 GHz) and 16GB of RAM. In practice, the efficient greedy algorithm is faster by more than an order of magnitude than the naive version, so it can handle larger graphs. We further observe that the RESTRICTED GREEDY EA algorithm (which only adds new edges that are incident to the sink) is much faster (by two orders of magnitude) than the GREEDY EA algorithm. We therefore significantly reduce the computation time by only considering edges incident to the sink without a decrease in its performance in reducing  $Q$  (see Figure 4.13).

Table B.2. Mean computation time (in seconds) of the different greedy algorithms on Barabási-Albert graphs (with  $m_{BA} = 3$ ) for different network sizes. We record the mean computation time over 10 simulations for each network size  $n$ .

n	GREEDY		GREEDY		GREEDY		RESTRICTED		RESTRICTED		GREEDY		GREEDY			
	ED	(efficient)	ED	(naive)	EA	(efficient)	EA	(naive)	GREEDY EA	(efficient)	GREEDY EA	(naive)	EDA	(efficient)	EDA	(naive)
20	0.1		0.7		0.3		2.5		0.0		0.1		0.2		1.3	
50	0.4		3.4		2.6		40.2		0.1		0.5		0.7		6.9	
100	1.1		11.8		15.3		311.0		0.2		2.2		2.2		25.5	
200	3.4		47.8		95.1		2559.7		0.7		8.6		6.5		104.5	
300	7.4		116.7		309.0		>10000		1.3		21.1		13.7		249.6	
400	13.6		233.6		796.2		>10000		2.2		41.8		23.9		501.2	
500	22.7		469.2		1690.1		>10000		3.7		75.5		40.4		970.4	
750	86.5		1585.8		>10000		>10000		14.2		271.4		157.9		3375.2	
1000	227.4		4103.6		>10000		>10000		36.3		672.1		550.6		8349.1	

## B.5 Performance of HARF and HDF with recalculation of edge rankings

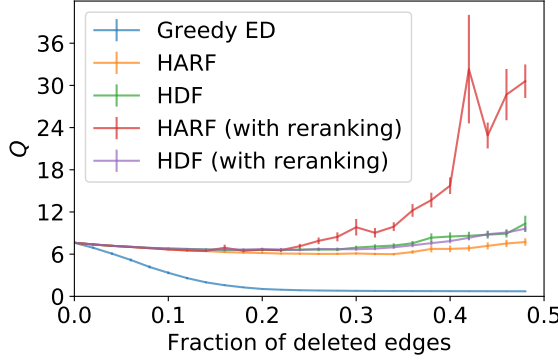
We present the results of the edge-deletion algorithms HARF and HDF (see Section 4.4.4) with ‘reranking’ (i.e., where recalculate the edge ranking after  $K = \lceil m/50 \rceil$  edge deletion). Recall that edges are ranked in decreasing order by their score, where the score of an edge  $\{i, j\}$  is  $\lambda_i \times \lambda_j$  (product of the arrival rates) in the HARF algorithm and  $d_i \times d_j$  (product of the degrees) in the HDF algorithm. The arrival rates and degree of nodes change as edges are deleted, so the rankings change over time. We apply the algorithms on graphs from the same six random-graph models (with the same parameters) as in Section 4.4.4. We delete edges until either  $T = \lceil m/2 \rceil$  edges are deleted or no more edges can be deleted without the resulting graph violating the reachability conditions.

We find that the HARF algorithm with reranking is consistently worse than the original HARF algorithm (see Figure B.5) for the six random-graph models that we tested. For random regular graphs and Watts–Strogatz graphs, in particular, the maximum arrival rate exceeds the service rate (which we set to 3 times the maximum arrival rate of the original graph) after 10% (for random regular graphs) or 30% (for WS graphs) of the edges. The total mean queue size values are unbounded beyond this point, so their curves do not continue beyond this point in Figure B.5.

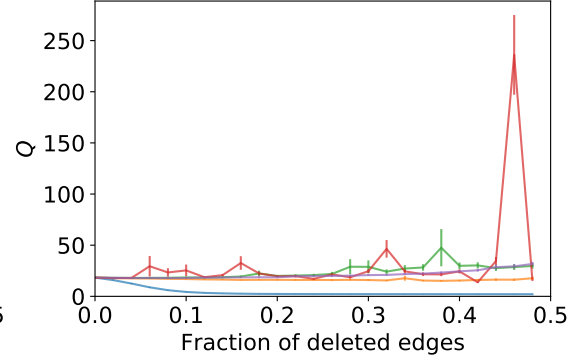
The HDF algorithm with reranking performs slightly better than the original version without reranking (see Figure B.5).

However, in all cases, the GREEDY ED algorithm outperforms the other algorithms.

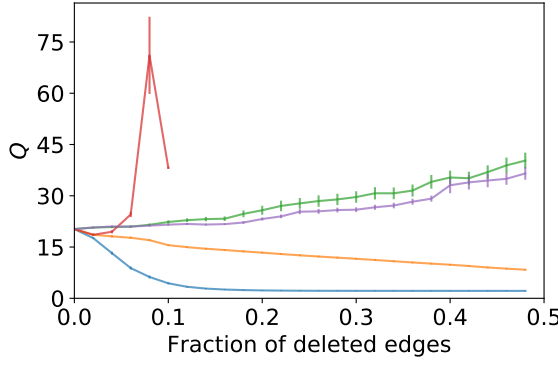




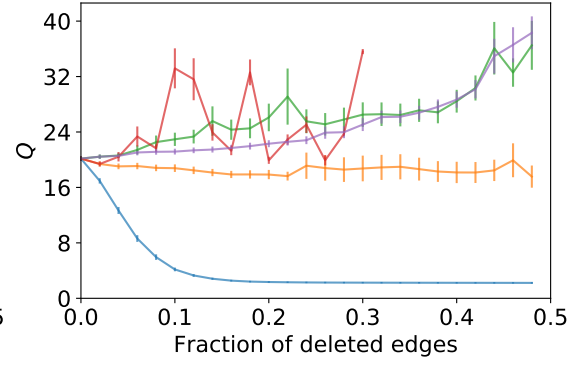
(a) Barabási–Albert networks



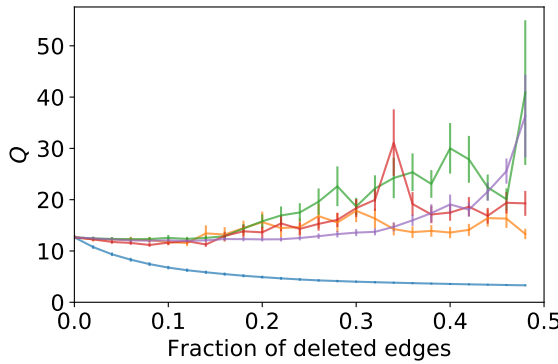
(b) Erdős–Rényi graphs



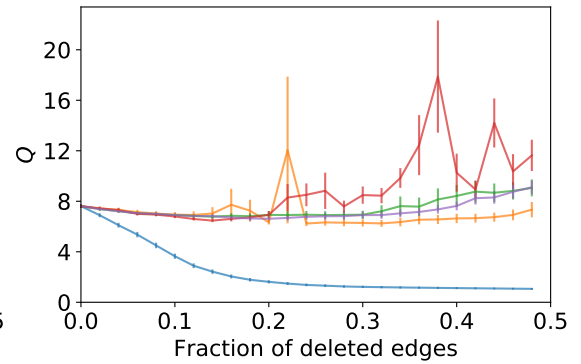
(c) Random regular graphs



(d) Watts–Strogatz graphs



(e) Random geometric graphs



(f) Chung–Lu model

Figure B.5. Comparison of the performance of the different edge-deletion algorithms (including HDF and HARF with reranking) for trying to minimize the total mean queue size  $Q$  with edge deletion. We plot the mean and standard error of  $Q$  as a function of the fraction of edges deleted.

## B.6 Reducing $\lambda_{\max}$ or $\lambda_{\text{total}}$ by adding or deleting edges

We apply variants of the greedy algorithms in Section 4.4 that reduce the maximum arrival rate  $\lambda_{\max}$  and total arrival rate  $\lambda_{\text{total}}$ . We apply the algorithms to 100 graphs from each of the six random-graph model that we used in Section 4.4. We also use the same model parameters as in Section 4.4. When we consider the GREEDY ED algorithm, we compare its performance with the performance of the HDF and HARF algorithms. Similarly, we compare the performance of the GREEDY EA and RESTRICTED GREEDY EA algorithms with the performance of the ETSF algorithm.

### B.6.1 Reducing $\lambda_{\text{total}}$

We use the variants of the greedy algorithms that reduce  $\lambda_{\text{total}}$ , and we observe that the reduction in  $\lambda_{\text{total}}$  as a function of the deleted edges (see Figures B.6, B.7, and B.9) is qualitatively similar to the reduction in  $Q$  (see Figures 4.12, 4.13, and 4.15). This suggests reducing  $Q$  (with  $\mu = 3\lambda_{\max}$ ) is approximately equivalent to reducing  $\lambda_{\text{total}}$ . In other words, we are in the large- $\mu$  regime with  $\mu = 3\lambda_{\max}$  in these random-graph models. We also make the following observations.

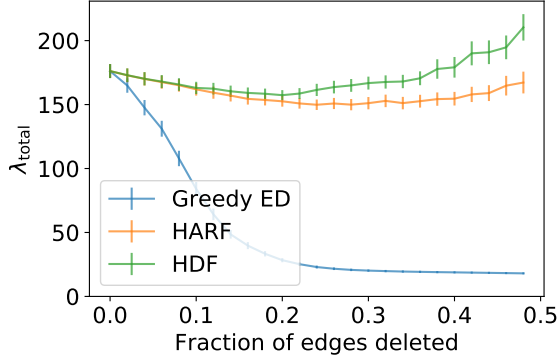
The GREEDY ED algorithm for  $\lambda_{\text{total}}$  yields a larger reduction in  $\lambda_{\text{total}}$  than the HARF and HDF algorithms (see Figure B.6).

The GREEDY EA algorithm for  $\lambda_{\text{total}}$  yields the same reduction in  $\lambda_{\text{total}}$  as the RESTRICTED GREEDY EA algorithm for  $\lambda_{\text{total}}$  and the ETSF algorithms (see Figure B.7). Furthermore, the edges that the GREEDY EA algorithm adds largely agree with the edges that the other two edge-addition algorithms add (see Figure B.8). The three algorithms reduce  $\lambda_{\text{total}}$  significantly more than the GREEDY ED algorithm.

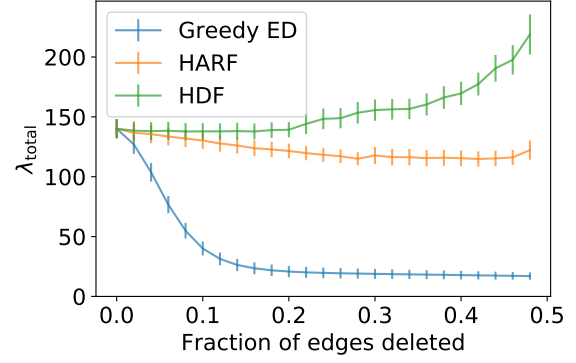
The RESTRICTED GREEDY EDA algorithm for  $\lambda_{\text{total}}$  performs slightly better (i.e., yields a slightly larger reduction in  $\lambda_{\text{total}}$ ) than the RESTRICTED GREEDY EA algorithm and much better than the GREEDY ED algorithm (see Figure B.9). Furthermore, it reduces  $\lambda_{\text{total}}$  to a value close to the conjectured minimum value of  $\lambda_{\text{total}}$ ; the mean values of the ratio  $R_{\lambda_{\text{total}}}$  between the minimum achieved value of  $\lambda_{\text{total}}$  and the conjectured minimum value of  $\lambda_{\text{total}}$  are all close to 1. (The conjectured minimum value of  $\lambda_{\text{total}}$  is the total arrival rate of  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$  (see Section 4.3.5).) For all three algorithms, the mean value of  $R_{\lambda_{\text{total}}}$  for each random-graph model is close to the corresponding mean value of  $R_Q$  (the ratio between the minimum achieved value of  $Q$  and the conjectured minimum value of  $Q$ ) in Section 4.4.6 (see Tables B.3 and 4.2).

Table B.3. Mean value of  $R_{\lambda_{\text{total}}}$  for the RESTRICTED GREEDY EDA, RESTRICTED GREEDY EA, and GREEDY ED algorithms for  $\lambda_{\text{total}}$  for the six different random-graph models (defined in Section 4.4.3). We generate 100 random graphs for each random-graph model and run the three algorithm on them.

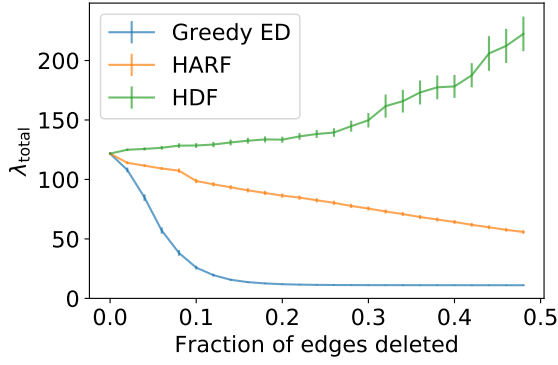
Model	RESTRICTED GREEDY EDA	RESTRICTED GREEDY EA	GREEDY ED
BA	1.014	2.268	5.077
ER	1.008	2.368	4.173
WS	1.000	2.365	3.658
RRG	1.000	2.378	3.313
RGG	1.024	2.214	57.308
Chung–Lu	1.044	2.321	22.295



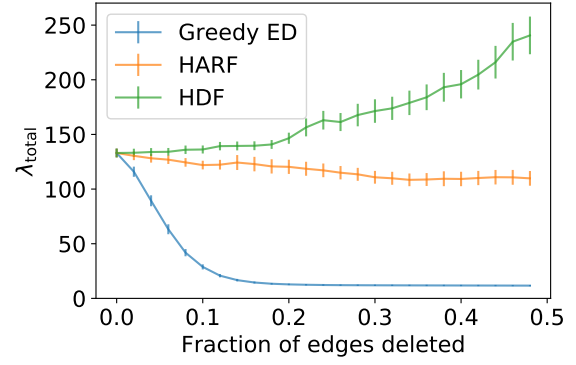
(a) *Barabási–Albert networks*



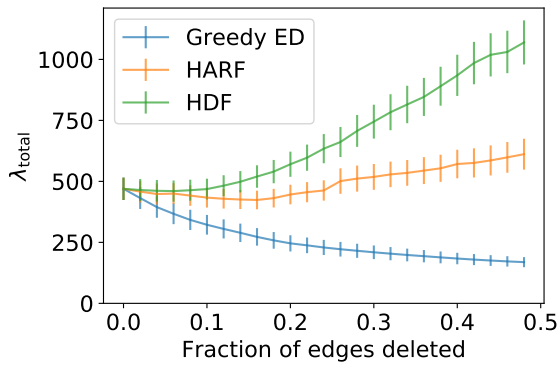
(b) *Erdős–Rényi graphs*



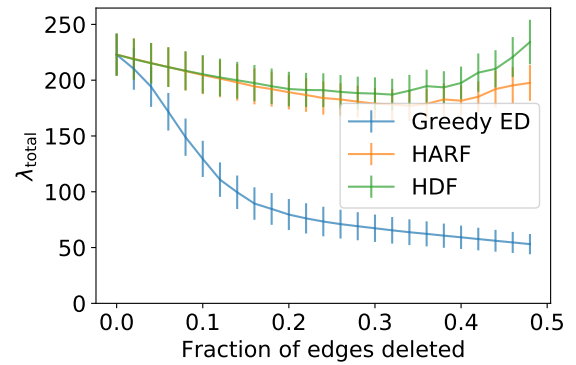
(c) *Random regular graphs*



(d) *Watts–Strogatz graphs*

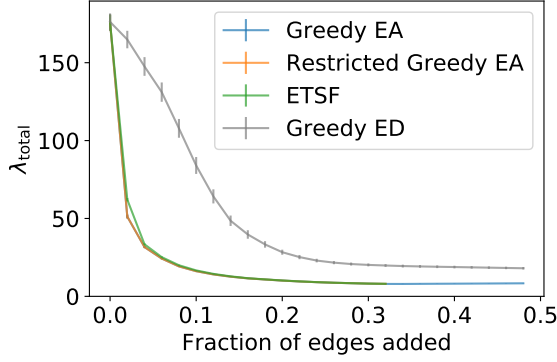


(e) *Random geometric graphs*

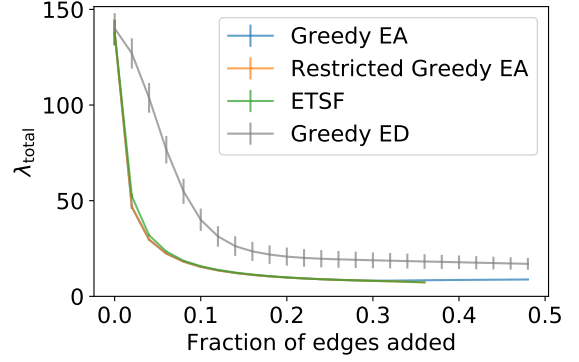


(f) *Chung–Lu model*

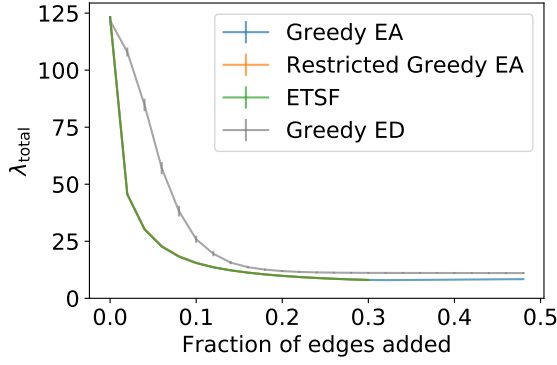
Figure B.6. Comparison of the performance of the GREEDY ED, HDF, and HARF algorithms in decreasing the total arrival rate  $\lambda_{\text{total}}$  with edge deletion. We plot the mean and standard error of  $\lambda_{\text{total}}$  as a function of the fraction of edges deleted.



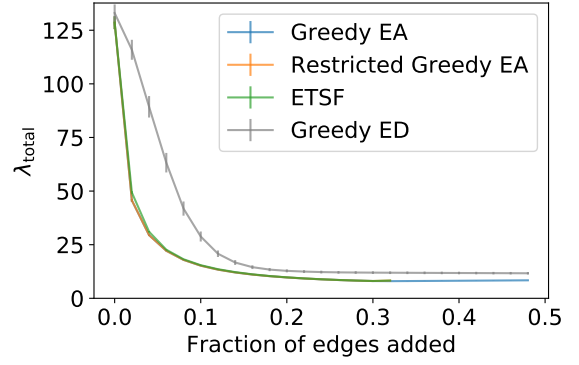
(a) *Barabási–Albert networks*



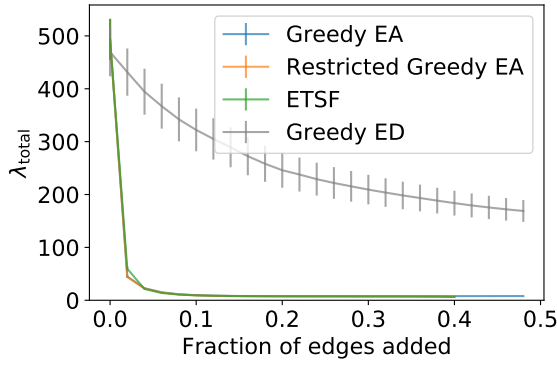
(b) *Erdős–Rényi graphs*



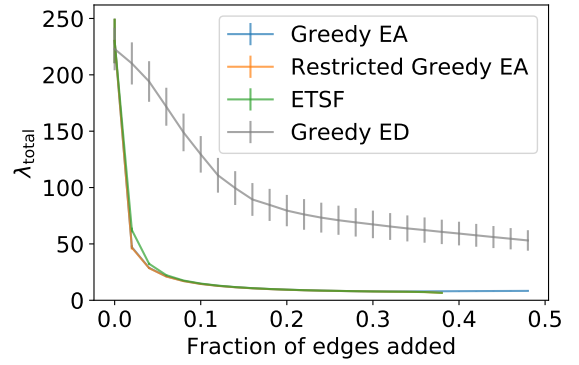
(c) *Random regular graphs*



(d) *Watts–Strogatz graphs*

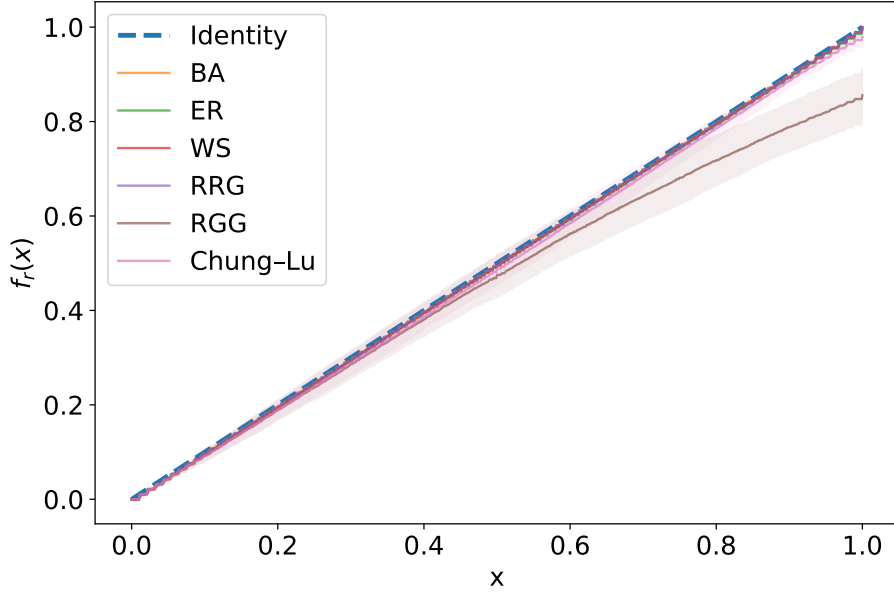


(e) *Random geometric graphs*

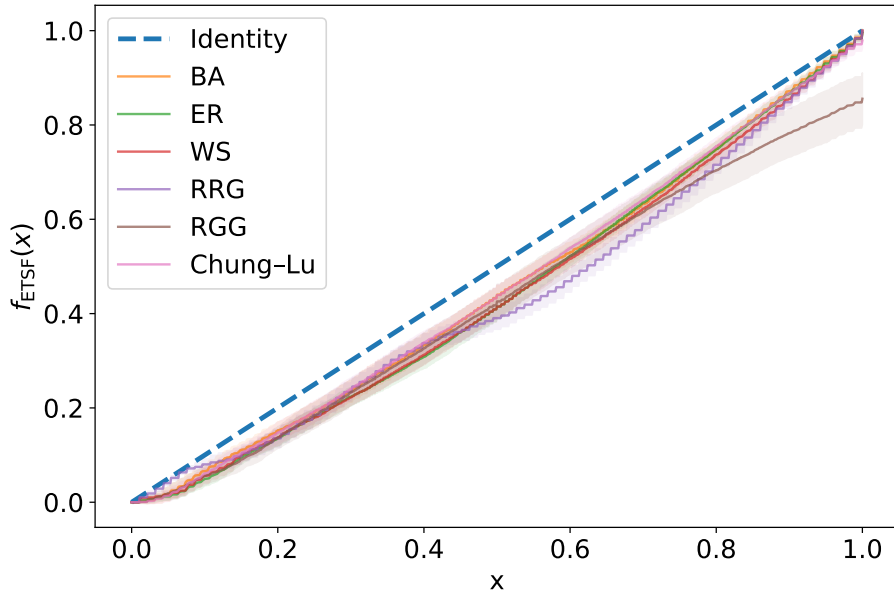


(f) *Chung–Lu model*

Figure B.7. Comparison of the performance of the GREEDY EA, RESTRICTED GREEDY EA, and ETSF algorithms in decreasing the total arrival rate  $\lambda_{\text{total}}$  with edge addition. We plot the mean and standard error of  $\lambda_{\text{total}}$  as a function of the fraction of edges added (i.e., the number of edges added divided by the number of edges in the original graph).

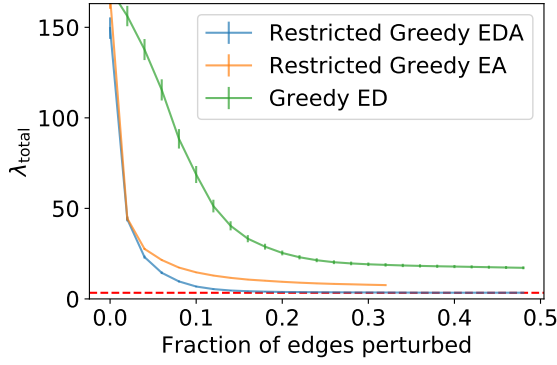


(a) Function  $f_r(x)$  (defined in Section 4.4.5) that shows the agreement between the added edges in the GREEDY EA algorithm and the added edges in the RESTRICTED GREEDY EA algorithm.

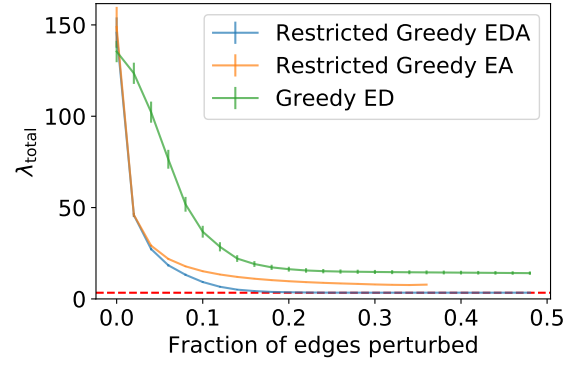


(b) Function  $f_{\text{ETSF}}(x)$  (defined in Section 4.4.5) that shows the agreement between the added edges in the GREEDY EA algorithm and the added edges in the ETSF algorithm.

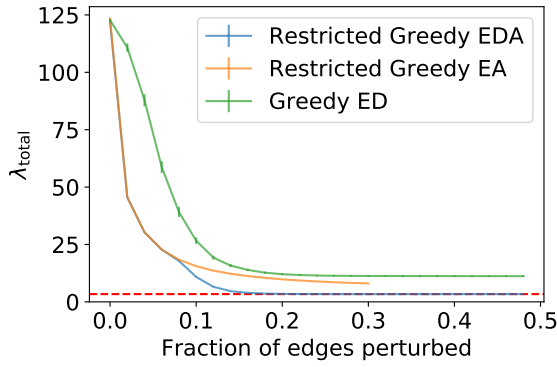
Figure B.8. Comparison between the GREEDY EA algorithm for  $\lambda_{\text{total}}$  with (a) the RESTRICTED GREEDY EA algorithm for  $\lambda_{\text{total}}$  and (b) the ETSF algorithm. We plot the mean values of  $f_r(x)$  and  $f_{\text{ETSF}}(x)$  (solid curve) in the respective panel and standard deviation (shaded area) for each of the six random-graph models. We also plot the identity line (dashed) in blue. The mean values of  $f_r(x)$  and  $f_{\text{ETSF}}(x)$  are close to the identity line, so the three algorithms largely add the same edges.



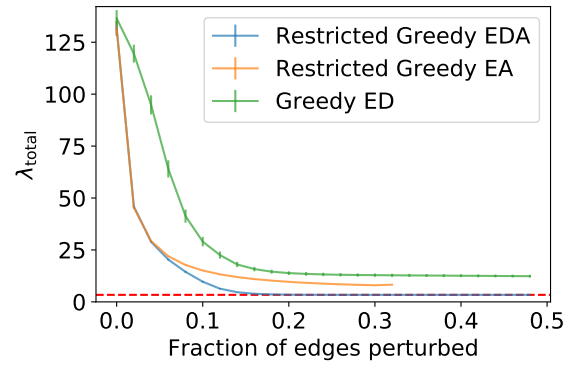
(a) *Barabási–Albert networks*



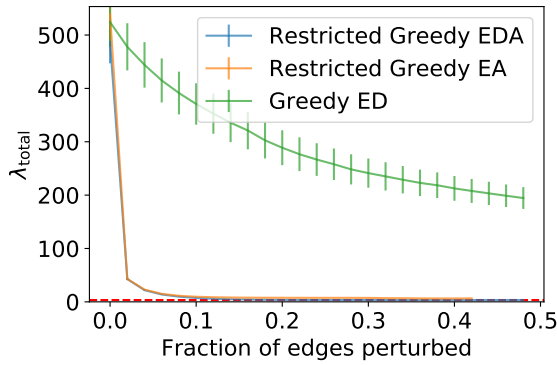
(b) *Erdős–Rényi graphs*



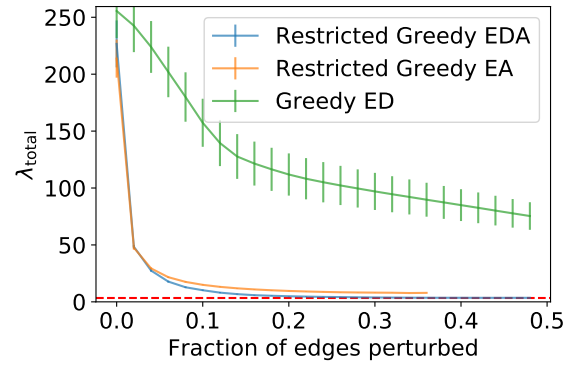
(c) *Random regular graphs*



(d) *Watts–Strogatz graphs*



(e) *Random geometric graphs*



(f) *Chung–Lu model*

Figure B.9. Comparison of the performance of the RESTRICTED GREEDY EDA, RESTRICTED GREEDY EA, and GREEDY ED algorithms in decreasing the total arrival rate  $\lambda_{\text{total}}$  with edge perturbations. We plot the mean and standard error of  $\lambda_{\text{total}}$  as a function of fraction of perturbed edges (i.e., the number of edge perturbations divided by the number of edges in the original graph). The red dashed line shows the conjectured minimum value of  $\lambda_{\text{total}}$  (which is achieved by  $G_{\mathcal{U}_n}^{\lambda_{\text{total}}}$ ).

### B.6.2 Reducing $\lambda_{\max}$

When we use the variants of the greedy algorithms that reduce  $\lambda_{\max}$ , we observe some differences in the results compared to when we use these algorithms to reduce  $Q$  (with  $\mu = 3\lambda_{\max}$ ) or  $\lambda_{\text{total}}$ . We highlight the differences and similarities in this section.

Unlike for  $Q$  (with  $\mu = 3\lambda_{\max}$ ) and  $\lambda_{\text{total}}$ , the GREEDY ED algorithm for  $\lambda_{\max}$  does not always achieve a reduction in  $\lambda_{\max}$ . For the random regular graph and the Watts–Strogatz graph,  $\lambda_{\max}$  stays roughly the same as we delete edges using the GREEDY ED algorithm for  $\lambda_{\max}$  (see Figure B.10). However, similar to when minimizing  $Q$  or  $\lambda_{\text{total}}$ , the algorithm performs better than the HARF and HDF algorithms.

Similar to when we reduce  $Q$  (with  $\mu = 3\lambda_{\max}$ ) and  $\lambda_{\text{total}}$ , the performance of the GREEDY EA algorithm for  $\lambda_{\max}$  (in terms how much it reduces  $\lambda_{\max}$ ) is similar to the performance of the RESTRICTED GREEDY EA algorithm for  $\lambda_{\max}$  and the ETSF algorithm (see Figure B.11). However, they do not add the same edges (see Figure B.12). The first 20% of added edges largely agree between the three algorithms, but they differ after that. This disagreement in the last 80% of added edges does not lead to a significant difference in the performance of these algorithms, as any added edges after the first 20% do not appear to yield a significant reduction in  $\lambda_{\max}$ . The algorithms reduce  $\lambda_{\max}$  significantly more than the GREEDY ED algorithm for  $\lambda_{\max}$ .

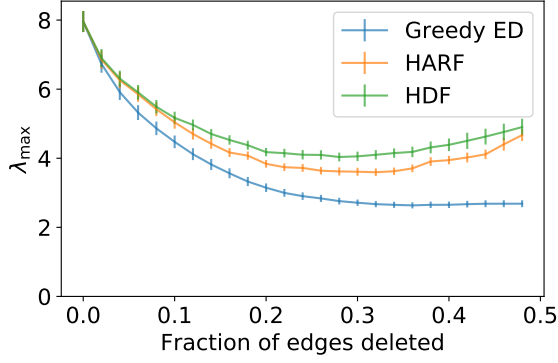
Similar to when we reduce  $Q$  (with  $\mu = 3\lambda_{\max}$ ) and  $\lambda_{\text{total}}$ , the RESTRICTED GREEDY EDA algorithm for  $\lambda_{\max}$  performs slightly better than the RESTRICTED EA algorithm and much better than the GREEDY ED algorithm. However, the achieved value of  $\lambda_{\max}$  when using the RESTRICTED GREEDY EDA algorithm is not as close to the conjectured minimum value compared using the algorithm to reduce  $Q$  (with  $\mu = 3\lambda_{\max}$ ) or  $\lambda_{\text{total}}$ . (The conjectured minimum value of  $\lambda_{\max}$  is the maximum arrival rate of  $G_{\mathcal{U}_n}^{\lambda_{\max}}$  (see Section 4.3.5).) For the RESTRICTED GREEDY EDA algorithm, the mean value of the ratio  $R_{\lambda_{\max}}$  between the minimum achieved value of  $\lambda_{\max}$  and conjectured minimum value of  $\lambda_{\max}$  is between 1.07 and 1.11 (see Table B.4). These values are larger than the corresponding mean values of  $R_Q$  and  $R_{\lambda_{\max}}$ , which are between 1.00 and 1.05 (see Tables B.3 and 4.2). For the other two algorithms, the mean values of  $R_{\lambda_{\max}}$  are significantly smaller than the corresponding mean values of  $R_Q$  and  $R_{\lambda_{\max}}$ . Therefore, they achieve values that are much closer to the conjectured minimum than when we used the same algorithms for reducing  $Q$  (with  $\mu = 3\lambda_{\max}$ ) or  $\lambda_{\text{total}}$ . The RESTRICTED GREEDY EA algorithm achieves a value of  $\lambda_{\max}$  that is only about 20–40% larger than the conjectured minimum; by contrast, when minimizing



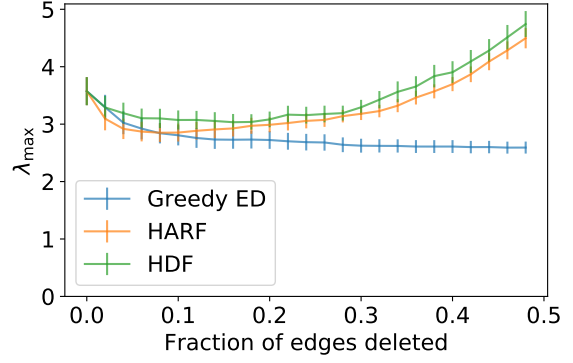
$Q$  or  $\lambda_{\text{total}}$ , the achieved values of this algorithm are at least twice the conjectured minimum (i.e., over 100% larger).

*Table B.4. Mean value of  $R_{\lambda_{\text{max}}}$  for the RESTRICTED GREEDY EDA, RESTRICTED GREEDY EA, and GREEDY ED algorithms for  $\lambda_{\text{max}}$  for the six different random-graph models. We generate 100 random graphs for each random-graph model and run the three algorithm on them.*

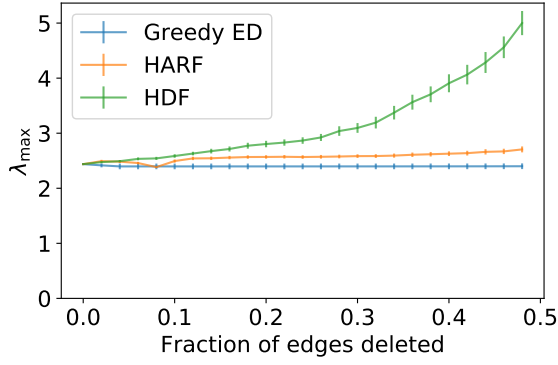
Model	RESTRICTED GREEDY EDA	RESTRICTED GREEDY EA	GREEDY ED
BA	1.075	1.180	2.463
ER	1.076	1.183	2.275
WS	1.082	1.196	2.165
RRG	1.081	1.188	2.260
RGG	1.108	1.362	9.917
Chung–Lu	1.098	1.165	3.383



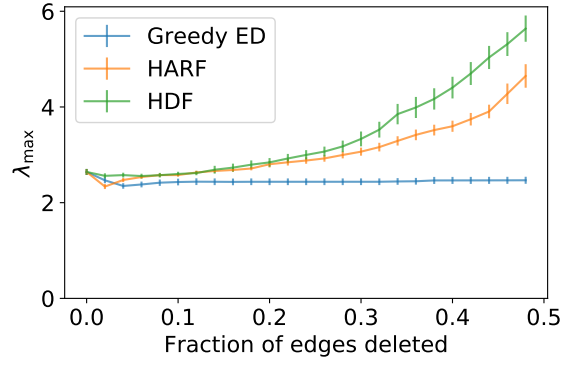
(a) *Barabási–Albert networks*



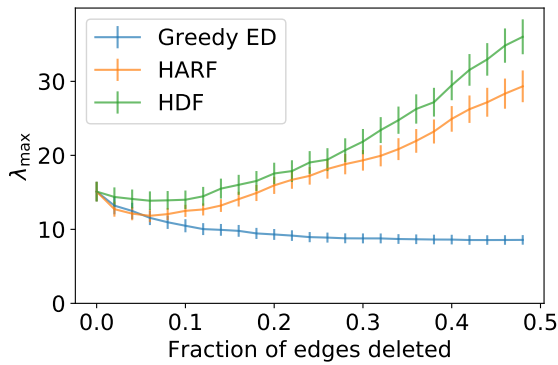
(b) *Erdős–Rényi graphs*



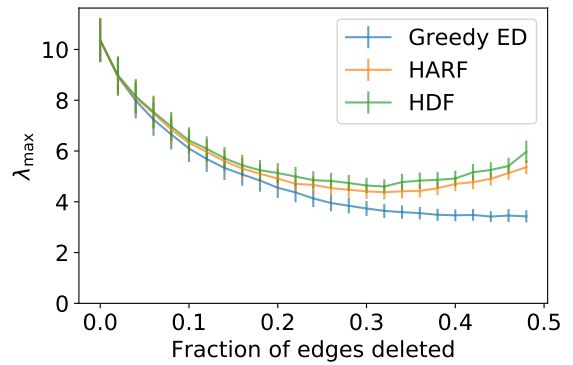
(c) *Random regular graphs*



(d) *Watts–Strogatz graphs*

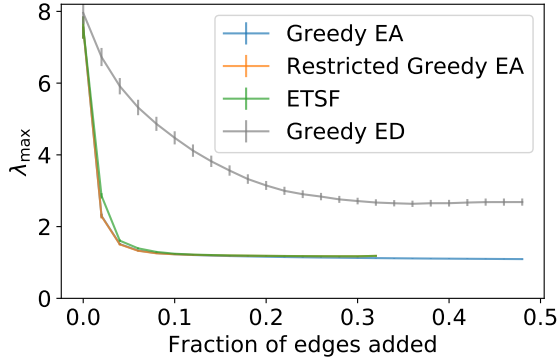


(e) *Random geometric graphs*

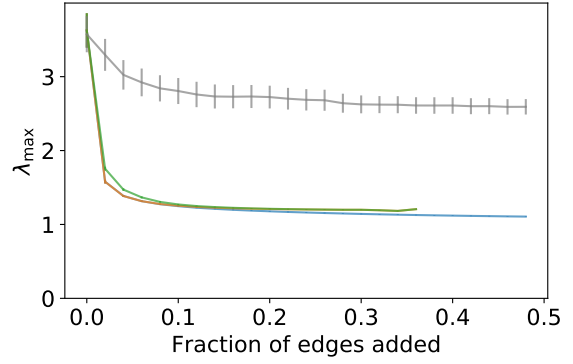


(f) *Chung–Lu model*

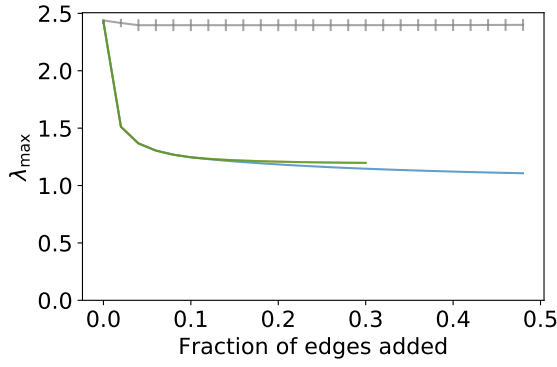
Figure B.10. Comparison of the performance of the GREEDY ED, HDF, and HARF algorithms in decreasing the maximum arrival rate  $\lambda_{\max}$  with edge deletion. We plot the mean and standard error of  $\lambda_{\max}$  as a function of the fraction of edges deleted.



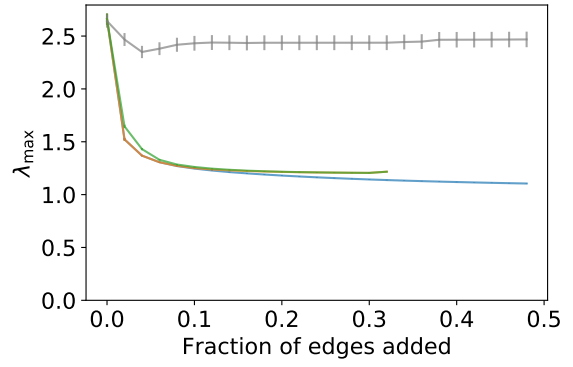
(a) Barabási–Albert networks



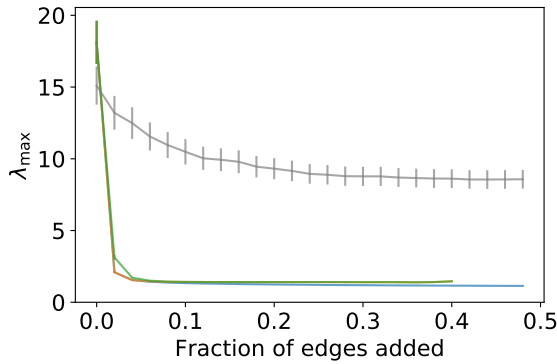
(b) Erdős–Rényi graphs



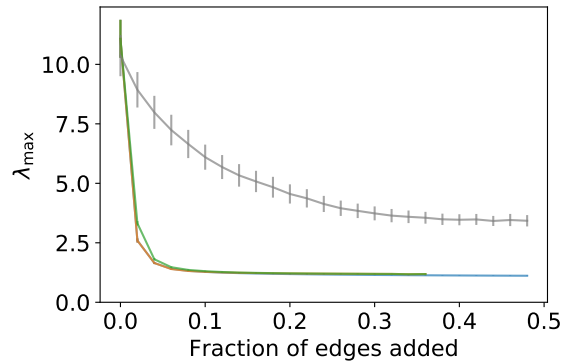
(c) Random regular graphs



(d) Watts–Strogatz graphs

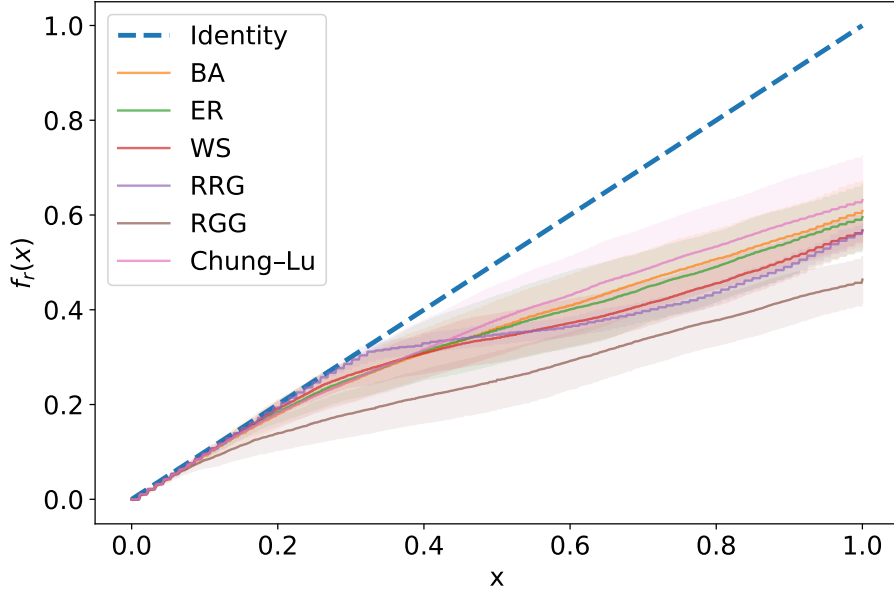


(e) Random geometric graphs

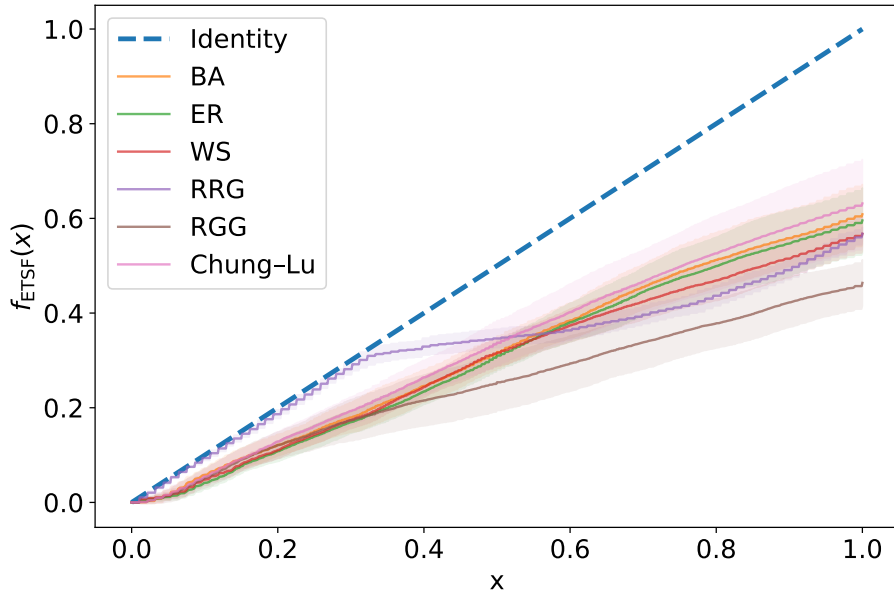


(f) Chung–Lu model

Figure B.11. Comparison of the performance of the GREEDY EA, RESTRICTED GREEDY EA, and ETSF algorithms in decreasing the maximum arrival rate  $\lambda_{\max}$  with edge addition. We plot the mean and standard error of  $\lambda_{\max}$  as a function of the fraction of edges added (i.e., the number of edges added divided by the number of edges in the original graph).

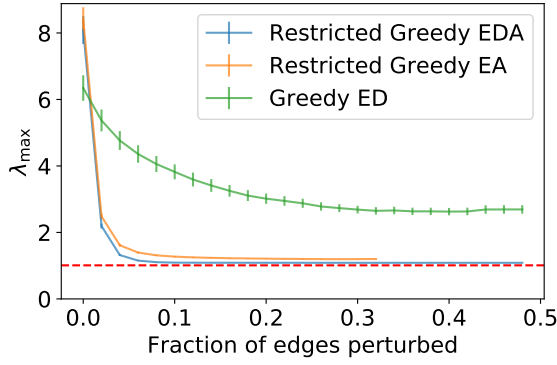


(a) Function  $f_r(x)$  (defined in Section 4.4.5) that shows the agreement between the added edges in the GREEDY EA algorithm and the added edges in the RESTRICTED GREEDY EA algorithm.

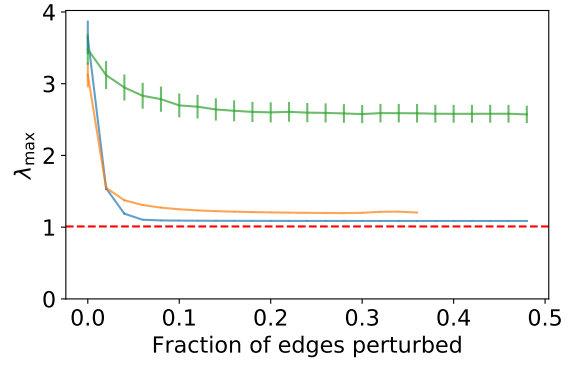


(b) Function  $f_{\text{ETSF}}(x)$  (defined in Section 4.4.5) that shows the agreement between the added edges in the GREEDY EA algorithm and the added edges in the ETSF algorithm.

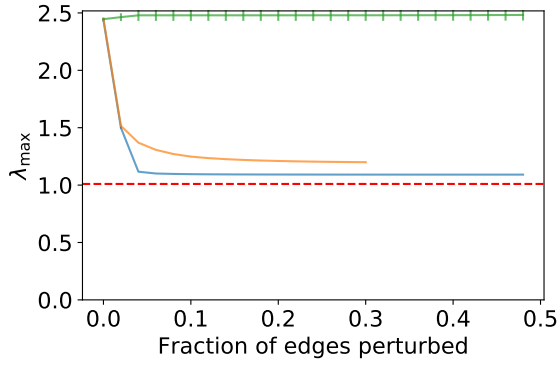
Figure B.12. Comparison between the GREEDY EA algorithm for  $\lambda_{\max}$  with (a) the RESTRICTED GREEDY EA algorithm for  $\lambda_{\max}$  and (b) the ETSF algorithm. We plot the mean values of  $f_r(x)$  and  $f_{\text{ETSF}}(x)$  (solid curve) in the respective panel and standard deviation (shaded area) for each of the six random-graph models. We also plot the identity line (dashed) in blue. The mean values of  $f_r(x)$  and  $f_{\text{ETSF}}(x)$  are close to the identity line, so the three algorithms largely add the same edges.



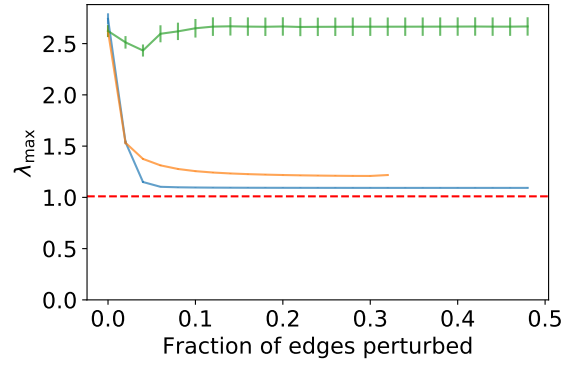
(a) Barabási–Albert networks



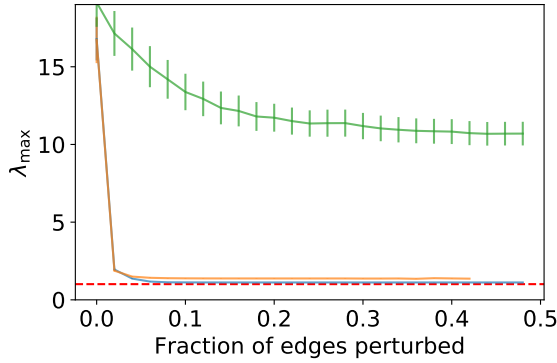
(b) Erdős–Rényi graphs



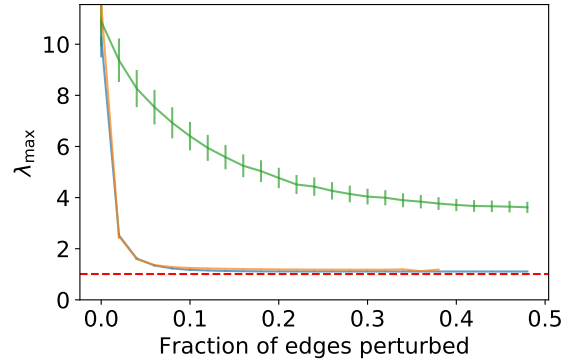
(c) Random regular graphs



(d) Watts–Strogatz graphs



(e) Random geometric graphs



(f) Chung–Lu model

Figure B.13. Comparison of the performance of the RESTRICTED GREEDY EDA, RESTRICTED EA ALGORITHM and GREEDY ED algorithms in decreasing the maximum arrival rate  $\lambda_{\max}$  with edge perturbations. We plot the mean and standard error of  $\lambda_{\max}$  as a function of fraction of perturbed edges (i.e., the number of edge perturbations divided by the number of edges in the original graph). The red dashed line shows the conjectured minimum value of  $\lambda_{\max}$  (which is achieved by  $G_{\mathcal{U}_n}^{\lambda_{\max}}$ ).

# Appendix C

## Appendix to Chapter 5

### C.1 Performance of the doubly-constrained gravity model with an exponential deterrence function

In the doubly-constrained gravity model with an exponential deterrence function, the OD matrix  $\mathbf{T}^{\text{model}}$  is given by Equation (5.3) with

$$f_{ij} = f_{\text{GE}}(d_{ij}) = e^{-\gamma d_{ij}/l}, \quad (\text{C.1})$$

where  $l$  is a spatial normalization factor, chosen to be the mean zone length.

We find that, on average, the gravity model performs worse with an exponential deterrence function than with a power-law deterrence function in terms of the CPC score and  $\text{NRMSE}_v$  (see Table C.1).

*Table C.1. Mean CPC scores and  $\text{NRMSE}_v$  when fitting the doubly-constrained gravity model with an exponential deterrence function versus the doubly-constrained gravity model with a power-law deterrence function.*

Model	Mean CPC	Mean $\text{NRMSE}_v$
Gravity (exponential)	0.677	0.049
Gravity (power law)	0.686	0.045

### C.2 Estimating $\{O_k^{\text{data}}\}_{k=1}^n$ and $D_1^{\text{data}}$ from purchase data

In our models, we used  $\{O_k^{\text{data}}\}_{k=1}^n$  and  $D_1^{\text{data}}$  to calculate mobility flow. In Chapter 5, we assumed that we know these values. In this section, we show how to estimate

$\{O_k\}_{k=1}^n$  and  $D_1^{\text{data}}$  from customer-level purchase data and compare its values with the empirical values from the shopping journeys derived from ordered customer-basket data. We use a data set for Store A that includes for each customer  $c$  their shopping journey (i.e., a list of zones at which the customer likely<sup>1</sup> picked up their purchased item) and the list of items that they purchased during their shopping journey. Note the important difference between a shopping journey and a list of purchase items. A shopping journey is derived from an ordered customer basket, whereas a list of items is derived from an unordered customer basket. Ordered customer-basket data is not generally available to retailers, whereas every retailer with a point of sale (POS) has unordered customer-basket data.

From each unordered customer basket (i.e., a list of items), we can use item-location data to identify the possible zones in which a customer could have picked up each item. We assume that a customer never visits a zone more than once to purchase something. For example, if a customer bought several items that are located in zone  $k$ , we assume that a customer picked up all these items during their first visit to zone  $k$ . The main challenge is how to account for purchased multi-located items. We calculate the number  $O_k^{\text{data},c}$  of trips that start from nodes  $k = 1, \dots, n$  for each customer  $c$  as follows. If a customer buys an item that is located only in zone  $k$ , we set  $O_k^{\text{data},c} = 1$ . Otherwise, we check whether a customer buys an item that is located both in zone  $k$  and in other zones. (In other words, there are multiple possible zone locations for that item.) If this is the case, let  $M$  be the number of such purchased items, and let  $Z_1, \dots, Z_M$  be the number of possible zone locations for each of the  $M$  items. We then set  $O_k^{\text{data},c} = \min \left\{ \sum_{l=1}^M 1/Z_l, 1 \right\}$ . Therefore, each item that is located in zone  $k$  and in  $Z - 1$  other zones counts as a  $1/Z$  of a visit, with  $O_k^{\text{data},c}$  capped at 1. If a customer has not purchased any items that are located in zone  $k$ , then  $O_k^{\text{data},c} = 0$ . By construction, the value of  $O_k^{\text{data},c}$  is at most 1, and it can take a fractional value when a customer buys items that are located at  $k$  as well as in other zones.

We calculate  $O_k^{\text{data}}$  (with  $k = 1, \dots, n$ ) with the formula

$$O_k^{\text{data}} = \sum_c O_k^{\text{data},c} + \delta_{k1} C_s, \quad (\text{C.2})$$

where  $C_s$  is the number of journeys in the data set and  $\delta_{k1}$  is the Kronecker delta. The term  $\delta_{k1} C_s$  accounts for the first trips of each shopping journey, as these start at

---

<sup>1</sup>As discussed in Section 5.2, we choose a list of zones that minimizes the distance of a shortest path visiting all items.

the entrance of a store. We calculate  $D_1^{\text{data}}$  with the formula

$$D_1^{\text{data}} = \sum_c O_1^{\text{data},c}. \quad (\text{C.3})$$

Finally, we rescale  $\{O_k^{\text{data}}\}_{k=1}^n$ , and  $D_1^{\text{data}}$  by dividing by  $\rho$  (i.e., the number of baskets in the data set divided by the total number of baskets during the time period).

In Figure C.1, we compare the values of  $\{O_k^{\text{data}}\}_{k=1}^n$  and  $D_1^{\text{data}}$  that we calculate directly from the shopping journeys (i.e., the empirical values) with ones that we estimate from the purchase data using the above procedure for Store A. In our calculations, they agree closely so we conclude that we can estimate  $O_k$  and  $D_k$  using only customer-level purchase data (and we do not need ordered customer-basket data).

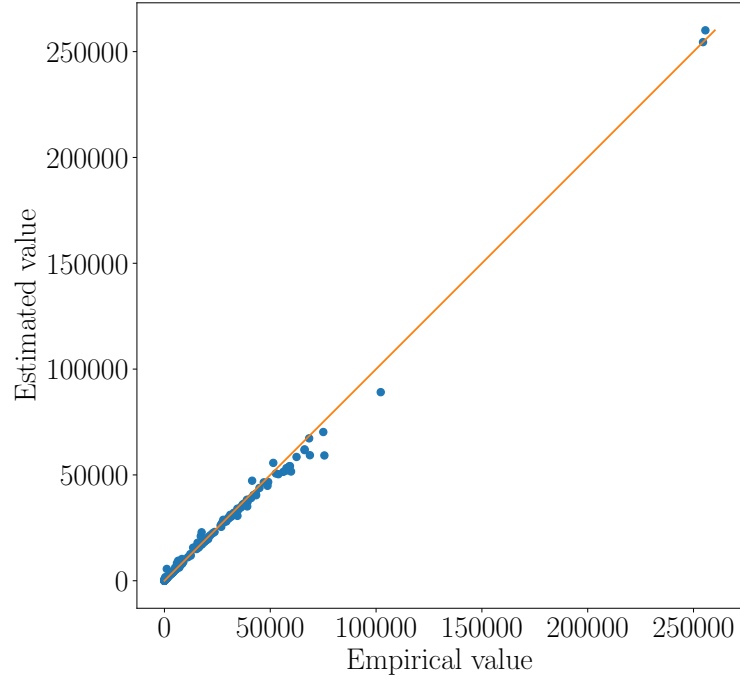


Figure C.1. Comparison of our estimated and empirical values of  $O_k$ ,  $D_1^{\text{data}}$ , and  $D_n^{\text{data}}$  for Store A.



# Appendix D

## Appendix to Chapter 6

### D.1 Transition matrix of shortest-path walker

We prove that the transition matrix  $\mathbf{P}^{(i,j)}$  of walkers in the group associated with the origin–destination pair  $(i, j)$  has entries

$$P_{kl}^{(i,j)} = \begin{cases} \frac{s_{kl}}{\sum_{l'=1}^n s_{kl'}}, & \text{if } \sum_{l'=1}^n s_{kl'} > 0, \\ 0, & \text{otherwise,} \end{cases} \quad (\text{D.1})$$

where  $s_{kl}$  is equal to the number of shortest paths from  $i$  to  $j$  that traverse the directed edge  $(k, l)$  if  $(k, l) \in E$  and is equal to 0 otherwise. (Note that Equation (D.1) corresponds to Equation (6.2) in Chapter 6.) Recall that a walker in the group associated with  $(i, j)$  enters the network at node  $i$  and takes a shortest path, chosen uniformly at random from all shortest paths from  $i$  to  $j$ . At node  $j$ , the walker is removed from the network.

Suppose that there are  $S$  different shortest paths from  $i$  to  $j$ . We order the shortest paths arbitrarily from 1 to  $S$ . Let  $\mathbf{P}^{(s)}$  for  $s = 1, \dots, S$  be the transition matrix of a walker that takes the  $s^{\text{th}}$  shortest path from  $i$  to  $j$ . (We do not include a  $(i, j)$  superscript on  $\mathbf{P}^{(s)}$  for notational ease.) The transition matrix  $\mathbf{P}^{(s)}$  has entries

$$P_{kl}^{(s)} = \begin{cases} 1, & \text{if the } s^{\text{th}} \text{ shortest path traverses } (k, l) \in E, \\ 0, & \text{otherwise.} \end{cases} \quad (\text{D.2})$$

Each walker in the group associated with  $(i, j)$  traverses the network according to one of the  $S$  transition matrices; in particular, it traverses with the  $s^{\text{th}}$  transition matrix  $\mathbf{P}^{(s)}$  with probability  $1/S$ . When  $(k, l) \notin E$  or  $k$  is not in any of the shortest paths from  $i$  to  $j$ , we have  $P_{kl}^{(i,j)} = 0$ . To find the probability  $P_{kl}^{(i,j)}$  of a walker at  $k$  to move to  $l$  in the next step (where  $(k, l) \in E$  and  $k$  is in one of the shortest paths from  $i$  to  $j$ ), we condition on the shortest path that it takes. For each node  $k \neq i, j$ , let  $\mathcal{S}_k$

be the number of the shortest paths from  $i$  to  $j$  that traverse  $k$ . A walker at  $k$  must traverse according to a transition matrix  $\mathbf{P}^{(s)}$  for some  $s \in \mathcal{S}_k$ . The probability of a walker at  $k$  to traverse the network according to the transition matrix  $\mathbf{P}^{(s)}$  is equal to  $1/|\mathcal{S}_k|$  if  $s \in \mathcal{S}_k$  and 0 otherwise.

The total probability of taking the transition from  $k$  to  $l$  is then

$$\begin{aligned} P_{kl}^{(i,j)} &= \frac{1}{|\mathcal{S}_k|} \sum_{s \in \mathcal{S}_k} P_{kl}^{(s)} \\ &= \frac{s_{kl}}{|\mathcal{S}_k|}, \end{aligned} \quad (\text{D.3})$$

when  $|\mathcal{S}_k| > 0$  and 0 otherwise. Finally,  $|\mathcal{S}_k| = \sum_{l'=1}^n s_{kl'}$ , because every shortest path that traverses  $k \neq i, j$  visits  $k$  only once and traverses only one edge  $(k, l')$  for some  $l' \in V$ . (Note that a shortest path never visits a node twice.) This yields Equation (D.1), as required.

## D.2 Arrival rates $\lambda_k$

We show that the arrival rate of customers from group  $(i, j)$  to node  $k$  is

$$\lambda_k^{(i,j)} = \omega_{ikj} \lambda_{0i}^{(i,j)} = \frac{\omega_{ikj} T_{ij}}{\tau}, \quad (\text{D.4})$$

where

$$\omega_{ikj} = \frac{|\mathcal{S}_k|}{S} \quad (\text{D.5})$$

is fraction of shortest paths from  $i$  to  $j$  that traverse  $k$ . (As in Section D.1,  $S$  is the number of shortest paths from  $i$  to  $j$  and  $\mathcal{S}_k$  is the number of shortest paths from  $i$  to  $j$  that visit  $k$ .) It suffices to show that Equation (D.4) satisfies the traffic equations (see Equation (2.52)). That is, we seek to show that  $\lambda_k^{(i,j)}$  satisfies

$$\lambda_l^{(i,j)} = \lambda_{0l}^{(i,j)} + \sum_{k=1}^n \lambda_k^{(i,j)} P_{kl}^{(i,j)}, \quad \text{for all } l = 1, \dots, n, \quad (\text{D.6})$$

where the external arrival rate  $\lambda_{0l}^{(i,j)}$  is

$$\lambda_{0l}^{(i,j)} = \begin{cases} T_{ij}/\tau, & \text{if } l = i, \\ 0, & \text{otherwise,} \end{cases} \quad (\text{D.7})$$

and the transition matrix  $\mathbf{P}^{(i,j)}$  is given by Equation (D.1). (Note that Equation (D.4) corresponds to Equation (6.3) in Chapter 6.)

For  $k \neq j$ , the number of shortest paths that traverse  $k$  is equal to the number of shortest paths that traverse an edge  $(k, l)$  for some node  $l$ . In other words, for  $k \neq j$ , we have that

$$|\mathcal{S}_k| = \sum_{l=1}^n s_{kl}. \quad (\text{D.8})$$

Substituting Equation (D.8) into Equation (D.5) yields

$$\omega_{ikj} = \frac{\sum_{l=1}^n s_{kl}}{S}, \quad (\text{D.9})$$

when  $k \neq j$ . Similarly, when  $l \neq i$ , the number of shortest paths that traverse  $l$  is equal to the number of shortest paths that traverse an edge  $(k, l)$  for some node  $k$ , so

$$|\mathcal{S}_l| = \sum_{k=1}^n s_{kl} \quad (\text{D.10})$$

and

$$\omega_{ilj} = \frac{\sum_{k=1}^n s_{kl}}{S}. \quad (\text{D.11})$$

When  $l = i$ , Equation (D.6) becomes

$$\begin{aligned} \lambda_i^{(i,j)} &= T_{ij}/\tau + 0 \\ &= \frac{\omega_{iij} T_{ij}}{\tau}, \end{aligned} \quad (\text{D.12})$$

as required, because  $P_{ki} = 0$  for all  $k$  and  $\omega_{iij} = 1$  (because every shortest path from  $i$  to  $j$  traverses  $i$ ).

When  $l \neq i$ , Equation (D.6) becomes

$$\lambda_l^{(i,j)} = \sum_{k: \sum_{l'} s_{kl'} > 0} \frac{\omega_{ikj} T_{ij}}{\tau} \frac{s_{kl}}{\sum_{l'=1}^n s_{kl'}}. \quad (\text{D.13})$$

The sum in Equation (D.13) is over all nodes  $k$  such that  $\sum_{l'} s_{kl'} > 0$ . Note that  $j$  is not among these nodes, because  $\sum_{l'} s_{jl'} = 0$  (all shortest paths end at  $j$  and do not traverse an out-edge of  $j$ ). Substituting Equation (D.9) (noting that  $k \neq j$ ) into Equation (D.13), we obtain

$$\lambda_l^{(i,j)} = \sum_{k: \sum_{l'} s_{kl'} > 0} \frac{T_{ij}}{\tau} \frac{\sum_{l'} s_{kl'}}{S} \frac{s_{kl}}{\sum_{l'=1}^n s_{kl'}} \quad (\text{D.14})$$

$$= \frac{T_{ij}}{\tau} \sum_{k: \sum_{l'} s_{kl'} > 0} \frac{s_{kl}}{S}. \quad (\text{D.15})$$

Note that  $s_{kl} = 0$  if  $\sum_{l'=1}^n s_{kl'} = 0$ , so the right-hand side of Equation (D.15) is unchanged when we change the sum to a sum over all nodes  $k$ . Equation (D.15) thus becomes

$$\lambda_l^{(i,j)} = \frac{T_{ij}}{\tau} \sum_{k=1}^n \frac{s_{kl}}{S} . \quad (\text{D.16})$$

Finally, we use Equation (D.11) to obtain

$$\lambda_l^{(i,j)} = \frac{T_{ij}}{\tau} \omega_{ilj} , \quad (\text{D.17})$$

as required.

Therefore, Equation (D.4) satisfies the traffic equations (D.1), as required.

# Bibliography

- [1] S. Akwawua and J. A. Pooler. “An intervening opportunities model of US interstate migration flows”. *Geography Research Forum*. Vol. 20. 2000, pp. 33–51.
- [2] S. Akwawua and J. A. Pooler. “The development of an intervening opportunities model with spatial dominance effects”. *Journal of Geographical Systems* 3.1 (2001), pp. 69–86.
- [3] R. Albert and A.-L. Barabási. “Statistical mechanics of complex networks”. *Rev. Mod. Phys.* 74.1 (2002), pp. 47–97.
- [4] J. E. Anderson. “A theoretical foundation for the gravity equation”. *The American Economic Review* 69.1 (1979), pp. 106–116.
- [5] T. R. Anderson. “Intermetropolitan migration: a comparison of the hypotheses of Zipf and Stouffer”. *American Sociological Review* 20.3 (1955), pp. 287–291.
- [6] A. Arenas, A. Díaz-Guilera, and R. Guimerà. “Communication in Networks with Hierarchical Branching”. *Phys. Rev. Lett.* 86.14 (2001), pp. 3196–3199.
- [7] H. Barbosa et al. “Human mobility: Models and applications”. *Physics Reports* 734 (2018), pp. 1–74.
- [8] M. Barthelemy. *Morphogenesis of Spatial Networks*. Cham: Springer International Publishing, 2018.
- [9] M. S. Bartlett. “An inverse matrix adjustment arising in discriminant analysis”. *The Annals of Mathematical Statistics* 22.1 (1951), pp. 107–111.
- [10] F. Baskett et al. “Open, closed, and mixed networks of queues with different classes of customers”. *Journal of the ACM (JACM)* 22.2 (1975), pp. 248–260.
- [11] J. H. Bergstrand. “The gravity equation in international trade: some microeconomic foundations and empirical evidence”. *The Review of Economics and Statistics* 67.3 (1985), pp. 474–481.
- [12] G. Bolch et al. *Queueing networks and Markov chains: modeling and performance evaluation with computer science applications*. John Wiley & Sons, 2006.
- [13] B. Bollobás and B. Béla. *Random Graphs*. 73. Cambridge University Press, 2001.
- [14] P. Boros et al. “Modeling supermarket re-layout from the owner’s perspective”. *Annals of Operations Research* 238.1-2 (2016), pp. 27–40.
- [15] U. Brandes. “A faster algorithm for betweenness centrality”. *Journal of Mathematical Sociology* 25.2 (2001), pp. 163–177.
- [16] H. C. Carey. *Principles of Social Science*. Vol. 3. JB Lippincott & Company, 1867.

- [17] S. Chen et al. “Traffic dynamics on complex networks: a survey”. *Mathematical Problems in Engineering* 2012 (2011).
- [18] F. Chung and L. Lu. “Connected components in random graphs with given expected degree sequences”. *Annals of Combinatorics* 6.2 (2002), pp. 125–145.
- [19] T. P. Davies et al. “A mathematical model of the London riots and their policing”. *Scientific Reports* 3 (2013), p. 1303.
- [20] W. E. Deming and F. F. Stephan. “On a least squares adjustment of a sampled frequency table when the expected marginal totals are known”. *The Annals of Mathematical Statistics* 11.4 (1940), pp. 427–444.
- [21] L. Donetti, P. I. Hurtado, and M. A. Munoz. “Entangled networks, synchronization, and optimal network topology”. *Physical Review Letters* 95.18, 188701 (2005).
- [22] P. Echenique, J. Gómez-Gardenes, and Y. Moreno. “Dynamics of jamming transitions in complex networks”. *EPL (Europhysics Letters)* 71.2 (2005), p. 325.
- [23] P. Echenique, J. Gómez-Gardeñes, and Y. Moreno. “Improved routing strategies for Internet traffic delivery”. *Physical Review E* 70.5, 056105 (2004).
- [24] Z. Eisler and J. Kertész. “Random walks on complex networks with inhomogeneous impact”. *Physical Review E* 71.5, 057104 (2005).
- [25] P. Erdős and A. Rényi. “On random graphs I”. *Publ. Math. Debrecen* 6 (1959), pp. 290–297.
- [26] S. Erlander and N. F. Stewart. *The Gravity Model in Transportation Analysis: Theory and Extensions*. Vol. 3. VSP, 1990.
- [27] J. U. Farley and L. W. Ring. “A stochastic model of supermarket traffic flow”. *Operations Research* 14.4 (1966), pp. 555–567.
- [28] B. K. Fosdick et al. “Configuring random graph models with fixed degree sequences”. *SIAM Review* 60.2 (2018), pp. 315–355.
- [29] A. S. Fotheringham. “A new set of spatial-interaction models: the theory of competing destinations”. *Environment and Planning A: Economy and Space* 15.1 (1983), pp. 15–36.
- [30] L. C. Freeman. “A set of measures of centrality based on betweenness”. *Sociometry* 40.1 (1977), pp. 35–41.
- [31] F. Gargiulo et al. “Commuting network models: Getting the essentials”. *Journal of Artificial Societies and Social Simulation* 15.2 (2012), p. 6.
- [32] E. Gelenbe et al. *Introduction to Queueing Networks*. 2nd edition. Vol. 2. John Wiley & Sons, Inc., 1998.
- [33] J. Gil et al. “The differentiating behaviour of shoppers: clustering of individual movement traces in a supermarket”. *Proceedings of the 7th International Space Syntax Symposium*. Royal Institute of Technology (KTH), 2009.
- [34] P. Grindrod. *Mathematical Underpinnings of Analytics: Theory and Applications*. Oxford University Press, 2014.
- [35] R. Guimerà et al. “Optimal network topologies for local search with congestion”. *Physical Review Letters* 89.24, 248701 (2002).

- [36] M. Gutiérrez-Roig et al. “Active and reactive behaviour in human mobility: the influence of attraction points on pedestrians”. *Royal Society Open Science* 3.7, 160177 (2016).
- [37] D. Helbing and P. Molnar. “Social force model for pedestrian dynamics”. *Physical Review E* 51.5, 4282 (1995).
- [38] B. Hillier et al. “Metric and topo-geometric properties of urban street networks: some convergences, divergences and new results”. *Journal of Space Syntax Studies* (2009).
- [39] W. Huang and T. W. Chow. “Effective strategy of adding nodes and links for maximizing the traffic capacity of scale-free network”. *Chaos: An Interdisciplinary Journal of Nonlinear Science* 20.3, 033123 (2010).
- [40] D. L. Huff. “Defining and estimating a trading area”. *Journal of Marketing* 28.3 (1964), pp. 34–38.
- [41] S. K. Hui, E. T. Bradlow, and P. S. Fader. “Testing behavioral hypotheses using an integrated model of grocery store shopping path and purchase behavior”. *Journal of Consumer Research* 36.3 (2009), pp. 478–493.
- [42] S. K. Hui, P. S. Fader, and E. T. Bradlow. “Research note—The traveling salesman goes shopping: The systematic deviations of grocery paths from TSP optimality”. *Marketing Science* 28.3 (2009), pp. 566–572.
- [43] S. K. Hui et al. “The effect of in-store travel distance on unplanned spending: Applications to mobile promotion strategies”. *Journal of Marketing* 77.2 (2013), pp. 1–16.
- [44] T. Inamoto, A. Ohno, and H. Murao. “An Approach of Vectorizing Shopping Paths Sensed by RFID Tags to Classify Retail Customers and Its Application with Principal Component Regression”. *Electronics and Communications in Japan* 97.7 (2014), pp. 63–72.
- [45] J. R. Jackson. “Networks of waiting lines”. *Operations Research* 5.4 (1957), pp. 518–521.
- [46] E. Jones, T. Oliphant, P. Peterson, et al. *SciPy: Open source scientific tools for Python*. [Online; accessed 2019-08-06]. 2001–.
- [47] I.-C. Jung and Y. S. Kwon. “Grocery customer behavior analysis using RFID-based shopping paths data”. *World Academy of Science, Engineering and Technology* 59 (2011), pp. 1404–1408.
- [48] W.-S. Jung, F. Wang, and H. E. Stanley. “Gravity model in the Korean highway”. *EPL (Europhysics Letters)* 81.4 (2008), p. 48005.
- [49] P. Kaluza et al. “The complex network of global cargo ship movements”. *Journal of the Royal Society Interface* 7.48 (2010), pp. 1093–1103.
- [50] F. Kelly and E. Yudovina. *Stochastic Networks*. Cambridge University Press, 2014.
- [51] F. P. Kelly. *Reversibility and Stochastic Networks*. Cambridge University Press, 2011.
- [52] J. H. Kim and V. H. Vu. “Generating random regular graphs”. *Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing*. ACM. 2003, pp. 213–222.

- [53] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. “Optimization by simulated annealing”. *Science* 220.4598 (1983), pp. 671–680.
- [54] N. Koizumi, E. Kuno, and T. E. Smith. “Modeling patient flows using a queuing network with blocking”. *Health Care Management Science* 8.1 (2005), pp. 49–60.
- [55] G. Krings et al. “Urban gravity: A model for inter-city telecommunication flows”. *Journal of Statistical Mechanics: Theory and Experiment* 2009.07, L07003 (2009).
- [56] B. Kunwar, F. Simini, and A. Johansson. “Large scale pedestrian evacuation modeling framework using volunteered geographical information”. *Transportation Research Procedia* 2 (2014), pp. 813–818.
- [57] J. S. Larson, E. T. Bradlow, and P. S. Fader. “An exploratory look at supermarket shopping paths”. *International Journal of Research in Marketing* 22.4 (2005), pp. 395–414.
- [58] S. H. Lee et al. “Matchmaker, matchmaker, make me a match: Migration of populations via marriages in the past”. *Physical Review X* 4.4, 041009 (2014).
- [59] M. Lenormand, A. Bassolas, and J. J. Ramasco. “Systematic comparison of trip distribution laws and models”. *Journal of Transport Geography* 51 (2016), pp. 158–169.
- [60] M. Lenormand et al. “A universal model of commuting networks”. *PLoS One* 7.10, e45985 (2012).
- [61] J. Leskovec, A. Rajaraman, and J. D. Ullman. *Mining of massive datasets*. Cambridge University Press, 2014.
- [62] J. Liang et al. “Peeking into the future: Predicting future person activities and locations in videos”. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 5725–5734.
- [63] X. Liang et al. “Unraveling the origin of exponential law in intra-urban human mobility”. *Scientific Reports* 3 (2013), p. 2983.
- [64] X. Ling et al. “Traffic of packets with non-homogeneously selected destinations in scale-free network”. *Physica A: Statistical Mechanics and its Applications* 387.18 (2008), pp. 4709–4715.
- [65] J. D. Little. “A proof for the queuing formula:  $L = \lambda W$ ”. *Operations Research* 9.3 (1961), pp. 383–387.
- [66] E. Liu and X. Yan. “New parameter-free mobility model: Opportunity priority selection model”. *Physica A: Statistical Mechanics and its Applications* 526, 121023 (2019).
- [67] Z. Liu et al. “Method to enhance traffic capacity for scale-free networks”. *Physical Review E* 76.3, 037101 (2007).
- [68] G. G. Løvås. “Modeling and simulation of pedestrian traffic flow”. *Transportation Research Part B: Methodological* 28.6 (1994), pp. 429–443.
- [69] F. J. Massey Jr. “The Kolmogorov-Smirnov test for goodness of fit”. *Journal of the American statistical Association* 46.253 (1951), pp. 68–78.
- [70] A. P. Masucci et al. “Gravity versus radiation models: On the importance of scale and heterogeneity in commuting flows”. *Physical Review E* 88.2, 022812 (2013).



- [71] N. Masuda, M. A. Porter, and R. Lambiotte. “Random walks and diffusion on networks”. *Physics Reports* 716–717 (2017), pp. 1–58.
- [72] G. McNeill, J. Bright, and S. A. Hale. “Estimating local commuting patterns from geolocated Twitter data”. *EPJ Data Science* 6.1 (2017), p. 24.
- [73] A. J. Miller. “A queueing model for road traffic flow”. *Journal of the Royal Statistical Society: Series B (Methodological)* 23.1 (1961), pp. 64–75.
- [74] D. H. Mitchell and J. M. Smith. “Topological network design of pedestrian networks”. *Transportation Research Part B: Methodological* 35.2 (2001), pp. 107–135.
- [75] A. E. Motter. “Cascade control and defense in complex networks”. *Physical Review Letters* 93.9, 098701 (2004).
- [76] G Mukherjee and S. Manna. “Phase transition in a directed traffic flow network”. *Physical Review E* 71.6, 066108 (2005).
- [77] J. A. Nelder and R. W. Wedderburn. “Generalized linear models”. *Journal of the Royal Statistical Society: Series A (General)* 135.3 (1972), pp. 370–384.
- [78] M. E. J. Newman. *Networks*. 2nd edition. Oxford University Press, 2018.
- [79] T. Ohira and R. Sawatari. “Phase transition in a computer network traffic model”. *Physical Review E* 58.1, 193 (1998).
- [80] T. M. Oshan. “A primer for working with the Spatial Interaction modeling (SpInt) module in the python spatial analysis library (PySAL)”. *REGION* 3.2 (2016), pp. 11–23.
- [81] M. Penrose. *Random Geometric Graphs*. Vol. 5. Oxford University Press, 2003.
- [82] D. Piovani et al. “Measuring accessibility using gravity and radiation models”. *Royal Society Open Science* 5.9, 171668 (2018).
- [83] M. A. Porter and J. P. Gleeson. “Dynamical systems on networks: A tutorial”. *Frontiers in Applied Dynamical Systems: Reviews and Tutorials* 4 (2016).
- [84] F. Pukelsheim. “Biproportional scaling of matrices and the iterative proportional fitting procedure”. *Annals of Operations Research* 215.1 (2014), pp. 269–283.
- [85] S. Putrevu and B. T. Ratchford. “A model of search behavior with an application to grocery shopping”. *Journal of Retailing* 73.4 (1998), pp. 463–486.
- [86] W. J. Reilly. *The Law of Retail Gravitation*. WJ Reilly, 1931.
- [87] E. R. Ruiter. “Toward a better understanding of the intervening opportunities model”. *Transportation Research* 1.1 (1967), pp. 47–56.
- [88] M. T. Schaub et al. “Structure of complex networks: Quantifying edge-to-edge relations by failure-induced flow redistribution”. *Network Science* 2.01 (2014), pp. 66–89.
- [89] M. Schneider. “Gravity models and trip distribution theory”. *Papers in Regional Science* 5.1 (1959), pp. 51–56.
- [90] A. Sim et al. “Great cities look small”. *Journal of The Royal Society Interface* 12.109, 20150315 (2015).
- [91] F. Simini, A. Maritan, and Z. Nédá. “Human mobility in a continuum approach”. *PLoS One* 8.3, e60069 (2013).

- [92] F. Simini et al. “A universal model for mobility and migration patterns”. *Nature* 484.7392 (2012), pp. 96–100.
- [93] H. Sorensen et al. “Fundamental patterns of in-store shopper behavior”. *Journal of Retailing and Consumer Services* 37 (2017), pp. 182–194.
- [94] T. Sørensen. “A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons”. *Biologiske Skrifter* 5 (1948), pp. 1–34.
- [95] S. A. Stouffer. “Intervening opportunities: a theory relating mobility and distance”. *American Sociological Review* 5.6 (1940), pp. 845–867.
- [96] A. J. Tatem et al. “The use of mobile phone data for the estimation of the travel patterns and imported *Plasmodium falciparum* rates among Zanzibar residents”. *Malaria Journal* 8.1 (2009), p. 287.
- [97] H. M. Taylor and S. Karlin. *An Introduction to Stochastic Modeling*. Academic Press, 2014.
- [98] W.-X. Wang et al. “Traffic dynamics based on local routing protocol on a scale-free network”. *Physical Review E* 73.2, 026111 (2006).
- [99] D. J. Watts and S. H. Strogatz. “Collective dynamics of ‘small-world’ networks”. *Nature* 393.6684 (1998), p. 440.
- [100] A. Wesolowski et al. “Impact of human mobility on the emergence of dengue epidemics in Pakistan”. *Proceedings of the National Academy of Sciences* 112.38 (2015), pp. 11887–11892.
- [101] A. G. Wilson. “A family of spatial interaction models, and associated developments”. *Environment and Planning A* 3.1 (1971), pp. 1–32.
- [102] A. G. Wilson. *Entropy in Urban and Regional Modelling*. Pion Ltd, 1970.
- [103] Z.-X. Wu et al. “Improved routing strategies for data traffic in scale-free networks”. *Journal of Statistical Mechanics: Theory and Experiment* 2008.11, P11002 (2008).
- [104] P. Yan. “Spatial-temporal data analytics and consumer shopping behavior modeling” (2010).
- [105] P. Yan and D. D. Zeng. “Clustering customer shopping trips with network structure”. *International Conference on Information Systems (ICIS)*. 2008.
- [106] P. Yan and D. D. Zeng. “In-store shopping activity modeling based on dynamic Bayesian networks”. *19th Workshop on Information Technologies and Systems*. 2009.
- [107] Y. Yang et al. “Limits of predictability in commuting flows in the absence of data for calibration”. *Scientific Reports* 4 (2014), p. 5662.
- [108] Y. Yuan et al. “Decentralised minimum-time consensus”. *Automatica* 49.5 (2013), pp. 1227–1235.
- [109] S. J. Yuhaski and J. M. Smith. “Modeling circulation systems in buildings using state dependent queueing models”. *Queueing Systems* 4.4 (1989), pp. 319–338.
- [110] G.-Q. Zhang, D. Wang, and G.-J. Li. “Enhancing the transmission efficiency by edge deletion in scale-free networks”. *Physical Review E* 76.1, 017101 (2007).

- [111] H. Zhang et al. “An adaptive routing strategy for packet delivery in complex networks”. *Physics Letters A* 364.3 (2007), pp. 177–182.
- [112] L. Zhao et al. “Onset of traffic congestion in complex networks”. *Physical Review E* 71.2, 026125 (2005).
- [113] G. K. Zipf. “The  $P_1P_2/D$  hypothesis: On the intercity movement of persons”. *American Sociological Review* 11.6 (1946), pp. 677–686.