DESIGNING NETWORKS, ROUTING FLEETS, AND TRYING TO FIND PARKING

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Juan Carlos Martínez Mori

August 2023

DESIGNING NETWORKS, ROUTING FLEETS, AND TRYING TO FIND PARKING

Juan Carlos Martínez Mori, Ph.D.

Cornell University 2023

This dissertation contains a bevy of problems in discrete mathematics; spanning optimization, real-world applications, and combinatorics. It presents—roughly in chronological order—my works and mathematics journey as a graduate student. Its title "Designing Networks, Routing Fleets, and Trying to Find Parking" reflects the kinds of problems I cover: *network design*, *vehicle routing*, and the combinatorics of *parking*.

BIOGRAPHICAL SKETCH

Juan Carlos Martínez Mori was born on July 29, 1994 in Guayaquil, Ecuador. In 2013, he graduated high school from the Colegio Americano de Guayaquil with an International Baccalaureate Diploma. In the same year, he received a generous scholarship from the Government of Ecuador to pursue higher education at the University of Illinois at Urbana-Champaign. In 2017, he graduated from Illinois with a Bachelor of Science in Civil Engineering and a minor in Computer Science. In the same year, he began graduate school at Cornell University. In 2019, he transferred to the Center for Applied Mathematics (CAM) at Cornell University. This document is necessary for the degree of Doctor of Philosophy in Applied Mathematics.

Upon completion of his doctoral degree, he will join the Simons Laufer Mathematical Sciences Institute (formerly MSRI) as a postdoctoral fellow for the Fall 2023 program on *Algorithms, Fairness, and Equity*. Following that, he will start a two-year postdoctoral placement as a *Schmidt Science Fellow*.

This document is dedicated to my grandfather Gilberto Humberto Martínez Carrión (1932–); to the memory of my grandmothers Rosa Piedad Lucero Chávez (1938–2021) and Cuty Neira García (1934–2019); and to the memory of my great grandmother Piedad Eloísa Chávez Hidalgo (1918–2015).

ACKNOWLEDGEMENTS

No estás deprimido, estás distraído.
Por eso crees que perdiste algo, lo que
es imposible porque todo te fue dado.
No hiciste ni un sólo pelo de tu
cabeza, por lo tanto no puedes ser
dueño de nada.

Facundo Cabral

This dissertation is the result of a collective effort, certainly on my part, but equally importantly from my mentors, mentees, colleagues, friends, and family.

I thank Samitha Samaranayake for supporting me in each and every step of this journey. He steered me toward true learning, and taught me how to be an independent researcher. Importantly, he had my back throughout this process. Looking back, my goals for graduate school were set to be an uphill battle from the get-go. I would not have accomplished any of those goals without him.

I thank David Shmoys for the mathematics I learned through him, for the career advice, and for being a tireless champion for my success (and that of countless graduate students).

I thank Bobby Kleinberg for seeing in me what I did not see in myself, and for doing something about it. One day years ago he raised the idea of me becoming a mathematician—that was a pivotal for me.

I thank Pamela Harris for the mathematics I learned through her, and for introducing me to a caring community of research mathematicians. She reinforced my identity and sense of belonging, and refined my long-term career goals.

I also thank a broader collection of individuals who made a difference at various stages of my career (a strict subset of them is listed here in alphabetical order): Geovanny Alvarado Villa, Mary Armijos, William Barbour, José Barreiros, Jimy Cai, Francisco Canales, Miguel Chevasco, Ning Duan, Carlosluis Duchicela, Jennifer Elder, Mallory Gaspard, Rebecca Garcia, Paul Gharzouzi, José Guadamud Bravo, Adrian Hernandez, Ronnie Houck, Hongyi Jiang, Andony Landívar Macías, Yanning Li, Brian Kamau, Patrick Kastner, Youngseo Kim, Nick Klein, Jan Kretschmann, Lucy Martinez, Gabriel Martínez Vera, Ayan Mukhopadhyay, Shriya Nagpal, Gokul Nair, Shawn Ong, Misha Padidar, Sophie Pavia, Artyom Pavlichenko, Sofía Recalde, Tomás Rosetti, Walter Romero, Max Ruth, Andrés Serrano, Tito Sempértegui, M. Grazia Speranza, Raphael Stern, Miguel Suarez Medina, Roger Tian, Alexandra Warton, Dwight Anderson Williams II, Anke van Zuylen, Daniel Work, Fangyu Wu, and Matthew Zalesak. I particularly thank Geovanny Alvarado Villa for steering me to be self-taught as a high school student, and Daniel Work and the members of his research group back at Illinois for getting me started in an academic career.

Lastly, I thank the most important people in my life: Nicole Sempértegui Plaza, my parents David Martínez Neira and Tania Mori Lucero, and my brother Rafael Martínez Mori.

This dissertation is based on works supported fully or in part by various agencies. These include the National Science Foundation (NSF) through grants DMS-1839346, CNS-1952011, and CMMI-2144127 (awarded to S. Samaranayake); the Federal Highway Administration (FHWA) through Dwight David Eisenhower Transportation Fellowship Program (DDETFP) grants 693JJ31845041, 693JJ31945044, and 693JJ32145020; the Institute for Computational and Experimental Research in Mathematics (ICERM), which is supported by the NSF through grant DMS-1929284; the Institute for Pure and Applied Mathematics (IPAM), which is supported by the NSF through grant DMS-

1925919; and the Simons Laufer Mathematical Sciences Institute (SLMath, formerly MSRI), which is supported by the NSF through grant DMS-1928930.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

This dissertation contains a bevy of problems in discrete mathematics; spanning optimization, real-world applications, and combinatorics. It is divided into six chapters which summarize—roughly in chronological order—my works and mathematics journey as a graduate student. Its title "Designing Networks, Routing Fleets, and Trying to Find Parking" reflects the kinds of problems I cover: *network design*, *vehicle routing*, and the combinatorics of *parking*.

All of the results presented herein are collaborative. To acknowledge this, I preface each chapter by naming my coauthors and providing any corresponding references. In each chapter, I include a brief introduction. Introductions are written using singular first-person pronouns (i.e., using the word "I") because in them I merely describe and contextualize the content that follows. All other sections are written using plural first-person pronouns (i.e., using the word "we").

This document is organized as follows. In Chapter 2, I take a data-driven approach to exhibit a structural property of road networks, namely bounded asymmetry, with immediate algorithmic implications for tour problems. In Chapter 3, I introduce a heuristic for the Steiner tree problem based on the bidirected cut relaxation. I leave obtaining an approximation guarantee for this heuristic (if one exists) as a problem for the future. In Chapter 4, I design an randomized rounding algorithm for an assignment problem arising in high-capacity ridesharing applications of online vehicle routing. In Chapter 5, I develop a framework with which to quantify the value of integrating transit with on-demand mobility. To do this, I introduce the dynamicity gap, a general concept that quantifies the attainable benefit of allowing (but not requiring) dynamic components in the response strategy to a multi-stage optimization problem. I build on theory from stochastic programming to estimate the dynamicity gap as a function of problem input parameters. In Chapter 6, I provide efficient polyhedral methods to account for new scheduling concerns in infrastructure planning (e.g., network planning), namely the order in which to carry out construction so as to maximize the benefit reaped over time, including during the construction phase. The main technical ingredient leverages a known linear inequality description of the permutahedron. In Chapter 7, I define parking functions, which are well-known combinatorial objects, and outline various results around related "parking objects" (i.e., variants, specializations, and generalizations of parking functions).

Let's go do all the math!

CHAPTER 2
A STRUCTURAL PROPERTY OF ROAD NETWORKS

This chapter is based on work with Samitha Samaranayake [55].

2.1 Introduction

Road networks fundamentally shape the physical exchange of goods, services, and information. Consequently, they are a classical setting for the applications of graph problems. In this chapter, I propose a graph property with immediate algorithmic implications (particularly for tour problems) and, through a data-driven approach, exhibit its prevalence in road networks.

2.2 Asymmetry Factor

Let $G = (V, A)$ be a complete directed graph with costs $c : A \to \mathbb{R}_{\geq 0}$ on the arcs. We assume c is an asymmetric metric, which implies $c_{uw} \leq c_{uv} + c_{vw}$ for all triples $u, v, w \in V$. Consider the following definitions.

Definition 2.2.1 (Pairwise Asymmetry Factor)**.** *Given $u, v \in V$ with $u \neq v$, let*

$$\Delta_{u,v} \coloneqq \max\left\{\frac{c_{uv}}{c_{vu}}, \frac{c_{vu}}{c_{uv}}\right\}.$$

Definition 2.2.2 (Asymmetry Factor)**.**

$$\Delta_G \coloneqq \max_{u,v \in V : u \neq v} \Delta_{u,v}.$$

In this way, Δ_G measures the worst-case pairwise "asymmetry" in G. In [55] we show the following.

Theorem 2.2.3. *Let $c' : A \to \mathbb{R}_{>0}$ be the symmetrized version of c where $c'_{uv} = \max\{c_{uv}, c_{vu}\}$. Then, c' is a metric and $c(F) \leq c'(F) \leq \Delta_G \cdot c(F)$ for every $F \subseteq A$.*

Corollary 2.2.4. *Let $F, H, I \subseteq A$ and $\alpha \in \mathbb{R}_{\geq 1}$. If $c'(F) \leq \alpha \cdot c'(H)$ and $c'(H) \leq c'(I)$, then $c(F) \leq \alpha \cdot \Delta_G \cdot c(I)$.*

Suppose we are given a graph problem in which we must find a "suitable" subset $F \subseteq A$ of minimum cost, given by $c(F) := \sum_{a \in F} c_a$. For example, we might require F to be a Hamiltonian cycle. This problem is known as the Asymmetric Traveling Salesperson Problem (ATSP), which is NP-hard. Now, suppose we must (again) find a "suitable" subset $F \subseteq A$ of minimum cost, this time given by $c'(F) := \sum_{a \in F} c'_a$. For example, we might (again) require F to be a Hamiltonian cycle. However, now that c' is a metric, this problem is known as the metric Traveling Salesperson Problem (TSP), which is (again) NP-hard.

While both the ATSP and the metric TSP are NP-hard, there is a sense (albeit informal) in which metric TSP is "easier." Indeed, there is a well-known 3/2-approximation algorithm for the metric TSP, due independently to Christofides [21] and Serdyukov [69]. This algorithm is combinatorial and straightforward to implement. Recently, Karlin et al. [42] obtained a $(3/2 - \epsilon)$-approximation algorithm for some $\epsilon > 10^{-36}$.

Conversely, obtaining a constant factor approximation algorithm for the ATSP was a long-standing open problem. For various decades, only algorithms with logarithmic or polylogarithmic approximation guarantees were known (e.g., [30, 7]). More recently Svensson et al. [73] settled this problem by introducing a 506-approximation algorithm. Unfortunately, while mathematically sophisticated, known approximation algorithms for the ATSP generally do not enjoy the implementation simplicity of the Christofides-Serdyukov algorithm for the metric TSP.

Corollary 2.2.4 bridges the metric TSP and the ATSP by quantifying the loss in approximation incurred by, instead of dealing with the ATSP directly, dealing with an appropriate instance of the metric TSP. This loss is parametrized by the asymmetry factor Δ_G.

Theorem 2.2.5. *Let $G = (V, E)$ be a complete directed graph with asymmetric metric costs $c : A \to \mathbb{R}_{\geq 0}$. If there is an α-approximation algorithm for the metric TSP, then there is a $(\alpha \cdot \Delta_G)$-approximation algorithm for the ATSP. In particular, there is a $(3/2 \cdot \Delta_G)$-approximation algorithm via the Christofides-Serdyukov algorithm.*

Therefore, if Δ_G is small (say, $\Delta_G = O(1)$), then there is a simple constant factor approximation algorithm for the ATSP.

2.3 Asymmetry in Road Networks

In light of Theorem 2.2.5, we are interested in graphs with bounded asymmetric factor. In this section, we take a data-driven approach to exhibit the prevalence of this property in road networks.

Let $R = (V_R, A_R)$ be directed graph representing a road network (e.g., of a given city), where roughly speaking V_R represents intersections and A_R represents road segments between intersections. Moreover, consider costs $c_R : A_R \to \mathbb{R}_{\geq 0}$ representing the physical length of a road segment (in the absence of traffic, length may be an appropriate proxy for travel time). This data may be obtained for most cities in the world via the `networkx` package of Boeing [14].

While c_R need not be an asymmetric metric, we may obtain one by considering its metric closure. That is, we consider the complete directed graph $G = (V, A)$, where $V = V_R$, together with costs $c : A \to \mathbb{R}_{\geq 0}$, where c_{uv} is the length of a shortest path from u to v in R with respect to c_R. Note that considering the metric closure is a standard assumption (one without loss of generality [33]) required by algorithms for the ATSP and the metric TSP.

In Figure 2.1, we exhibit the worst-case pairwise asymmetry factor for various cities in the world, each a representative of the typology of cities proposed by Louf and Barthelemy [50]. In particular, we evaluate Δ_G on pairs of nodes subject to a moving high-pass filter on the distance between them. That is, for any given $\ell \in \mathbb{R}_{\geq 0}$, we evaluate

$$\Delta_G(\ell) := \max_{u,v \in G : u \neq v, c_{uv} \geq \ell} \Delta_{uv}.$$

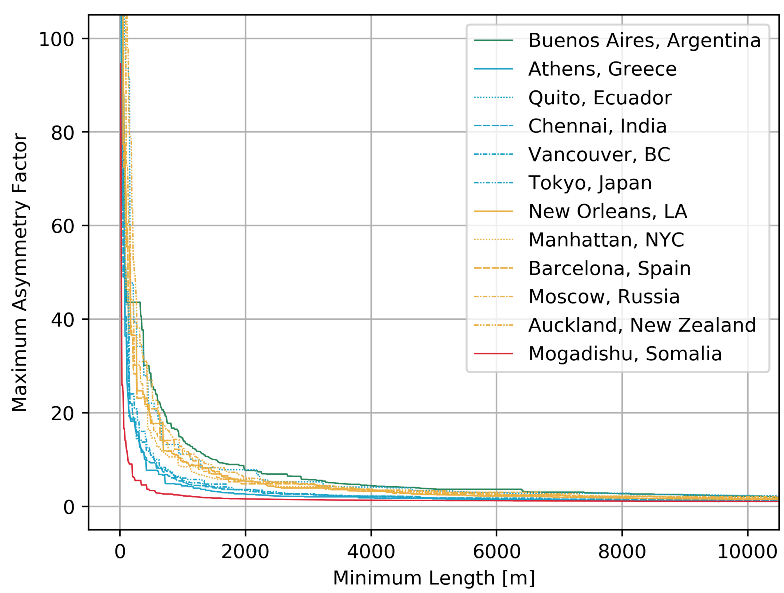
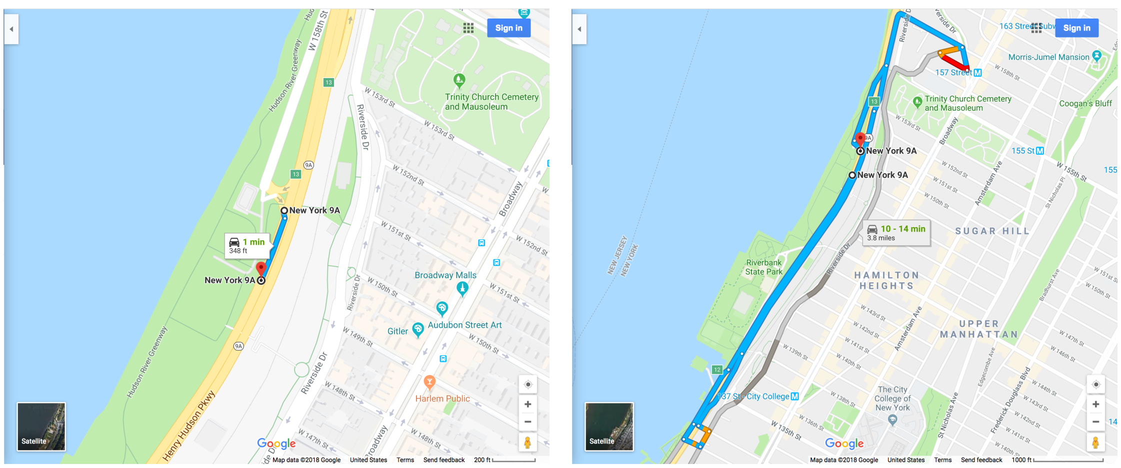


Figure 2.1: Road network asymmetry factors.

Figure 2.1 shows that large pairwise asymmetry factors concentrate on pairs of nodes with small distance between them (where the distance between a pair of nodes $u, v \in V$ is given by $\min\{c_{uv}, c_{vu}\}$). Conversely, pairwise asymmetry factors are small for pairs of nodes with sufficiently large distance between them.

We demonstrate the reason for this in Figure 2.2, which gives a concrete example of the "local" effects that one-directional roads (in this case, a restricted-access freeway) have on the asymmetry factor of a road network. The nodes u and v in the figure are located at latitude-longitude coordinates (40.8313944, −73.9508067) and (40.8310864, −73.950962) respectively, are about 37 meters apart from one another, and yield $\Delta_{\{u,v\}} \approx 214$. Intuitively, as the distance between pairs of nodes increases, routing peculiarities of this kind tend to disappear.



(a) Shortest path from u to v in R.

(b) Shortest path from v to u in R.

Figure 2.2: Example of large pairwise asymmetry factor in a road network. Mapping source: Map data ©2018 Google.

Road networks have one-way roads, and therefore are generally asymmetric. However, Theorem 2.2.5, together with the empirical results summarized in Figure 2.1, provide strong justification to focus on the metric TSP (as opposed to the ATSP) whenever the nodes to be visited by a tour are sufficiently far from one another.

CHAPTER 3

A HEURISTIC FOR THE STEINER TREE PROBLEM

This chapter is based on unpublished work with Samitha Samaranayake and David Shmoys.

3.1 Introduction

In the Steiner tree problem, we are given an undirected graph $G = (V, E)$, a cost function $c : E \to \mathbb{R}_{\geq 0}$, and a set $R \subseteq V$ of *terminal* nodes (the nodes in $V \setminus R$ are called *Steiner* nodes). The goal is to find a minimum cost subgraph of G spanning R. In this chapter, I give a heuristic linear-programming based algorithm for this problem. I leave obtaining an approximation guarantee (if one exists) as a problem for the future.

3.2 Related Work

The Steiner tree problem is NP-hard [43], and it is NP-hard to approximate it within a factor of 96/95 [20]. A simple 2-approximation algorithm finds a minimum spanning tree on the metric closure of R [47]. The same approximation guarantee can be achieved, for example, via iterative rounding [41] or via primal-dual methods [2, 36] using the classical *cut covering* LP relaxation. Byrka et al. [16] give a $\ln(4) + \epsilon < 1.39$-approximation algorithm via iterative rounding using a *hypergraphic* LP relaxation [18]. Unfortunately, hypergraphic LP relaxations are notoriously hard to solve, intuitively because they involve exponentially many variables *and* exponentially many constraints. They are NP-hard to solve exactly [35], but can be approximated within a factor of $1 + \epsilon$ in $n^{\exp(1/\epsilon)}$ time, where $n = |V|$.

One potential way to obtain a good approximation guarantee (i.e., better than 2) through an efficiently solvable LP is to use the *bidirected cut* LP relaxation. The bidirected cut relaxation was introduced by Wong [78] as a generalization of Edmonds' [24] bidirected cut description of the spanning tree polytope. Although this relaxation is believed to be much stronger than the cut covering relaxation, its integrality gap is only known to be somewhere between 6/5 [74] and 2. Closing this gap is a long-standing open problem, although positive results are known for some special cases. When the graph is Steiner claw-free (i.e., there are no Steiner nodes with three or more Steiner neighbors), the bidirected cut relaxation is equivalent to the (generally stronger [62]) hypergraphic relaxations [29], and so its integrality gap is at most $\ln(4)$ [35]. When the graph is quasi-bipartite (i.e., the Steiner nodes form an independent set), the same equivalence implies its integrality gap is somewhere between 8/7 (see Skutella's instance in [45]) and 73/60 [35].

3.3 A Heuristic Based on the Bidirected Cut Relaxation

We present our algorithm in terms of the *subtour elimination* relaxation. This relaxation is one of many equivalent to the bidirected cut relaxation (see [34, 74]). The subtour elimination relaxation minimizes $\sum_{e\in E} c_e x_e$ over the polytope

$$P_{xy}(G) := \left\{ (x,y) \in \mathbb{R}_{\geq 0}^{|E|+|V|} : \begin{array}{rll} \sum_{e\in E} x_e & = \sum_{u\in V} y_u - 1, & \\ \sum_{e\in E[U]} x_e & \leq \sum_{u\in U-v} y_u, & \forall \emptyset \neq U \subseteq V, \forall v \in U \\ y_u & \leq 1, & \forall u \in V \\ y_u & \geq 1, & \forall u \in R \end{array} \right\},$$

where $E[U]$ denotes the edges in the subgraph of G induced by U. Intuitively, y and x indicate the nodes and edges that are selected as part of the solution, respectively. The inequalities $y_u \geq 1$ for $u \in R$ ensure all terminal nodes are selected. The inequalities $\sum_{e\in E[U]} x_e \leq \sum_{u\in U-v} y_u$ for $\emptyset \neq U \subseteq V$ and $v \in U$ prevent the formation of cycles. The equality $\sum_{e\in E} x_e = \sum_{u\in V} y_u - 1$ ensures the selected edges form a tree spanning the selected nodes. A Steiner tree T is associated with the solution $(x,y) \in \{0,1\}^{|E|+|V|}$ where $x_e = 1$ if and only if $e \in T$ and $y_u = 1$ if and only if $u \in V(T)$.

Note that when $R = V$, we recover Edmonds' [25] subtour elimination description of the spanning tree polytope. This observation motivates the following claim, where $G_x = (V_x, E_x)$ is the subgraph of G induced by the support of $x \in \mathbb{R}_{\geq 0}^{|E|}$ and $P_{\text{span}}(G_x)$ denotes the spanning tree polytope of G_x.

Proposition 3.3.1. *Let $(x,y) \in P_{xy}(G)$. Then, x satisfies all subtour elimination inequalities in the description of $P_{span}(G_x)$.*

Proof. Consider any $\emptyset \neq U \subseteq V_x$ and any $v \in U$. We have

$$\sum_{e\in E_x[U]} x_e = \sum_{e\in E[U]} x_e \leq \sum_{u\in U-v} y_u \leq |U - v| = |U| - 1,$$

where the first equality holds since $x_e = 0$ for all $e \in E[U] \setminus E_x[U]$, the first inequality holds since $(x,y) \in P_{xy}(G)$, and the second inequality holds since $y_u \leq 1$ for all $u \in V_x$. □

Proposition 3.3.1 says that if $(x, y) \in P_{xy}(G)$, then x is either in or *almost* in $P_{\text{span}}(G_x)$. In particular, if $x \notin P_{\text{span}}(G_x)$, then the only violated constraint is the equality constraint. It follows that there is some sort of *lifted* version of x that does in fact belong to $P_{\text{span}}(G_x)$. Intuitively, if the lifted version of x is not too far from x itself, then Algorithm 1 is reasonable.

Algorithm 1: Minimum spanning tree of support graph.

1. Find a basic feasible solution (x, y) minimizing $\sum_{e \in E} c_e x_e$ over P_{xy};
2. Let T be a minimum spanning tree of G_x;
3. Remove all pendant edges in T; `// An edge is pendant if its removal does not affect feasibility`
4. **return** T

We formalize our intuition. Consider the following linear program, where $x \in \mathbb{R}^{|E|}_{\geq 0}$ comes from step 1 of Algorithm 1 and is therefore *not* a decision variable.

$$\begin{array}{lrll} \text{minimize} & \alpha^* := \max\limits_{e \in E} \alpha_e & & \\ \text{s.t.} & \sum_{e \in E_x} \alpha_e x_e & = |V_x| - 1, & \\ & \sum_{e \in E[U]} \alpha_e x_e & \leq |U| - 1, & \forall \emptyset \neq U \subseteq V_x, \\ & \alpha_e & \geq 1, & \forall e \in E. \end{array} \tag{3.1}$$

This linear program finds a lifting vector $\alpha \in \mathbb{R}^{|E|}_{\geq 1}$ such that $\alpha \circ x := (\alpha_e \cdot x_e)_{e \in E} \in P_{\text{span}}(G_x)$ and $\alpha^* := \max_{e \in E} \alpha_e$ is minimized. We know such a vector exists through Proposition 3.3.1. We thus obtain the following sufficient condition for Algorithm 1 to be a β-approximation algorithm.

Theorem 3.3.2. *If for all instances of the Steiner tree problem $\alpha^* \leq \beta$ holds, then Algorithm 1 is a β-approximation algorithm and the integrality gap of the bidirected cut relaxation is at most β.*

Proof. The algorithm first finds a minimum spanning tree T of G_x. At this stage we have

$$c(T) \le \sum_{e\in E_x} c_e(\alpha_e x_e) \le \sum_{e\in E_x} c_e(\alpha^* x_e) = \alpha^* \sum_{e\in E_x} c_e x_e = \beta \sum_{e\in E_x} c_e x_e \le \beta \sum_{e\in E} c_e x_e \le \beta \cdot \mathrm{OPT},$$

where the first inequality holds since $\alpha \circ x \in P_{\text{span}}(G_x)$. Finally, removing pendant edges in T can only decrease its cost. □

We illustrate our algorithm in Figure 3.1. The example shows that there may be a gap between the performance of the algorithm, the integrality gap of the bidirected cut relaxation, and α^*.

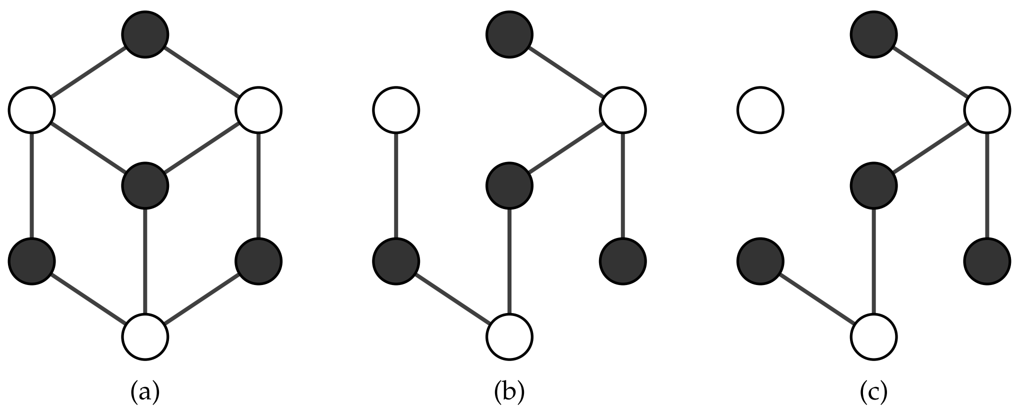


Figure 3.1: (a) Unweighted graph $G = (V, E)$ with terminal nodes R in black and Steiner nodes $V \setminus R$ in white. An optimal solution to the bidirected cut relaxation sets $x_e = 1/2$ for all $e \in E$, and so $\sum_{e\in E} c_e x_e = 9/2$. Note that $G_x = G$ in this case. The lifting vector $\alpha_e = 4/3$ for all $e \in E$ lifts x into $P_{\text{span}}(G_x)$ while minimizing $\alpha^* := \max_{e\in E} \alpha_e = 4/3$. (b) A minimum spanning tree T of G_x, with cost $c(T) = 6$. (c) The tree T after removing all pendant edges, with cost $c(T) = 5$. Note that T is an optimal solution, and so there is a gap between α^* and the actual performance of the algorithm. The integrality gap of the bidirected cut relaxation on this instance is $10/9 < \alpha^*$.

However, note that this condition is only sufficient (in particular, it is plausible that the premise of Theorem 3.3.2 is false in the sense that no finite β, let alone small, exists). Furthermore, to prove good approximation, it might necessary to account for the cost-savings associated with the removal of pendant edges, which is not accounted for in Theorem 3.3.2. For example, based on computational experiments, a class of Steiner tree instances arising from hard set cover instances lead large α^* factors, but this is alleviated once pendant edges are removed. It is not clear how to account for the removal of pendant edges systematically.

There is a crucial subtlety when implementing our algorithm: without proper care, it is possible that $\alpha^* > 1$ even on instances in which the integrality gap of the bidirected cut relaxation is one. To see this, note that although P_{xy} is an extended formulation, our algorithm works on the projected space $P_x := \text{proj}_x(P_{xy})$. A basic feasible solution (x, y) minimizing $\sum_{e \in E} c_e x_e$ over P_{xy} induces a feasible solution x minimizing $\sum_{e \in E} c_e x_e$ over P_x, but x need not be extreme in P_x. If x is not extreme in P_x, then its support (and hence G_x and α^*) is unnecessarily large, as seen in the example in Figure 3.2. We address this subtlety more generally in Figure 3.3.

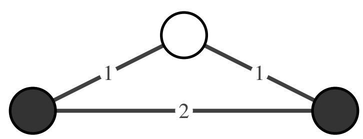


Figure 3.2: Graph $G = (V, E)$ with terminal nodes R in black, Steiner nodes $V \setminus R$ in white, and edge weights as shown. The solution $x_e = 1/2$ for all $e \in E$ is optimal, but not extreme in P_x. Such a solution induces $\alpha^* = 4/3$ even though the integrality gap of the bidirected cut relaxation on this instance is one.

Having addressed the subtlety in Figure 3.3, we pose the following question:

Open Question 3.3.3. *Is $\alpha^* \leq \beta$ for some class of instances of the Steiner tree problem, for some $\beta < 2$?*

Lastly, we note that we could replace steps 2 and 3 of Algorithm 1 with a call to the 2-approximating minimum spanning tree heuristic on the metric closure of R given G_x. However, this seems even more difficult to analyze.

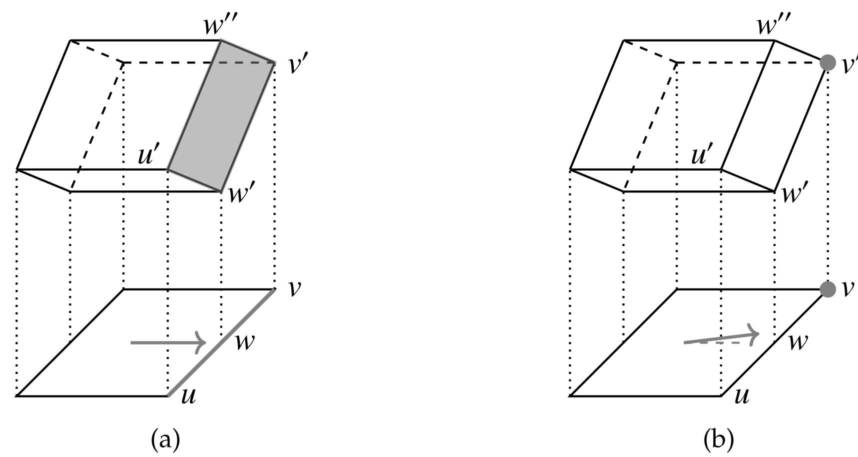


Figure 3.3: (a) Suppose the lower-dimensional facet with vertices u and v is optimal, as indicated by the gray arrow representing the objective function. Then, the higher-dimensional facet with vertices u', v', w' and w'' is optimal. Vertices u' and v' project down to vertices u and v, respectively. However, vertices w' and w'' project down to w, which is not a vertex. (b). We slightly perturb the objective function (e.g., by adding $U(-\epsilon, \epsilon)$ noise to the cost of each edge, for ϵ small enough), as indicated by the slightly perturbed gray arrow, so that *i*) there is a unique optimal solution in the lower-dimensional space, and *ii*) said solution was optimal before the perturbation. Then, any higher-dimensional optimal solution projects down to the unique lower-dimensional optimal solution. In this example, v is the unique lower-dimensional optimal solution.

CHAPTER 4

AN ASSIGNMENT PROBLEM IN ONLINE VEHICLE ROUTING

This chapter is based on work with Samitha Samaranayake [58].

4.1 Introduction

The *request-trip-vehicle* (RTV) assignment problem is at the heart of a popular decomposition strategy for ridesharing applications of "online" vehicle routing. In this framework, assignments are made in batches over time in order to exploit any "shareability" among vehicles and incoming travel requests. See Alonso-Mora et al. [6] for details and [22] for an example of its deployment.

Bei and Zhang [9] show this problem is NP-hard even when no more than two requests can share a vehicle. Therefore, algorithms research around this problem has primarily focused on experimental performance (e.g., [61, 71, 51, 67]). Nevertheless, algorithms with provable performance guarantees have been proposed under a variety of structural assumptions (e.g., [9, 48, 53, 52]).

In this chapter, I consider a natural IP formulation for this problem and propose an LP-based randomized rounding algorithm that, whenever the instance is feasible, leverages mild assumptions to return an assignment whose: *i*) expected cost is at most that of an optimal solution, and *ii*) expected fraction of unassigned requests is at most $1/e$. If trip-vehicle assignment costs are α-approximate, the algorithm pays an additional factor of α in the expected cost. To the best of my knowledge, this is the first algorithm with provable performance guarantees for the RTV assignment problem in its full generality.

4.2 Problem Formulation

In the RTV assignment problem, we are given a set R of travel requests, a set T of candidate trips (i.e., a collection of subsets of R, possibly *all* of them)[1], and a set V of vehicles. Assigning a vehicle $v \in V$ to a trip $t \in T$ has an associated cost $c_{tv} \in \mathbb{R}_{\geq 0}$; this term typically stems from a single-vehicle routing problem (i.e., a tour problem) and represents distance traveled or incurred delays. The problem is to find a minimum cost set of trip-vehicle assignments such that: *i*) each request appears in exactly one trip-vehicle assignment, and *ii*) each vehicle is assigned to at most one trip.

The framework of Alonso-Mora et al. [6] exploits two key structural properties: *i*) there are tight quality of service constraints such as maximum wait time and maximum travel time (in turn distinguishing this class of vehicle routing problems from those arising in logistics), and *ii*) the feasible space is *downward closed*. In particular, a necessary condition for a potential trip-vehicle assignment to be feasible given the quality of service constraints is that all of its sub-trip-vehicle assignments are feasible. This means T itself is downward closed. Together, these properties help prune a priori infeasible trips and vehicle assignments, thereby thwarting the combinatorial explosion. As a result, the following

[1]For technical reasons we require $\emptyset \in T$.

IP formulation is natural even though it is exponentially large in the worst case.

$$\begin{aligned} \text{minimize} \quad & \sum_{t \in T} \sum_{v \in V} c_{tv} x_{tv} \\ \text{subject to} \quad & \\ & \sum_{(t,v) \in T(r) \times V} x_{tv} \geq 1, \qquad \forall r \in R \\ & \sum_{t \in T} x_{tv} \leq 1, \qquad \forall v \in V \\ & x_{tv} \in \{0, 1\}, \quad \forall t \in T, v \in V \end{aligned} \tag{4.1}$$

Here, $T(r)$ is the set of trips that contain request $r \in R$. Similarly, $V(t)$ be the set of vehicles that can serve trip $t \in T$ and $T(v)$ is the set of trips than can be served by vehicle $v \in V$. For ease of presentation, we assume any vehicle can serve any trip. Then, $V(t) = V$ for each $t \in T$ and $T(v) = T$ for each $v \in V$. This assumption does not affect our results.

The objective is to minimize the total cost of trip-vehicle assignments. The first set of constraints ensure each request is served. We make the natural assumption that the cost of a trip-vehicle assignment cannot decrease by including an additional request. That is, we assume the cost function $c : V \times T \to \mathbb{R}_{\geq 0}$ is *monotonic increasing with respect to request inclusion*. Together with the fact that T is downward closed, this allows us to introduce these constraints as covering constraints. The second set of constraints ensure each vehicle is assigned to at most one trip.

4.3 Randomized Rounding

Note that the LP relaxation of (4.1) has exponentially many variables in the worst case. Therefore, it is natural to consider solving it through the ellipsoid method in theory and through a column generation approach in practice. In [58], we argue that the nature of its dual separation problem renders an unfavorable prospect for this approach. The difficulty has to do with a gap in the literature on the complexity of even *approximating* the dual separation problem when the cost function c corresponds to a single-vehicle routing problem (it is already NP-hard in this case), and an existing hardness of approximation result for a closely-related problem [27] (when c corresponds to a network design problem).

As an alternative, we assume T is pre-computed and that $|T| = \text{poly}(|R|, |V|)$. While this might appear to be a strong assumption a priori, we argue that it is not within the context of ridesharing applications. Since vehicles have a fixed seating capacity, say $k > 0$, the size of T is typically $O(n^k)$. This is not always the case since a vehicle that drops off passengers along the way may serve more than k requests. Nevertheless, tight quality of service constraints typically prevent large trips from being feasible (and hence can be excluded from T a priori).

This implies that, if c or an approximation of it is oracle-given (e.g., see the works cited in Chapter 2, particularly [21, 30, 7, 73]), then we can explicitly write and solve the LP relaxation of (4.1). We then leverage mild assumptions to design a simple LP-based randomized rounding algorithm in the style of Raghavan and Thompson [64]. We describe the algorithm as follows.

Recall each vehicle $v \in V$ satisfies $\sum_{t \in T} x_{tv} = 1$ by the LP constraints. We interpret this as a vehicle-specific probability distribution over the trips. Therefore, independently for each vehicle $v \in V$, assign it to a randomly chosen trip, where the probability of assigning it to trip $t \in T$ is given by $0 \leq x_{tv} \leq 1$. Let X be the set of random variables corresponding to this step.

Note that a request $r \in R$ may appear in multiple trip-vehicle assignments. Allow r to pick one arbitrarily. Here we are crucially using the fact that T is downward closed. Further, by the monotonicity of the cost function c with respect to request inclusion, doing this cannot increase the cost of our solution. Define X' analogously to X, except this time it corresponds to the final output after the multiplicity correction step.

We obtain the following.

Theorem 4.3.1. *Suppose we have a feasible instance of the RTV assignment problem with T polynomial-sized. If trip-vehicle costs are oracle-given and monotonic increasing with respect to request inclusion, there is a randomized algorithm such that:*

- *The expected cost of the final solution is at most that of an optimal solution.*
- *The expected fraction of unassigned requests is at most $1/e$ (i.e., less than 36.8% of all requests).*

If trip-vehicle costs are α-approximate, we pay an additional factor of α in the expected cost.

Proof. It is clear that the expected cost of the final RTV assignment X' is at most the objective value of the LP relaxation of (4.1). We now consider the fraction of unassigned requests.

Since r is assigned in X if and only if it is assigned in X', we can focus on the former. Let Y_v be a binary random variable indicating whether vehicle $v \in V$ is assigned, in X, to a trip $t \in T$ such that $r \in t$. Then, $\Pr[Y_v = 1] = \sum_{t \in T(r)} x_{tv}$.

Note that r is left unassigned in X if and only if $Y_v = 0$ for all $v \in V$. Now, since the rounding is done independently at the vehicle level, the variables Y_v

for $v \in V$ are independent, and so r is left unassigned with probability

$$\prod_{v \in V}(1 - \Pr[Y_v = 1]) \leq e^{-\sum_{v \in V} \Pr[Y_v=1]}$$
$$= e^{-\sum_{(t,v) \in T(r) \times V} x_{tv}} = e^{-1},$$

where, as before, the last equality holds by the LP constraints. Lastly, by linearity of expectation, the expected number of requests left unassigned in the final RTV assignment is at most $|R|/e$ (i.e., less than 36.8% of all requests). □

In practice, a feasible solution to the RTV assignment problem might not exist (e.g., when vehicle demand exceeds supply). Therefore, one may need to consider the penalty version of the problem, in which a penalty is paid for each ignored request. In this setting, unassigned requests are typically carried over to the next round of batch assignments (e.g., 5 to 30 seconds later). In [58], we argue that one can still use our algorithm for this version of the problem in the sense that, after a number of rounds and for any request $r \in R$, with high probability the algorithm either assigns it to a vehicle or deliberately ignores it; in accordance with the sequence of under-supplied problem instances and not because of a rounding error.

4.4 Computational Experiments

To evaluate our algorithm, we run a number of simulations of a day in the operations of a high-capacity ridesharing system. We use publicly available NYC Taxi and Limousine Commission (TLC) data [60] and an implementation[2] of the

[2]Due to Matthew Zalesak.

Alonso-Mora et al. framework [6]. In each simulation, we solve the penalty version of the distance-minimizing RTV assignment problem using a commercial ILP solver. Next, we gather all instances of the RTV assignment problem solved throughout our simulations and solve them once again using our techniques. Note that we are *not* comparing simulation paths to evaluate system-level performance. Rather, we use simulation paths to produce sets of test instances. This way we can have a side-by-side comparison of our algorithm against the use of a commercial ILP solver (the same we use to solve our LP relaxations). Each simulation produces $1,440$ instances.

We run a total of 10 simulations, one for each combination[3] of 500 or 1000 vehicles and vehicles of capacity $k = 2, 3, 4, 5, 6$. For simulations with 500 vehicles, the mean number of requests per instance is around 370 requests[4]. Likewise, for simulations with 1000 vehicles, the mean number of requests per instance is around 560 requests.

In addition, we evaluate a deterministic rounding heuristic inspired by our randomized rounding algorithm. The only difference between the two is that, in our heuristic, each vehicle $v \in V$ is deterministically assigned to the trip $t^* \in T$ with largest fractional value (i.e., vehicle $v \in V$ is assigned to trip $t^* = \arg\max_{t \in T}\{x_{tv}\}$). Note that we may still need to execute the multiplicity correction step.

Our computational experiments reported in [58] show that our rounding algorithms achieve a performance similar to that of the ILP at a reduced computation time, far improving on our theoretical guarantee. The reason for this is

[3]For vehicle capacities $k \geq 4$, our simulations employ time-outs in the trip generation step.

[4]Even with a fixed number of vehicles, different capacities k yield different simulation paths (e.g., with different service/reneging rates), and so the instances are not identical.

that, although the assignment problem is hard in theory, the natural LP relaxation tends to be very tight in practice (confirming anecdotal information from other researchers and practitioners). We showcase this next.

Consider Figure 4.1, which shows a histogram of the fractional values supported on the LP solutions across all $1,440$ instances of a simulation with 1000 vehicles of capacity $k = 4$. Note that the overwhelming majority ($\sim 96\%$) of non-zero fractional values are in fact integral. Of course, the rounding algorithms make no mistakes on these assignments.

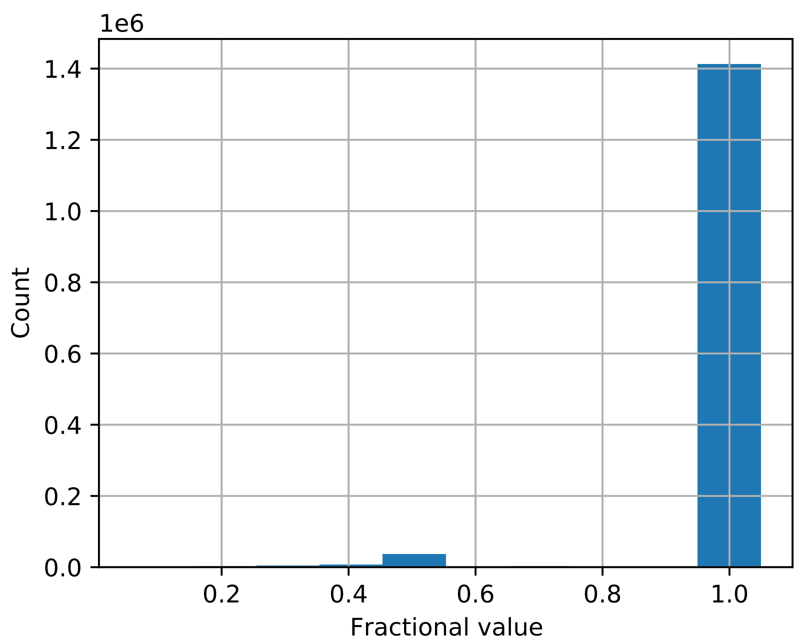


Figure 4.1: Histogram of LP support.

We extend this observation in Figure 4.2, which shows a histogram of the fractional values strictly between 0 and 1. Note that whenever an assignment is not integral, it is likely to be half-integral ($\sim 62\%$ of non-integral variables are half-integral).

We believe it would be interesting to understand the structural reasons for

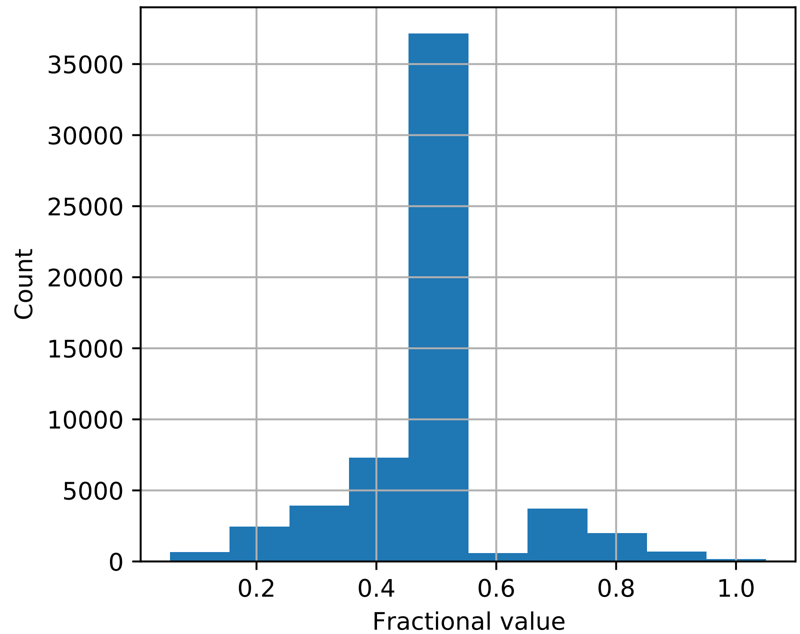


Figure 4.2: Histogram of non-integral LP support.

the near-integrality of the fractional solutions observed in our experiments. In any case, we emphasize that, in practice, the bottleneck remains to be the generation of the candidate trip list T. The difficulty of this step corresponds to our discussion about the dual separation problem.

CHAPTER 5

A GAP MEASURE FOR THE VALUE OF STATIC-DYNAMIC SYSTEM INTEGRATION

This chapter is based on work with Samitha Samaranayake and M. Grazia Speranza [57].

5.1 Introduction

The rise of on-demand mobility technologies over the past decade has sparked interest in the integration of traditional transit and on-demand systems—the number of *microtransit* (i.e., high-capacity on-demand shuttles) pilot programs conducted by transit agencies across the United States is a testament to this (see [77] for a compilation of experiences). One of the main reasons behind this is the potential for microtransit to address a fundamental trade-off in transit: the *ridership versus coverage* dilemma. It is well-known that, given a limited budget, transit networks that maximize ridership and transit networks that maximize coverage (e.g., the geographical service area) tend to be vastly different (see [75] for a practitioner-oriented discussion). Intuitively, integrated systems may bridge this gap by letting each sub-system do what it does best: transit should focus on ridership, microtransit should extend coverage as a first/last mile service, and the two should be jointly optimized.

However, unlike purely fixed systems or purely on-demand systems, integrated systems are not well-understood: their planning and operational problems are significantly more challenging, and their broader implications are the source of a heated debate. Some transportation researchers and practitioners have suggested that on-demand systems can complement traditional transit (e.g., [70, 28, 38, 5, 72, 49]). At the same time, others have raised concerns about or even flat-out dismissed the supposed benefits (e.g., [75, 65, 76, 77, 59]).

Motivated by this debate, in this chapter I introduce the *dynamicity gap* as a general concept that quantifies the attainable benefit of allowing (but not requiring) dynamic components in the response strategy for a multi-stage optimization problem. I focus on the dynamicity gap within the context of the strategic planning of transit infrastructure networks; the first step in the transit system design process, and arguably the most decisive one since all subsequent steps depend on it. However, the concept is more generally applicable in domains where goals can be met through a combination of static and dynamic (i.e, stage-specific) decisions.

First, I develop an analytical framework with which to estimate the dynamicity gap as a function of the problem input parameters. The framework allows one to certify the value of dynamism (i.e., a dynamicity gap greater than one) for certain combinations of input parameters. Then, I focus on the dynamicity gap within the context of the strategic planning of transit infrastructure networks. I pose the design of integrated transit networks as a multi-stage network design problem, and showcase the framework with two sets of computational experiments. I provide both qualitative and quantitative insights about the settings in which the integration of transit and on-demand systems may certifiably be a worthwhile investment.

As I formalize this study, I point to Table 5.1 for a summary of notation.

Symbol	Description
$G = (V, E)$	Underlying graph topology
$T \in \mathbb{N}$	Number of stages
$\delta > 0$	Stage duration
$\mathcal{I} := \{I_1, I_2, \ldots, I_k\} \subseteq 2^{V \times V}$	Possible input scenarios (a.k.a. travel demand realizations)
$I_i \in \mathcal{I}$	ith possible input scenario
$I^t \in \mathcal{I}$	Input scenario during the tth stage
$\mathcal{P}_i \subseteq \{0, 1\}^{2m}$	Set of integrated networks configurations that can serve I_i
$\mathcal{P}^t \subseteq \{0, 1\}^{2m}$	Set of integrated networks configurations that can serve I^t
$(x, z_i) \in \mathcal{P}_i$	Integrated static x and dynamic z_i network that can serve I_i
$(x, z^t) \in \mathcal{P}^t$	Integrated static x and dynamic z^t network that can serve I^t
$\mathcal{D}$	Probability distribution over $\mathcal{I}$
p_i	$\Pr_{\mathcal{D}}[I^t = I_i]$ for every $t \in [T]$
$c_s := c_s(\delta) \in \mathbb{R}^m_{\geq 0}$	Per-stage cost vector for static network (with usage duration δ)
$c_d := c_d(\delta) \in \mathbb{R}^m_{\geq 0}$	Per-stage cost vector for dynamic network (with usage duration δ)
$c \in \mathbb{R}^m_{\geq 0}$	Cost vector for a single run of the static network
$\eta > 0$	Surcharge coefficient
$\delta_s > 0$	Transit headway
$\delta_d > 0$	Microtransit batching interval
$\alpha \geq 1$	Dynamicity gap (see (5.2) for definition)
$\theta > 0$	Relative cost coefficient (in Remark 5.4.2, let $\delta := \delta_d$ and $\theta = \eta \cdot (\delta_s/\delta_d)$)
$\theta^\dagger := \alpha(1)$	Dynamicity gap when $\theta = 1$ (see Theorem 5.4.1)

Table 5.1: Summary of notation.

5.2 Transit Network Design

We study the dynamicity gap within the context of centrally designed integrated transit networks. To this end, we first describe the *Steiner forest problem*; the prototypical problem in network design. We focus on this abstraction because it lends itself to mathematical analysis and comprehensive experimentation that we believe to be useful at the level of the strategic planning of transit infrastructure networks. One can in principle enhance it with operational features such as capacity constraints, detour constraints (i.e., ensuring travel demand is met through relatively direct routes), fleet rebalancing constraints, pricing and consumer choice models, by extending the notion of picking edges to picking “lines” (i.e., picking paths in G), by limiting the number of line transfers passengers can take, and so on. However, we note that producing and efficiently solving the resulting models is a research area in and of itself (see [12, 11, 54] for some recent representative work).

In the Steiner forest problem, we are given a connected graph $G = (V, E)$ with costs $c : E \to \mathbb{R}_{\geq 0}$ and a collection $I \subseteq V \times V$ of origin-destination pairs. The problem is to find a minimum cost subset $X \subseteq E$ of edges supporting a path between each origin-destination pair. In the context of transit, G represents the underlying graph topology (e.g., a road network), I represents the travel demand, and X represents the installed network. More generally, we aim to distinguish between two types of installed networks: a transit network and a microtransit network. Therefore, we extend the Steiner forest abstraction by allowing us to pick two subsets $X, Z \subseteq E$ of edges such that $X \cup Z$ supports a path between each origin-destination pair. We treat X as a transit network and Z as a microtransit network so that $X \cup Z$ is an *integrated network*—this distinction becomes much more meaningful in subsequent sections of this chapter, where we consider a multi-stage version of the problem. Going forward, we represent subsets $X, Z \subseteq E$ of edges by their characteristic vectors $x, z \in \{0, 1\}^m$ where $m = |E|$.

5.3 Dynamicity Gap

We consider a multi-stage version of the design of integrated transit networks, wherein the transit network is static while the microtransit network is dynamic. The temporal planning horizon is partitioned into $T \in \mathbb{N}$ stages indexed by $[T] := \{1, 2, \ldots, T\}$. It is implicit that each stage has a (say uniform) duration $\delta > 0$; we elaborate on this shortly. Let $\mathcal{I} := \{I_1, I_2, \ldots, I_k\} \subseteq 2^{V \times V}$ be the collection of possible travel demand realizations (i.e., the collection of possible sets of origin-destination pairs), which we also refer to as input scenarios. For each $t \in [T]$, let $I^t \in \mathcal{I}$ be the travel demand during the tth stage and $\mathcal{P}^t$ be the finite and non-empty set of integrated network configurations that can serve it. Let $x \in \{0,1\}^m$ be decision variables corresponding to the static network (e.g., transit) paid for at cost $c_s := c_s(\delta) \in \mathbb{R}^m_{\geq 0}$ on every stage. Let $z^1, z^2, \ldots, z^T \in \{0,1\}^m$ be decision variables corresponding to the dynamic network (e.g., microtransit) over the stages, each paid for at cost $c_d := c_d(\delta) \in \mathbb{R}^m_{\geq 0}$. For $a, b \in \mathbb{R}^m$, let $a \cdot b = \sum_{j=1}^m a_j b_j$ denote their dot product. Then, the design of integrated transit networks can be posed as a multi-stage optimization problem of the form:

$$\begin{aligned} \min_{x, z^1, z^2, \ldots, z^T} \quad & \sum_{t=1}^{T} (c_s \cdot x + c_d \cdot z^t) \\ \text{s.t.} \quad & (x, z^t) \in \mathcal{P}^t, \qquad \forall t \in [T]. \end{aligned} \tag{5.1}$$

The constraints $(x, z^t) \in \mathcal{P}^t$ for each $t \in [T]$ ensure the static network x and the dynamic network z^t together serve the travel demand I^t during the tth stage.

We emphasize the general dependency of the per-stage costs $c_s := c_s(\delta)$ and $c_d := c_d(\delta)$ on the stage duration δ. If a system operates at a time-scale different from δ, it is crucial that its per-stage cost is appropriately pro-rated. To illustrate this, let $c \in \mathbb{R}^m_{\geq 0}$ encode the cost of a single dispatch of the static system (e.g., the cost per vehicle mile times the total vehicle miles covered by transit in a single dispatch of all routes) and $\eta \cdot c$ encode the cost of a single dispatch of the dynamic system, for some $\eta > 0$. The surcharge coefficient η captures the notion that static systems and dynamic systems have different operational costs on a per mile basis, independent of their relative frequencies (e.g., accounting only for backend costs such as fuel, labor, and use of software). We allow the systems to have different time-scales by distinguishing between the transit headway $\delta_s > 0$ (i.e., the time interval between subsequent transit dispatches) and the microtransit batching interval $\delta_d > 0$ (i.e., the time interval over which incoming travel demands are aggregated and microtransit routes re-optimized). Then, we pro-rate the per-stage costs of transit and microtransit as $c_s = (\delta/\delta_s) \cdot c$ and $c_d = (\delta/\delta_d) \cdot \eta \cdot c$, respectively. For example, if the stages are of duration $\delta = 1$ minute, a transit system with headway $\delta_s = 10$ minutes incurs only 1/10th of its dispatch cost on any given stage. Going forward, we tie the stage duration to the microtransit batching interval so that $\delta \coloneqq \delta_d$.

We introduce the following concept.

Definition 5.3.1 (Dynamicity Gap)**.** *Let OPT denote the cost of an optimal solution to (5.1) and* OPT^{Σ} *denote the cost of an optimal static solution to (5.1), that is one in which we additionally require* $z^1 = z^2 = \cdots = z^T = \mathbf{0}$*, where* $\mathbf{0}$ *refers to the zero vector. We define the* dynamicity gap α *of (5.1) as the unit-less coefficient*

$$\alpha \coloneqq \frac{OPT^{\Sigma}}{OPT} \geq 1. \tag{5.2}$$

Large values of α indicate large gains from introducing dynamism. Conversely, values of α close or equal to 1 indicate little to no gains from introducing dynamism. In this way, the dynamicity gap quantifies the value of dynamism.

5.4 Analytical Framework

The dynamicity gap α quantifies the value of dynamism, but computing it involves solving (5.1), which may be intractable. Moreover, we observe from (5.1) that α is influenced by implicit and explicit parameters such as the costs c_s and c_d, the stage duration δ, and the relationship between the sets of feasible configurations $\mathcal{P}^1, \mathcal{P}^2, \ldots, \mathcal{P}^T$.

The overarching goal in this chapter is to *parametrically* study the behavior of α without the need of solving the underlying multi-stage optimization problem. To this end, we assume $c_s = c$ for some $c \in \mathbb{R}^m_{\geq 0}$ and $c_d = \theta \cdot c$ for some relative cost coefficient $\theta > 0$. That is, we restrict our analysis to problems of the form:

$$\begin{array}{lll} \min\limits_{x,z^1,z^2,\ldots,z^T} & \sum\limits_{t=1}^{T} (c \cdot x + \theta \cdot c \cdot z^t) & \\ \text{s.t.} & (x, z^t) \in \mathcal{P}^t, & \forall t \in [T]. \end{array} \tag{5.3}$$

However, when tying this form back to transit, we assume $\delta \coloneqq \delta_d$ and, as described in Section 5.3, we pro-rate the per-stage costs as $c_s = (\delta_d/\delta_s) \cdot c$ and $c_d = \eta \cdot c$. Then, for any fixed $\delta_s, \delta_d > 0$, form (5.1) becomes:

$$\begin{array}{llll} & \min\limits_{x,z^1,z^2,\ldots,z^T} & \sum\limits_{t=1}^{T} \left(\frac{\delta_d}{\delta_s} \cdot c \cdot x + \eta \cdot c \cdot z^t \right) & \\ & \text{s.t.} & (x, z^t) \in \mathcal{P}^t, & \forall t \in [T] \\ = & \frac{\delta_d}{\delta_s} \cdot \min\limits_{x,z^1,z^2,\ldots,z^T} & \sum\limits_{t=1}^{T} \left(c \cdot x + \eta \cdot \frac{\delta_s}{\delta_d} \cdot c \cdot z^t \right) & \\ & \text{s.t.} & (x, z^t) \in \mathcal{P}^t, & \forall t \in [T]. \end{array} \tag{5.4}$$

Note that this is a scaled version of (5.3) with $\theta = \eta \cdot (\delta_s/\delta_d)$. In this way, we embed any dependency on η, δ_s, δ_d within θ, which then captures both the difference in service frequency and the difference in operational costs on a per mile basis.

Our analytical contributions around (5.3) are with respect to the case in which the input scenarios $I^1, I^2, \ldots, I^T$ are sampled i.i.d. from a probability distribution $\mathcal{D}$ over $\mathcal{I}$. This constitutes a specialization because, in our definition (5.2) of dynamicity gap, the sequence of input scenarios *need not* be stochastic (where we have distributional information over input parameters), or even uncertain (as in robust optimization, where we only *uncertainty set* information over input parameters). In [57] we show that if the input scenarios $I^1, I^2, \ldots, I^T$ are sampled i.i.d. from a probability distribution $\mathcal{D}$ over $\mathcal{I}$, and moreover $T \to \infty$, then we can reformulate (with almost sure (a.s.) convergence in the sets of near-optimal solutions) the horizon-normalized version of the multi-stage problem (5.3), wherein we scale the objective function by $1/T$, as the two-stage stochastic problem:

$$\begin{aligned} \min_{x, z_1, z_2, \ldots, z_k} \quad & c \cdot x + \sum_{i=1}^{k} p_i \cdot \theta \cdot c \cdot z_i \\ \text{s.t.} \quad & (x, z_i) \in \mathcal{P}_i, \qquad \forall i \in [k]. \end{aligned} \tag{5.5}$$

Recall $\mathcal{I} := \{I_1, I_2, \ldots, I_k\}$ is the collection of possible input scenarios. For each $i \in [k]$, let $I_i \in \mathcal{I}$ be the ith input scenario and let $\mathcal{P}_i$ be the finite and non-empty set of integrated network configurations that can serve it. Let $p_i = \Pr_{\mathcal{D}}[I^t = I_i]$ for every $t \in [T]$. Let $x \in \{0, 1\}^m$ be decision variables corresponding to the first-stage network and, for each $i \in [k]$, let $z_i \in \{0, 1\}^m$ be decision variables corresponding to the second-stage network under input scenario I_i. Then, the constraints $(x, z_i) \in \mathcal{P}_i$ for each $i \in [k]$ ensure the first-stage network x and the second-stage network z_i together serve the input scenario I_i. This intuitive result is closely related to the convergence of the Sample Average Approximation (SAA) method shown by Kleywegt et al. [44].

As a corollary, in this case, the dynamicity gap α of (5.3) reduces a.s. to the dynamicity gap of (5.5); the ratio between the cost of an optimal static solution to (5.5), that is one in which we additionally require $z_1 = z_2 = \cdots = z_k = \mathbf{0}$, and the cost of an optimal solution to (5.5). This equivalence allows us to treat the dynamicity gap $\alpha := \alpha(\theta)$ as a function $\alpha : \mathbb{R}_{>0} \to \mathbb{R}_{\geq 1}$ of the relative cost coefficient θ. In this way, our main analytical contribution is a certificate of the value of dynamism (i.e., a certificate that $\alpha(\theta) > 1$) whenever the relative cost coefficient does not exceed a certain value. Although this certificate is not tight in general, in [57] we argue it does not require solving the two-stage stochastic problem but a collection of independent, deterministic single-stage optimization problems, and thus it is *relatively* tractable[1].

Theorem 5.4.1. *Suppose $I^1, I^2, \dots, I^T$ are sampled i.i.d. from a probability distribution $\mathcal{D}$ over $\mathcal{I}$ and, moreover, $T \to \infty$. Let $\theta^\dagger := \alpha(1)$ be the dynamicity gap of (5.5) when $\theta = 1$—equivalently a.s., the dynamicity gap of (5.3) when $\theta = 1$. For $\theta > 0$ we a.s. have*

$$\alpha(\theta) \geq \max\left\{\frac{\theta^\dagger}{\theta}, 1\right\}. \tag{5.6}$$

Our choice of notation $\theta^\dagger := \alpha(1)$ (as opposed to $\alpha^\dagger := \alpha(1)$) follows from the way we use (5.6): it implies that if the relative cost coefficient θ satisfies $\theta < \theta^\dagger$, then $\alpha(\theta) > 1$. In [57] we strengthen this result to estimate $\alpha(\theta)$ to any arbitrary precision, provided we solve a finite number of two-stage stochastic problems.

Remark 5.4.2. *Tying this result back to transit, under the transformation from (5.1) to (5.4) wherein $\delta := \delta_d$, $c_s = (\delta_d/\delta_s) \cdot c$, and $c_d = \eta \cdot c$, the condition $\theta < \theta^\dagger$ is equivalent to $\eta \cdot (\delta_s/\delta_d) < \theta^\dagger$. To see this, note that for any fixed $\delta_s, \delta_d > 0$ we have*

$$\alpha := \frac{OPT^\Sigma}{OPT} = \frac{\begin{array}{lll} \frac{\delta_d}{\delta_s} \cdot \min\limits_{x} & \sum\limits_{t=1}^{T} c \cdot x & \\ \text{s.t.} & (x, \mathbf{0}) \in \mathcal{P}^t, & \forall t \in [T] \end{array}}{\begin{array}{lll} \frac{\delta_d}{\delta_s} \cdot \min\limits_{x, z^1, z^2, \dots, z^T} & \sum\limits_{t=1}^{T} \left(c \cdot x + \eta \cdot \frac{\delta_s}{\delta_d} \cdot c \cdot z^t\right) & \\ \text{s.t.} & (x, z^t) \in \mathcal{P}^t, & \forall t \in [T] \end{array}}$$

$$= \frac{\begin{array}{lll} \min\limits_{x} & \sum\limits_{t=1}^{T} c \cdot x & \\ \text{s.t.} & (x, \mathbf{0}) \in \mathcal{P}^t, & \forall t \in [T] \end{array}}{\begin{array}{lll} \min\limits_{x, z^1, z^2, \dots, z^T} & \sum\limits_{t=1}^{T} \left(c \cdot x + \eta \cdot \frac{\delta_s}{\delta_d} \cdot c \cdot z^t\right) & \\ \text{s.t.} & (x, z^t) \in \mathcal{P}^t, & \forall t \in [T] \end{array}}.$$

[1]For example, although the single-stage problems may remain NP-hard and the collection may be large, the fact that they are independent allows parallelized computation.

By Theorem 5.4.1, the last ratio is a.s. greater than one whenever $\theta = \eta \cdot (\delta_s/\delta_d) < \theta^\dagger$, where $\theta^\dagger := \alpha(1)$ is computed for the special case in which $\theta = \eta \cdot (\delta_s/\delta_d) = 1$—for example, if $\eta = 1$ and $\delta_s = \delta_d$.

We view this as a quick, high-level rule of thumb giving a green light for the full-blown integrated transit system design process: given a microtransit batching interval $\delta_d > 0$ and a probability distribution $\mathcal{D}$ over input scenarios $\mathcal{I}$ of duration $\delta := \delta_d$, we set $\eta \cdot (\delta_s/\delta_d) = 1$ to (relatively) tractably compute $\theta^\dagger := \alpha(1)$, with which we can certify the value of microtransit for certain combinations of transit frequency δ_s and surcharge coefficient η, namely whenever $\eta \cdot \delta_s < \theta^\dagger \cdot \delta_d$.

5.5 Computational Experiments

By (5.6), higher values of $\theta^\dagger$ lead to a larger θ regime wherein the dynamicity gap $\alpha(\theta)$ is certifiably greater than one. Therefore, as our third contribution, we use $\theta^\dagger$ as a proxy measure for the value of dynamism and study how it is influenced by other parameters implicit in (5.3) within the context of integrated transit networks. We conduct two sets of computational experiments, one very stylized and one more realistic.

5.5.1 Stylized Experiments

We consider a multi-stage version of the Steiner tree problem as the most elemental abstraction for the design of integrated transit networks. In the Steiner tree problem, we are given a graph $G = (V, E)$ and a set $I \subseteq V$ of terminals. The problem is to find a minimum cost set of edges connecting every pair in I. The possible input scenarios $\mathcal{I}$ correspond to the possible terminal sets.

We consider all 995 unweighted connected simple graphs on $2 \leq n \leq 7$ nodes. Such a list has been compiled by Read and Wilson [66] and is retrievable in `python` through the `networkx` package [37]. These are admittedly small graphs, but this is what enables us to run exhaustive experiments—the number of such graphs grows exponentially in n, and the Steiner tree problem is well-known to be NP-hard.

We consider probability distributions $\mathcal{D}$ over $\mathcal{I}$ arising from independent Bernoulli trials on the nodes. For each $u \in V$, let $q_u := \Pr[u \in I]$ be the probability that u is a terminal. This yields a probability distribution $\mathcal{D}$ over $\mathcal{I}$ with

$$p_i = \Pr_{\mathcal{D}}[I = I_i] = \prod_{u \in I_i} q_u \prod_{u \in V \setminus I_i} (1 - q_u) \tag{5.7}$$

for all $i \in [k]$. Different choice of parameters $0 \leq q_u \leq 1$ for $u \in V$ yields different distributions. We test three different rules to generate these parameters:

1. If $q_u = 1/2$ for each $u \in V$, then equation (5.7) yields $p_i = 1/2^n$ for each $i \in [k]$. This is the uniform distribution over $\mathcal{I}$, which we denote by $\mathcal{U}$.

For $u, v \in V$, let $\ell(u, v)$ be the shortest-path length between u and v (e.g., with respect to costs $c : E \to \mathbb{R}_{\geq 0}$). The *closeness centrality* of a node $u \in V$, denoted by $C(u)$, is given by $C(u) := (n-1)/\sum_{v \in V} \ell(u, v)$. This centrality measure, first introduced by [8], characterizes a node as "central" if it is close to all other nodes. We say a node is "peripheral" if it is not central. Intuitively, if $G = (V, E)$ represents a road network, nodes in a prominent downtown area are "central" while nodes in a suburban area are "peripheral."

2. If $q_u = C(u) \cdot n/(n-1)$ for each $u \in V$, then (5.7) yields a distribution biased towards terminal sets consisting of "central" nodes yet still supported on "peripheral" nodes. We denote this distribution by $\mathcal{D}^{+\text{cent}}$. Intuitively, we think of $\mathcal{D}^{+\text{cent}}$ as a distribution where demand concentrates in downtown areas as a function of how "central" they are, but still arises sparingly elsewhere.

3. Conversely, if $q_u = 1 - C(u) \cdot n/(n-1)$ for each $u \in V$, then (5.7) yields a distribution biased towards terminal sets consisting of "peripheral" nodes yet still supported on "central" nodes. We denote this distribution by $\mathcal{D}^{-\text{cent}}$. Intuitively, we think of $\mathcal{D}^{-\text{cent}}$ as a distribution where demand concentrates in suburban areas as a function of how "peripheral" they are, but still arises sparingly elsewhere.

Lastly, to study the effects of the underlying topology, we characterize graphs through three different measures of connectivity:

1. The *average degree* of a graph G, denoted by $\bar{d}(G)$, is the average degree over all nodes. Formally, $\bar{d}(G) := \sum_{u \in V} |\mathcal{N}(u)|/n = 2m/n$, where $\mathcal{N}(u) \subseteq V$ is the set of neighbors of $u \in V$.

2. The *average node connectivity* of a graph G, first introduced by [10] and denoted by $\bar{\kappa}(G)$, is the average, over all pairs of nodes, of the maximum number of internally node-disjoint paths connecting them. Formally, $\bar{\kappa}(G) := \sum_{u,v \in V : u \neq v} \kappa_G(u,v)/\binom{n}{2}$, where $\kappa_G(u,v)$ is the maximum number of internally node-disjoint paths connecting u and v in G.

3. Let L_G be the Laplacian matrix of a graph G and $\lambda_1, \lambda_2, \ldots, \lambda_n$ be its eigenvalues, counting multiplicities, in decreasing order. The *algebraic connectivity* of G, denoted by $a(G)$, is the second smallest eigenvalue of L_G counting

multiplicities. That is, $a(G) \coloneqq \lambda_{n-1}$. It holds that $a(G) > 0$ if and only if G is connected. It moreover holds that $a(G) \leq n$, with the inequality holding at equality if and only if G is the complete graph on n nodes.

In Figure 5.1, we present scatter plots of $\theta^{\dagger}$ as a function of graph connectivity measures for graphs on $n = 7$ nodes and different distributions $\mathcal{D}$ over $\mathcal{I}$.

The figure shows medium to strong correlation between graph connectivity and the value of dynamism. For the distribution $\mathcal{D}^{+\text{cent}}$ biased toward "central" nodes, dynamism tends to be more valuable on sparsely-connected graphs. This can be explained as follows: in well-connected graphs, a large proportion of nodes are highly "central," in which case a large proportion of nodes are "almost always" (in a colloquial sense of the term) terminals under $\mathcal{D}^{+\text{cent}}$. Given that road networks are far from being complete graphs, this supports the notion that microtransit may be valuable if demand concentrates in downtown areas as a function of their prominence yet still appears sparingly in suburban areas. Conversely, for the uniform distribution $\mathcal{U}$ and the distribution $\mathcal{D}^{-\text{cent}}$ biased toward "peripheral" nodes, dynamism tends to be more valuable on well-connected graphs. This can be explained as follows: in sparsely-connected graphs, a large proportion of edges are utilized under most input scenarios (especially if terminals are likely to be on the "periphery", as is the case for both $\mathcal{U}$ and $\mathcal{D}^{-\text{cent}}$), rendering dynamism unnecessary. This suggests that, in cities with high spatial segregation of urban functions (e.g., residential, commercial, industrial), wherein there is no prominent "downtown" and demand concentrates in "peripheral" areas, microtransit may be unnecessary.

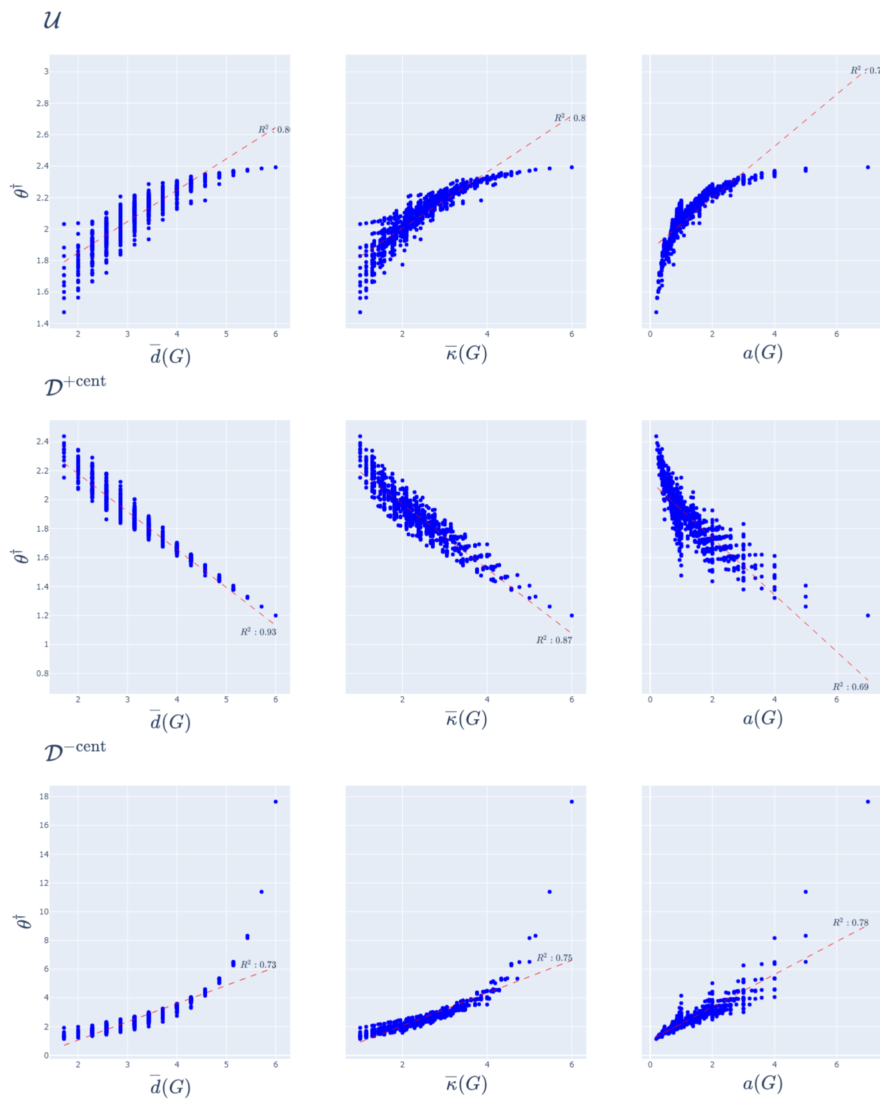


Figure 5.1: $\theta^\dagger := \alpha(1)$ as a function of graph connectivity measures for the multi-stage Steiner tree problem on graphs on $n = 7$ nodes and different distributions $\mathcal{D}$ over $\mathcal{I}$: the uniform distribution $\mathcal{U}$, the distribution $\mathcal{D}^{+\text{cent}}$ biased toward "central" nodes, and the distribution $\mathcal{D}^{-\text{cent}}$ biased toward "peripheral" nodes. The dashed lines correspond to linear fits.

5.5.2 Realistic Experiments

We now consider a more realistic abstraction for the design of integrated transit networks—the Steiner forest problem and its multi-stage version, as described in Section 5.2 and Section 5.3, respectively. We obtain a crowdsourced graph $G =$

(V, E) representing the Manhattan road network through the `osmnx` package of Boeing [14]. The nodes V represent intersections and the edges E represent road segments weighted by length $c : E \to \mathbb{R}_{\geq 0}$ in meters. We treat length as a proxy for both operating cost and travel time. We represent travel demand with taxi trip records from June 2016, available for download from [60].

To bring this abstraction closer to reality, and to reduce the number of variables needed in our integer linear programming formulations via multi-commodity flows, we impose detour constraints on pairwise connectivity. Namely, if I^t is the travel demand during the tth stage, $(u, v) \in I^t$, and the shortest-path length between u and v in G with respect to costs $c : E \to \mathbb{R}_{\geq 0}$ is $\ell(u, v)$, then the shortest-path length between u and v in the integrated network during the tth stage must be less than or equal to $\rho \cdot \ell(u, v)$ for some allowable detour factor $\rho \geq 1$.

We preprocess the raw data as follows. For tractability purposes, we focus on a subset of Manhattan roughly south of the Flatiron Building and prune G accordingly. We turn G into a simple undirected graph after deleting any self-loops, bi-directing every edge, and removing any duplicates. We delete any nodes of unit degree and contract any edges shorter than 30 meters. To account for lower speed limits and lower traffic light priority on streets (roughly traversing G from east to west) compared to avenues (roughly traversing G from south to north), we augment the length of edges labeled as "residential" or as "unclassified" by a factor of 1.5—road class labels are part of the crowdsourced data obtained via `osmnx`. We focus on trips starting on weekdays between 7:00 AM and 8:00 AM. We match the geographical start of a trip (encoded by latitude and longitude) to the nearest node in G and discard the trip if the Euclidean distance exceeds 250 meters. We do the same with the geographical end of a trip. We discard any trips shorter than 1000 meters as these are unlikely to take place in transit.

Recall from Section 5.3 that δ is the stage duration, δ_d is the microtransit batching interval, and that we match $\delta \coloneqq \delta_d$. Given any fixed δ_d, we distribute the trips into bins of uniform duration $\delta \coloneqq \delta_d$ based on their start timestamp. The trips assigned to each bin constitute the input scenario of each stage—we assume these satisfy the i.i.d. condition of the two-stage reformulation outlined in Section 5.4. Since the data is finite, the number of stages depends on the choice of δ_d. For example, since there were 22 weekdays in July 2016, we have $(60/1)\cdot 22 = 1,320$ stages for $\delta_d = 1$ minute but only $(60/15)\cdot 22 = 88$ stages for $\delta_d =$ 15 minutes. Since trips correspond to the same hourly interval on weekdays, for binning purposes we focus on δ_d a divisor of 60 minutes.

We aim to use $\theta^\dagger \coloneqq \alpha(1)$ as a proxy for the critical relative cost coefficient $\theta^* \coloneqq \arg\min_{\theta>0}\{\alpha(\theta) = 1\}$. We compute OPT(1) by solving each stage independently in a non-anticipatory manner, as justified in Section 5.4. We do so with a 5% optimality tolerance and a timeout of $\max\{10, \delta_d\}$ minutes. If there are $T \in \mathbb{N}$ stages, we let $p_i = |\{t \in [T] : I^t = I_i\}|/T$ for all $i \in [k]$. However, for our scale of G, computing OPT^Σ remains challenging as the Steiner forest problem is NP-hard. Therefore, as a polynomial time solvable approximation, we use the length of a minimum spanning tree of G. In [57], we justify this as follows.

Proposition 5.5.1. *Let* $\hat{\theta}^\dagger \coloneqq \ell(MST(G))/OPT(1)$, *where* $\ell(MST(G))$ *denotes the length of a minimum length spanning tree of* G. *If the input scenarios* $I^1, I^2, \ldots, I^T$ *are sampled i.i.d. from a probability distribution* $\mathcal{D}$ *over* $\mathcal{I}$ *with* $p_i > 0$ *for all* $i \in [k]$, *then a.s. as* $T \to \infty$ *we have* $\theta^\dagger \geq \hat{\theta}^\dagger$.

Recall the condition $\eta \cdot (\delta_s/\delta_d) < \theta^\dagger$ certifying the value of dynamism in Remark 5.4.2. For any fixed stage duration $\delta := \delta_d$ and any fixed allowable detour factor ρ, our experiments use Proposition 5.5.1 to compute the lower bound $\hat{\theta}^\dagger$ on $\theta^\dagger$—this allows us to certify the value of dynamism whenever $\eta \cdot (\delta_s/\delta_d) < \hat{\theta}^\dagger$. If $\delta_s = \delta_d$, the condition reduces to $\eta < \hat{\theta}^\dagger$. More generally, for $\delta_s \neq \delta_d$, the condition reduces to $\eta < \hat{\theta}^\dagger \cdot (\delta_d/\delta_s)$.

Figure 5.2 shows the term $\hat{\theta}^\dagger \cdot (\delta_d/\delta_s)$ as a function of δ_d, δ_s and ρ for $\delta_d \leq \delta_s$—which is to say that the microtransit batching interval less than or equal to the transit headway. In this way, we obtain quantitative estimates on the parameter combinations under which dynamism is certifiably valuable. For example, the curves suggest that if the microtransit batching interval and the transit headway are each 15 minutes, then on-demand integration is worthwhile as long as the surcharge coefficient η is less than around 1.5. We caution that these experiments, although more realistic than those in Section 5.5.1, are still based on an abstraction that does not capture factors such as demand-side effects, fleet size, vehicle capacities, and rebalancing. Nevertheless, the figure provides robust qualitative insights. First, for any fixed transit headway δ_s, the value of dynamism increases with the microtransit batching interval δ_d assuming passengers tolerate longer waiting times relative to the transit headway. This effect is amplified for small δ_s, where the surcharge coefficient η can be large—for very small δ_s, transit operations are already very costly, in which case microtransit can be valuable even if the surcharge coefficient is very large (again, assuming passengers tolerate long waiting times relative to the transit headway). We moreover observe slight gains from increased passenger tolerance to detours as captured by ρ, particularly for small δ_d. Namely, if we treat $\hat{\theta}^\dagger := \hat{\theta}^\dagger(\delta_d, \rho)$ as a function $\hat{\theta}^\dagger : \mathbb{R}_{>0} \times \mathbb{R}_{>0} \to \mathbb{R}_{\geq 1}$, the plots show that for fixed δ_d and $\rho_1 \geq \rho_2$ we have

$\hat{\theta}^{\dagger}(\delta_d, \rho_1) \geq \hat{\theta}^{\dagger}(\delta_d, \rho_2)$. These effects can be explained as follows: for small δ_d, there are fewer requests per stage and so travel demands are met with more direct, less shared paths. In this case, increasing ρ enhances sharing thereby reducing costs. For large δ_d, there are more requests per stage and so travel demands are more likely to overlap, naturally enhancing sharing without the need of increasing ρ. In other words, the longer customers wait to be served by the dynamic system, the cheaper it is for the system to offer them shared yet direct travel.

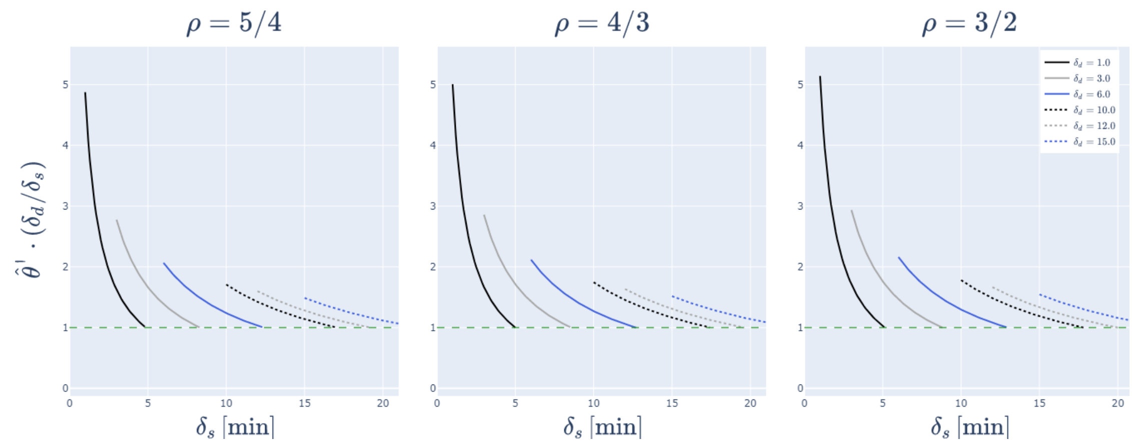


Figure 5.2: $\hat{\theta}^{\dagger} \cdot (\delta_d/\delta_s)$ as a function of δ_d, δ_s, and ρ. The left, center, and right panels correspond to ρ equal to 5/4, 4/3, and 3/2, respectively. Within each panel, each curve corresponds to a different choice of $\delta := \delta_d$, as indicated by the legend. For fixed δ_d, δ_s, and ρ, we certify the value of dynamism for surcharge coefficient η less than the value along the corresponding curve. For example, if $\delta_d = 6$ and $\rho = 5/4$ (i.e., solid blue curve on the left panel), and moreover $\delta_s = 10$, we can certify the value of dynamism for $\eta \leq 1.25$ (i.e., the vertical axis value of the curve at the horizontal axis value of 10).

The effects of δ_d are further evidenced in Figure 5.3. Recall we compute an optimal non-anticipatory solution with characteristic vector χ^t for input scenario $I^t \in \mathcal{I}$ (i.e., a minimum cost χ^t such that $(\chi^t, \mathbf{0}) \in \mathcal{P}^t$) individually for each stage $t \in [T]$. For each $j \in [m]$ corresponding to the jth edge, we compute the frequency $|\{t \in [T] : \chi^t_j = 1\}|/T$ with which it appears as part of the non-anticipatory solutions. In other words, we rank road segments by the frequency (over the stages) with which they appear as part of the installed network, and thereby their importance as potential trunk lines. We observe that the smaller $\delta := \delta_d$ is, the fewer high rank road segments there are. However, these few road segments are precisely the best candidates for forming the static network: they appear as part of the installed network in most stages despite the fact that there are few requests per stage for small δ_d—they are the ones that enable sharing. We believe this can be leveraged in a subsequent step of the transit system design process: the design of the operational network. In particular, we believe frequently used road segments, especially those that are frequently used for small $\delta := \delta_d$, can be combined to obtain a good, data-driven initial set of candidate lines for line planning via column generation (see [15, 31]).

$\delta_d = 1$ [min] $\delta_d = 6$ [min] $\delta_d = 15$ [min]

Figure 5.3: We compute an optimal non-anticipatory solution independently for each stage $t \in [T]$, with $\rho = 5/4$ and different choices of $\delta := \delta_d$ in minutes. The darker a road segment, the higher its frequency (over the stages) as part of the installed network.

CHAPTER 6

A SCHEDULING PROBLEM IN INFRASTRUCTURE PLANNING

This chapter is based on work with Samitha Samaranayake [56].

6.1 Introduction

A water company decides to expand its network with a set of water lines, but it cannot build them all at once. However, it starts reaping benefits from a partial network expansion. In what order should the company build the water lines?

In this chapter, I formalize a class of permutatorial scheduling problem with combinatorial (or continuous) subproblems, capturing this and other applications of incremental deployment. I show that, for additive (or linear) objective functions, efficient polyhedral methods for the subproblems extend to the scheduling problem. The main technical ingredient is the permutahedron.

6.2 Problem Formulation

Let $E = \{1, 2, \dots, m\}$ be a ground set of elements and let $\mathcal{F} \subseteq 2^E$ be a collection of subsets of the ground set. For brevity we denote $[m] := \{1, 2, \dots, m\}$. Let $f : 2^E \to \mathbb{R}$ be a set function. Our starting point is combinatorial problems of the form

$$\max\{f(S) : S \in \mathcal{F}\}, \tag{6.1}$$

where $\mathcal{F}$ is the feasible region and f is the objective function. Let Π be the set of permutations of E—the bijections from E onto itself. Note that each permutation $\pi \in \Pi$ induces a distinct maximal chain $\emptyset = H^0(\pi) \subset H^1(\pi) \subset H^2(\pi) \subset \cdots \subset H^m(\pi) = E$ of subsets of E ordered by inclusion, where $H^j(\pi) := \{i \in E : \pi(i) \leq j\}$. For a subset $H \subseteq E$ of elements, let $\mathcal{F}(H) := \{S \subseteq H : S \in \mathcal{F}\}$ be the restriction of $\mathcal{F}$ to H. We consider permutatorial problems of the form

$$\max\{g(\pi) : \pi \in \Pi\}, \tag{6.2}$$

where $g : \Pi \to \mathbb{R}$ is the permutation function given by

$$g(\pi) := \sum_{j=1}^{m} \max\{f(S) : S \in \mathcal{F}(H^j(\pi))\}.$$

This form captures problems where a set of elements to be realized has been fixed, but the order in which they are realized is to be determined. The objective function sums over the objective values of a sequence of *decoupled* combinatorial subproblems of the form (6.1), each of which may only use elements realized by their corresponding step. At any given step, whether a combination of realized elements can be used is still subject to $\mathcal{F}$. Our main result is that, for f additive, efficient polyhedral methods for the combinatorial subproblem (6.1) extend to the permutatorial problem (6.2).

Theorem 6.2.1. *If f is additive and (6.1) can be optimized in polynomial time via the polyhedral approach, then (6.2) can be optimized in polynomial time via the polyhedral approach.*

Our technique more generally applies if we replace the combinatorial subproblem (6.1) by a continuous subproblem of the form

$$\max\{F(x) : x \in P\}, \tag{6.3}$$

where $P \subseteq \mathbb{R}^n$ is a polyhedral feasible region and $F : \mathbb{R}^n \to \mathbb{R}$ is the objective function. In this case, the corresponding permutatorial problem is

$$\max\{g(\pi) : \pi \in \Pi\}, \tag{6.4}$$

where $g : \Pi \to \mathbb{R}$ is the permutation function given by

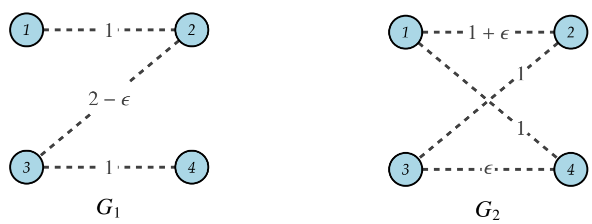
$$g(\pi) := \sum_{j=1}^{m} \max\{F(x) : x \in P(\chi_{H^j(\pi)})\},$$

where $\chi_{H^j(\pi)} \in \{0,1\}^m$ is the characteristic vector corresponding to $H^j(\pi)$ and $P(\chi_{H^j(\pi)})$ is a linear "restriction" of P to $\chi_{H^j(\pi)}$. The precise form of this "restriction" is problem-specific, but in any case it ensures that $x \in P(\chi_{H^j(\pi)})$ may only use elements realized by the jth step and is still subject to P. In particular, if $P \subseteq [0,1]^n$ and $n = m$, then $P(\chi_{H^j(\pi)}) := \{x \in \mathbb{R}^m : x \leq \chi_{H^j(\pi)} \wedge x \in P\}$ is a natural choice. We show that, for F a linear function, efficient polyhedral methods for the continuous subproblem (6.3) extend to the permutatorial problem (6.4). We in turn use this to prove our main result.

6.2.1 Greedy Can Fail

To motivate our focus on polyhedral methods, consider the performance of two natural greedy algorithms for (6.2). The most natural greedy algorithm maintains a set H of realized elements and, on every step, brings $e^* := \arg\max_{e \in E \setminus H}\{\max\{f(S) : S \in \mathcal{F}(H+e)\} - \max\{f(S) : S \in \mathcal{F}(H)\}\}$ into H. Another natural greedy algorithm first finds $S^* := \arg\max\{f(S) : S \in \mathcal{F}\}$ and greedily brings its elements into H. It then brings any elements remaining in $E \setminus S^*$ into H. As we showcase in the following examples, neither algorithm is optimal, even when f is an additive set function.

Example 6.2.2 (Matchings)**.** *Consider the following weighted graphs with $\epsilon > 0$ small.*

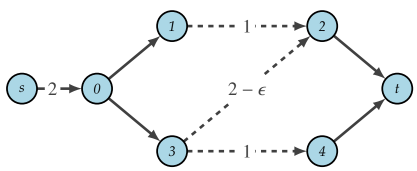


Suppose the dashed edges are realized one at time. The problem is to find an ordering in which to realize the dashed edges so that the sum of the weights of the maximum weight matchings over the steps, each of which may only use the edges realized by their corresponding step, is maximized. Note that this problem is captured by (6.2): $\mathcal{F}$ represents the set of matchings in G, $f(S)$ represents the sum of the weights of the edges in S, and $\mathcal{F}(H^j(\pi))$ represents the set of matchings in G when only the edges in $H^j(\pi)$ can be used.

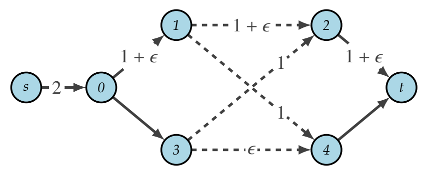
On G_1, the first algorithm is optimal, realizing the edges $\{2,3\}$, $\{1,2\}$, and $\{3,4\}$ in that order for a cumulative weight of $6-2\epsilon$. Conversely, the second algorithm is suboptimal, realizing the edges $\{1,2\}$, $\{3,4\}$, and $\{2,3\}$ in that order for a cumulative weight of 5. Therefore, the second greedy algorithm cannot achieve an approximation factor better than $5/6$ for this problem. On G_2, the first algorithm is suboptimal, realizing the edges $\{1,2\}$, $\{3,4\}$, $\{1,4\}$ and $\{2,3\}$ in that order for a cumulative weight of $5+5\epsilon$. Conversely, the second algorithm is optimal, realizing the edges $\{1,4\}$, $\{2,3\}$, $\{1,2\}$, and $\{3,4\}$ in that order for a cumulative weight of 7. Therefore, the first greedy algorithm cannot achieve an approximation factor better than $5/7$ for this problem. G_1 and G_2 can be combined to produce an instance in which neither algorithm is optimal (e.g., $G_1 \cup G_2$).

We note that if $\mathcal{F} = 2^E$ (i.e., unconstrained optimization) and f is monotone submodular, then $E = \arg\max\{f(S) : S \in \mathcal{F}\}$ and so the two greedy algorithms are equivalent. In this special case, the greedy algorithm can be shown to be a $1/e$-approximation algorithm to (6.2) using standard techniques. Next, we formalize our motivational example and consider the performance of the analogous greedy algorithms for (6.4).

Example 6.2.3 (Flows)**.** *Let $D = (V, A)$ be a directed graph with capacities $c : A \to \mathbb{Q}_{\geq 0}$. The arcs represent, for example, capacitated water lines planned for construction. Let $s, t \in V$ be distinct nodes. Suppose a water company wants to maximize the amount of water transmitted from s to t over time, including during the construction phase. Consider the problem on the following capacitated directed graphs with $\epsilon > 0$ small, which we adapt from Example 6.2.2. Unlabeled arcs are uncapacitated.*



D_1

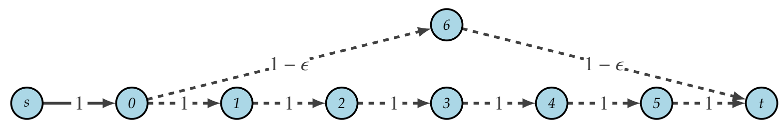


D_2

Suppose the dashed arcs are realized one at time. The problem is to find an ordering in which to realize the dashed arcs so that the sum of the maximum flows from s to t over the steps, each of which may only use the arcs realized by their corresponding step, is maximized. Note that this problem is captured by (6.4): P represents the $s - t$ flow polytope of D, $F(x)$ represents the size of the flow x, and $P(\chi_{H^j(\pi)})$ represents the $s - t$ flow polytope of D when only the edges in $H^j(\pi)$ can be used.

On D_1, the first algorithm is optimal and achieves a cumulative flow of $6-\epsilon$, whereas the second algorithm is suboptimal and achieves a cumulative flow of 5. On D_2, the first algorithm is suboptimal and achieves a cumulative flow of $5 + 5\epsilon$, whereas the second algorithm is optimal and achieves a cumulative flow of 7. D_1 and D_2 can be combined to produce an instance in which neither algorithm is optimal.

In fact, there are instances of this problem in which the approximation factor of either greedy algorithm is $\Omega(|V|)$, as we showcase with the capacitated directed graph D_3 below.



D_3

The first algorithm can be unlucky and realize arcs allowing zero flow from s to t for the first 5 steps and unit flow for only the last 3 steps, whereas it is possible to have zero flow for only the first step, $1-\epsilon$ flow for the next 6 steps, and unit flow for the last step. The second algorithm behaves identically if in the first step it picks the maximum flow supported solely along the long path in D_3. The long path in D_3 can be extended with any number of nodes to obtain the $\Omega(|V|)$ bound.

6.3 From Permutations to Chains

For $a, b \in \mathbb{R}$ with $a \leq b$ denote $[a, b] := \{x \in \mathbb{R} : a \leq x \leq b\}$. Let $P(\Pi) \subseteq [1, m]^m$ be the convex hull of permutations of E (a.k.a., the permutahedron). The following is well-known (see also Billera and Sarangarajan [13], for example).

Proposition 6.3.1 (Rado [63]). *$P(\Pi)$ has a linear inequality description given by*

$$P(\Pi) = \left\{ y \in [1, m]^m : \begin{array}{ll} \sum_{i \in E} y_i & = \binom{m+1}{2} \\ \sum_{i \in S} y_i & \leq \binom{m+1}{2} - \binom{m+1-|S|}{2}, \quad \forall S \subseteq E : S \neq \emptyset \end{array} \right\}.$$

Since $\binom{m+1}{2} - \binom{m+1-|S|}{2} - \sum_{i \in S} y_i$ is a submodular set function, this description has a strongly polynomial time separation oracle [68] (a set function $f : 2^E \to \mathbb{R}$ is said to be submodular if $f(S) + f(T) \geq f(S \cup T) + f(S \cap T)$ for every $S, T \subseteq E$). Alternatively, Goemans [32] gives a an extended formulation with $\Theta(m \log m)$ extension complexity using the sorting network of Ajtai, Komlós, and Szemerédi [4].

We give a set of $O(m^2)$ linear inequalities that, given an extreme point $y \in P(\Pi)$, produce a sequence $h^1, h^2, \dots, h^m \in [0,1]^m$ of characteristic vectors corresponding to the chain $\emptyset = H^0(y) \subset H^1(y) \subset H^2(y) \subset \cdots \subset H^m(y) = E$ of subsets of E. Let h_i^j denote the ith entry of the jth vector.

Proposition 6.3.2. *Let $y \in P(\Pi)$ be an extreme point. Then, $h^1, h^2, \dots, h^m \in [0,1]^E$ is a sequence of characteristic vectors corresponding to the chain $\emptyset = H^0(y) \subset H^1(y) \subset H^2(y) \subset \cdots \subset H^m(y) = E$ if and only if the following inequalities are satisfied:*

$$h_i^0 = 0, \qquad \forall i \in [m] \tag{6.5a}$$

$$h_i^j \le h_i^{j+1}, \qquad \forall j \in [m-1], i \in [m] \tag{6.5b}$$

$$\sum_{i\in[m]} h_i^j = j, \qquad \forall j \in [m] \tag{6.5c}$$

$$\sum_{k\in[j]} h_i^k \ge j - y_i + 1, \qquad \forall j \in [m], \forall i \in [m] \tag{6.5d}$$

Proof. Assume without loss of generality (up to rearranging) that $y_i = i$ for all $i \in [m]$.

First, suppose the inequalities are satisfied. Consider the case in which $j = 1$. (6.5d) with $i = 1$ requires $h_1^1 \ge j - y_1 + 1 = 1 - 1 + 1 = 1$. (6.5a) and (6.5b) further require $h_i^1 \ge 0$ for $i > 1$. (6.5c) requires $\sum_{i\in[m]} h_i^1 = 1$, and so $h_i^1 = 1$ for $i = 1$ and $h_i^1 = 0$ for $i > 1$ is the only candidate solution remaining. Note that h^1 is the characteristic vector of $H^1(y)$.

By way of induction, suppose h^j is the characteristic vector of $H^j(y)$ for some $1 \le j < m$. (6.5b) and induction require $h_i^{j+1} \ge 1$ for $1 \le i \le j$ and $h_i^{j+1} \ge 0$ for $i > j$. (6.5d) with $i = j + 1$ requires

$$h_{j+1}^{j+1} = h_{j+1}^1 + h_{j+1}^2 + \cdots + h_{j+1}^j + h_{j+1}^{j+1} \ge j + 1 - y_{j+1} + 1 \ge j + 1 - (j+1) + 1 = 1,$$

where the first equality holds by induction. (6.5c) requires $\sum_{i\in[m]} h_i^{j+1} = j+1$, and so $h_i^{j+1} = 1$ for $1 \leq i \leq j+1$ and $h_i^{j+1} = 0$ for $i > j+1$ is the only candidate solution remaining. Note that h^{j+1} is the characteristic vector of $H^{j+1}(y)$.

Lastly, we need to show that letting $h^1, h^2, \ldots, h^m$ be characteristic vectors corresponding to the chain $\emptyset = H^0(y) \subset H^1(y) \subset H^2(y) \subset \cdots \subset H^m(y) = E$ indeed satisfies the inequalities. (6.5a)-(6.5c) are clearly satisfied. It remains to show (6.5d) is satisfied. Take any $j \in [m]$ and $i \in [m]$. If $i \geq j+1$, then $y_i = i \geq j+1$ and so $j - y_i + 1 \leq j - (j+1) + 1 = 0$ and the inequality is trivially satisfied. If $i \leq j$, then $h_i^k = 1$ for all $i \leq k \leq j$ and so $\sum_{k\in[j]} h_i^k \geq j - i + 1 = j - y_i + 1$. □

For notational convenience, we treat the linear system (6.5a)-(6.5d) as a transformation Z that maps y to $h^1, h^2, \ldots, h^m$. We first consider the case in which our subproblem is continuous of the form (6.3). We use $P(\Pi)$ and Z to obtain

$$\max\{G(y) : y \in P(\Pi)\}, \tag{6.6}$$

as a continuous relaxation of (6.4), where $G : [1, m]^m \to \mathbb{R}$ is the continuous extension of g given by

$$G(y) := \sum_{j=1}^{m} \max\{F(x) : Z(y) = h^1, h^2, \ldots, h^m \wedge x \in P(h^j)\}.$$

In this way, the feasible region of (6.6) and the feasible region of each subproblem in $G(y)$ admit (possibly exponential sized) linear inequality descriptions.

We now consider the case in which our subproblem is combinatorial of the form (6.1). Let $P(\mathcal{F})$ be the convex hull of characteristic vectors corresponding to $\mathcal{F}$. Let $F : [0, 1]^m \to \mathbb{R}$ be a continuous extension of f. For example, if f is additive with coefficients $w \in \mathbb{R}^m$ (meaning $f(S) = \sum_{i\in S} w_i$ for all $S \subseteq E$), we may simply let $F(x) = \sum_{i=1}^{m} w_i x_i$. Then we obtain

$$\max\{F(x) : x \in P(\mathcal{F})\} \tag{6.7}$$

as a continuous relaxation of (6.1). Note that $P(\mathcal{F})$ always admits a (possibly exponential sized) linear inequality description, and that when f is additive this relaxation is exact. Similarly, we use F, $P(\Pi)$, and Z to obtain

$$\max\{G(y) : y \in P(\Pi)\} \tag{6.8}$$

as a continuous relaxation of (6.2), where $G : [1, m]^m \to \mathbb{R}$ is the continuous extension of g given by

$$G(y) := \sum_{j=1}^{m} \max\{F(x) : Z(y) = h^1, h^2, \ldots, h^m \wedge x \leq h^j \wedge x \in P(\mathcal{F})\}.$$

As before, the feasible region of (6.8) and the feasible region of each subproblem in $G(y)$ admit (possibly exponential sized) linear inequality descriptions.

6.4 Solution Approach

In this section we show that for f additive/F linear, the continuous relaxations of (6.2) and (6.4) given in Section 6.3 are in fact exact. Moreover, if the combinatorial/continuous subproblem can be solved in polynomial time via linear programming, then so can the corresponding permutatorial problem.

Theorem 6.4.1. *If F is linear and linear functions can be optimized over P in polynomial time, then (6.4) and (6.6) are equivalent and can be solved in polynomial time via linear programming.*

Proof. Since the terms of the summation in the objective function of (6.6) are decoupled given $y \in P(\Pi)$, we may rewrite (6.6) as

$$\max \left\{ \sum_{j=1}^{m} F(x^j) : y \in P(\Pi) \wedge Z(y) = h^1, h^2, \ldots, h^m \wedge x^j \in P(h^j) \right\}, \tag{6.9}$$

which can be solved in polynomial time since each set of inequalities describing the feasible region has either a compact description or a polynomial time separation oracle (in the case of $x^j \in P(h^j)$, we use the equivalence of separation and optimization and the assumption that linear functions can be optimized over P in polynomial time). Since the objective function and constraints of (6.9) are linear, there exists an extreme point optimal solution. Since in such a solution y is an extreme point of $P(\Pi)$ (this can be seen by a contradiction to optimality, assuming y is a convex combination of vertices of $P(\Pi)$), Proposition 6.3.2 implies (6.9) and

$$\max \left\{ \sum_{j=1}^{m} F(x^j) : \pi \in \Pi \wedge x^j \in P(\chi_{H^j(\pi)}) \right\} \tag{6.10}$$

are equivalent. Lastly, note that (6.10) and (6.4) are equivalent since the terms of the summation in the objective function of (6.4) are decoupled given $\pi \in \Pi$. □

We now prove our main result Theorem 6.2.1: if f is additive and linear functions can be optimized over $P(\mathcal{F})$ in polynomial time, then (6.2) and (6.8) are equivalent and can be solved in polynomial time via linear programming.

Proof of Theorem 6.2.1. Since f is additive (say, with coefficients $w \in \mathbb{R}^m$), we may simply let $F(x) = \sum_{i=1}^{m} w_i x_i$, which is linear. Then, (6.8) is the special case of (6.6) in which F is linear and $P(h^j) = \{x \in \mathbb{R}^m : x \leq h^j \wedge x \in P(\mathcal{F})\}$. Moreover, by the assumption that linear functions can be optimized over $P(\mathcal{F})$ in polynomial time, all conditions of Theorem 6.4.1 are met. Therefore, (6.8) is equivalent to the special case of (6.4) in which F is linear and $P(\chi_{H^j(\pi)}) = \{x \in \mathbb{R}^m : x \leq \chi_{H^j(\pi)} \wedge x \in P(\mathcal{F})\}$, and moreover can be solved in polynomial time via linear programming. Lastly, note that since f is additive, $\max\{F(x) : x \leq \chi_{H^j(\pi)} \wedge x \in P(\mathcal{F})\}$ is equivalent to $\max\{f(S) : S \in \mathcal{F}(H^j(\pi))\}$, and so (6.4) and (6.2) are equivalent. □

Remark 6.4.2. *More broadly, our proofs show that, for f additive/F linear, the time complexity of solving the combinatorial/continuous problem via the polyhedral approach (be it polynomial or not) extends to solving the corresponding permutatorial problem via the polyhedral approach at the expense of $O(m^2)$ additional variables and linear inequalities. For example, our technique can be incorporated within an integer linear programming framework.*

Our result extends, for example, to minimizing linear functions with non-negative coefficients over the dominant $\mathrm{dom}(P(\mathcal{F}))$ of $P(\mathcal{F})$, provided it has a compact description or a polynomial time separation oracle. The dominant $\mathrm{dom}(P) := \{x + y : x \in P, y \in \mathbb{R}^m_{\geq 0}\}$ of a polytope $P \subset \mathbb{R}^m$ is often used in optimization since it may have a simpler description than P and since minimizing non-negative linear functions over $\mathrm{dom}(P)$ is equivalent to minimizing non-negative linear functions over P.

We showcase our framework by revisiting the examples from Section 1.

Example 6.4.3 (Matchings, cont.)**.** *Let $G = (V, E)$ be a bipartite graph with weights $w : E \to \mathbb{Q}_{\geq 0}$. Let $m = |E|$ so that we treat the edge set as $E = [m]$. Then, by*

Theorem 6.2.1, solving

$$\text{maximize} \quad \sum_{j=1}^{m} \sum_{i \in E} w_i x_i^j \tag{6.11a}$$

$$y \in P(\Pi) \tag{6.11b}$$

$$h^1, h^2, \ldots, h^m = Z(y), \tag{6.11c}$$

$$x^j \le h^j, \qquad \forall j \in [m] \tag{6.11d}$$

$$\sum_{i \in \delta(v)} x_i^j \le 1, \qquad \forall j \in [m], \forall v \in V \tag{6.11e}$$

$$x^j \ge 0, \qquad \forall j \in [m] \tag{6.11f}$$

is equivalent to solving (6.2). If G were not bipartite, we could replace (6.11e) with Edmonds' [23] description of the matching polytope.

Example 6.4.4 (Flows, cont.)**.** *Let $m = |A|$ so that we treat the arc set as $A = [m]$. Then, by Theorem 6.4.1, solving*

$$\text{maximize} \quad \sum_{j=1}^{m} \sum_{i \in \delta^+(s)} f_u^j \tag{6.12a}$$

$$y \in P(\Pi) \tag{6.12b}$$

$$h^1, h^2, \ldots, h^m = Z(y), \tag{6.12c}$$

$$x^j = h^j, \qquad \forall j \in [m] \tag{6.12d}$$

$$\sum_{i \in \delta^+(u)} f_i^j = \sum_{i \in \delta^-(u)} f_i^j, \qquad \forall j \in [m], \forall u \in V \setminus \{s, t\} \tag{6.12e}$$

$$f_i^j \le c_i x_i^j, \qquad \forall j \in [m], \forall i \in A \tag{6.12f}$$

$$x^j, f^j \ge 0, \qquad \forall j \in [m] \tag{6.12g}$$

is equivalent to solving (6.4). For $u \in V$, $\delta^-(u)$ and $\delta^+(u)$ denote its incoming and outgoing arcs.

CHAPTER 7

COMBINATORICS OF PARKING OBJECTS

This chapter is based on work with Yasmin Aguillon, Dylan Alvarenga, Pamela E. Harris, Surya Kotapati, Casandra D. Monroe, Zia Saylor, Camelle Tieu, and Dwight Anderson Williams II [3]; with Pamela E. Harris, Brian M. Kamau, and Roger Tian [39]; with Douglas M. Chen, Pamela E. Harris, Eric J. Pabón-Cancel, and Gabriel Sargent [19]; and with Pamela E. Harris and Jan Kretschmann [40].

7.1 Introduction

Consider a one-way street with $n \in \mathbb{N}$ parking spots. A total of n cars enter the street sequentially, each with a preferred spot. When car $i \in [n] := \{1, 2, \ldots, n\}$ enters the street, it drives to its preferred spot $x_i \in [n]$. If spot x_i is unoccupied, it parks there. Otherwise, it continues driving down the street until it finds an unoccupied spot in which to park, if there is one. Let the list $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in [n]^n$ encode the cars' parking preferences. If the preference list $\mathbf{x}$ allows all cars to park, we say it is a parking function (of length n).

Parking functions were introduced by Konheim and Weiss [46] in their study of hashing functions. By now, they are a classical combinatorial objects with plenty of well-established results (see Yan [79] for a survey). For example, let $\mathsf{PF}_n, \mathsf{PF}_n^{\uparrow} \subseteq [n]^n$ be the set of parking functions and non-decreasing parking functions of length n, respectively. The following results are well-known.

Theorem 7.1.1 (Lemma 1, [46])**.** *Let $n \in \mathbf{N}$. Then, $|\mathsf{PF}_n| = (n+1)^{n-1}$.*

Theorem 7.1.2. *Let* $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in [n]^n$ *and* $\mathbf{x}' = (x'_1, x'_2, \ldots, x'_n)$ *be its non-decreasing rearrangement. Then,* $\mathbf{x} \in PF_n$ *if and only if* $x'_i \leq i$ *for all* $i \in [n]$*.*

Theorem 7.1.3. *Let* $n \in \mathbf{N}$*. Then,* $|\mathsf{PF}_n^{\uparrow}| = C_n$*, where* $C_n = \frac{1}{n+1}\binom{2n}{n}$ *is the nth Catalan number.*

Despite the maturity of their study, parking functions keep giving. In particular, variants on their classical version have been repeatedly shown to have connections to other areas in combinatorics (see Carlson et al. [17] for a series of examples). In this chapter, I outline the results of collaborative work along these lines; showing the ties of parking objects (variants, specializations, or generalizations of classical parking functions) to other areas in combinatorics.

7.2 Summary of Results

In this section, I outline the results in Aguillon et al. [3], Harris et al. [39], Chen et al. [19], and Harris, Kretschmann, and Martínez Mori [40].

7.2.1 A Connection to the Tower of Hanoi

The displacement of a parking function measures the total difference between where cars want to park and where they ultimately park. In [3], we prove that the set of parking functions of length n with displacement one is in bijection with the set of ideal states in the famous Tower of Hanoi game with $n+1$ disks and $n+1$ pegs, both sets being enumerated by the Lah numbers $\frac{n!(n-1)}{2}$.

7.2.2 A New Parking Procedure

In [39], we introduce a new parking procedure called *MVP parking* in which n cars sequentially enter a one-way street with a preferred parking spot from the n parking spots on the street. If their preferred spot is empty, they park there. Otherwise, they park there and the car parked in that spot is bumped to the next unoccupied spot on the street. If all cars can park under this parking procedure, we say the list of preferences of the n cars is an *MVP parking function* of length n. We show that the set of (classical) parking functions is exactly the set of MVP parking functions, although the parking outcome (order in which the cars park) is different under each parking process. This motivates the following question: Given a permutation describing the outcome of the MVP parking process, what is the number of MVP parking functions resulting in that given outcome? Our main result establishes a bound for this count which is tight precisely when the permutation describing the parking outcome avoids the patterns 321 and 3412. We then consider special cases of permutations and give closed formulas for the number of MVP parking functions with those outcomes. In particular, we show that the number of MVP parking functions which park in reverse order (that is the permutation describing the outcome is the longest word in the symmetric group $\mathfrak{S}_n$, which does not avoid the pattern 321) is given by the nth Motzkin number. We also give families of permutations describing the parking outcome for which the cardinality of the set of cars parking in that order is exponential and others in which it is linear.

7.2.3 Cars with Assorted Lengths

In [19] we introduce *parking assortments*, a generalization of parking functions with cars of assorted lengths. In this setting, there are $n \in \mathbb{N}$ cars of lengths $\mathbf{y} = (y_1, y_2, \ldots, y_n) \in \mathbb{N}^n$ entering a one-way street with $m = \sum_{i=1}^{n} y_i$ parking spots. The cars have parking preferences $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in [m]^n$, where $[m] := \{1, 2, \ldots, m\}$, and enter the street in order. Each car $i \in [n]$, with length y_i and preference x_i, follows a natural extension of the classical parking rule: it begins looking for parking at its preferred spot x_i and parks in the first y_i contiguously available spots thereafter, if there are any. If all cars are able to park under the preference list $\mathbf{x}$, we say $\mathbf{x}$ is a parking assortment for $\mathbf{y}$. Parking assortments also generalize *parking sequences*, introduced by Ehrenborg and Happ [26] and further studed by Adeniran and Yan [1], since each car seeks for the first contiguously available spots it fits in past its preference. Given a parking assortment $\mathbf{x}$ for $\mathbf{y}$, we say it is *permutation invariant* if all rearrangements of $\mathbf{x}$ are also parking assortments for $\mathbf{y}$. While all parking functions are permutation invariant, this is not the case for parking assortments in general, motivating the need for a characterization of this property. Although obtaining a full characterization for arbitrary $n \in \mathbb{N}$ and $\mathbf{y} \in \mathbb{N}^n$ remains elusive, we do so for $n = 2, 3$. Given the technicality of these results, we introduce the notion of *minimally invariant* car lengths, for which the only invariant parking assortment is the all ones preference list. We provide a concise, oracle-based characterization of minimally invariant car lengths for any $n \in \mathbb{N}$. Our results around minimally invariant car lengths also hold for parking sequences.

7.2.4 A Connection to Quicksort

In [40] we consider `Quicksort`, which is a classical divide-and-conquer sorting algorithm. It is a comparison sort that makes an average of $2(n+1)H_n - 4n$ comparisons on an array of size n ordered uniformly at random, where $H_n \coloneqq \sum_{i=1}^{n} \frac{1}{i}$ is the nth harmonic number. Therefore, it makes $n!\,[2(n+1)H_n - 4n]$ comparisons to sort all possible orderings of the array. We prove that this count also enumerates the parking preference lists of n cars parking on a one-way street with n parking spots resulting in exactly $n-1$ lucky cars (i.e., cars that park in their preferred spot). For $n \geq 2$, both counts satisfy the second order recurrence relation $f_n = 2nf_{n-1} - n(n-1)f_{n-2} + 2(n-1)!$ with $f_0 = f_1 = 0$.

BIBLIOGRAPHY

[1] Ayomikun Adeniran and Catherine H. Yan. On increasing and invariant parking sequences. *Australasian Journal of Combinatorics*, 79(1):167–182, 2021.

[2] Ajit Agrawal, Philip Klein, and Ramamoorthi Ravi. When trees collide: An approximation algorithm for the generalized steiner problem on networks. *SIAM journal on Computing*, 24(3):440–456, 1995.

[3] Yasmin Aguillon, Dylan Alvarenga, Pamela E Harris, Surya Kotapati, J Mori, Casandra D Monroe, Zia Saylor, Camelle Tieu, and Dwight Anderson Williams II. On parking functions and the tower of hanoi. *arXiv preprint arXiv:2206.00541*, 2022.

[4] Miklós Ajtai, János Komlós, and Endre Szemerédi. Sorting in clogn parallel steps. *Combinatorica*, 3(1):1–19, 1983.

[5] María J Alonso-González, Theo Liu, Oded Cats, Niels Van Oort, and Serge Hoogendoorn. The potential of demand-responsive transport as a complement to public transport: An assessment framework and an empirical evaluation. *Transportation Research Record*, 2672(8):879–889, 2018.

[6] Javier Alonso-Mora, Samitha Samaranayake, Alex Wallar, Emilio Frazzoli, and Daniela Rus. On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment. *Proceedings of the National Academy of Sciences*, 114(3):462–467, 2017.

[7] Arash Asadpour, Michel X Goemans, Aleksander Mądry, Shayan Oveis Gharan, and Amin Saberi. An o (log n/log log n)-approximation algorithm for the asymmetric traveling salesman problem. *Operations Research*, 65(4):1043–1061, 2017.

[8] Alex Bavelas. Communication patterns in task-oriented groups. *The journal of the acoustical society of America*, 22(6):725–730, 1950.

[9] Xiaohui Bei and Shengyu Zhang. Algorithms for trip-vehicle assignment in ride-sharing. In *AAAI*, volume 18, pages 3–9, 2018.

[10] Lowell W Beineke, Ortrud R Oellermann, and Raymond E Pippert. The average connectivity of a graph. *Discrete mathematics*, 252(1-3):31–45, 2002.

[11] Dimitris Bertsimas, Yee Sian Ng, and Julia Yan. Data-driven transit network design at scale. *Operations Research*, 69(4):1118–1133, 2021.

[12] Dimitris Bertsimas, Yee Sian Ng, and Julia Yan. Joint frequency-setting and pricing optimization on multimodal transit networks at scale. *Transportation Science*, 54(3):839–853, 2020.

[13] Louis J Billera and Aravamuthan Sarangarajan. The combinatorics of permutation polytopes. In *Formal power series and algebraic combinatorics*, volume 24, pages 1–23, 1994.

[14] Geoff Boeing. Osmnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Computers, Environment and Urban Systems*, 65:126–139, 2017.

[15] Ralf Borndörfer, Martin Grötschel, and Marc E Pfetsch. A column-generation approach to line planning in public transport. *Transportation Science*, 41(1):123–132, 2007.

[16] Jarosław Byrka, Fabrizio Grandoni, Thomas Rothvoß, and Laura Sanità. Steiner tree approximation via iterative randomized rounding. *Journal of the ACM (JACM)*, 60(1):1–33, 2013.

[17] Joshua Carlson, Alex Christensen, Pamela E Harris, Zakiya Jones, and Andrés Ramos Rodríguez. Parking functions: Choose your own adventure. *The College Mathematics Journal*, 52(4):254–264, 2021.

[18] Deeparnab Chakrabarty, Jochen Könemann, and David Pritchard. Hypergraphic lp relaxations for steiner trees. *SIAM Journal on Discrete Mathematics*, 27(1):507–533, 2013.

[19] Douglas M Chen, Pamela E Harris, J Carlos Martínez Mori, Eric J Pabón-Cancel, and Gabriel Sargent. Permutation invariant parking assortments. *arXiv preprint arXiv:2211.01063*, 2022.

[20] Miroslav Chlebík and Janka Chlebíková. The steiner tree problem on graphs: Inapproximability results. *Theoretical Computer Science*, 406(3):207–214, 2008.

[21] Nicos Christofides. Worst-case analysis of a new heuristic for the travelling salesman problem. Technical report, Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, 1976.

[22] Data Driven NYC. Reengineering urban transit // Daniel Ramot and Saar Golde, Via (Firstmark's data driven). [YouTube video], April 2017. Accessed Oct. 28, 2020.

[23] Jack Edmonds. Maximum matching and a polyhedron with 0, 1-vertices. *Journal of Research of the National Bureau of Standards B*, 69(125-130):55–56, 1965.

[24] Jack Edmonds. Optimum branchings. *Journal of Research of the National Bureau of Standards, B*, 71:233–240, 1967.

[25] Jack Edmonds. Matroids and the greedy algorithm. *Mathematical programming*, 1(1):127–136, 1971.

[26] Richard Ehrenborg and Alex Happ. Parking cars of different sizes. *The American Mathematical Monthly*, 123(10):1045–1048, 2016.

[27] Joan Feigenbaum, Christos H Papadimitriou, and Scott Shenker. Sharing the cost of multicast transmissions. *Journal of computer and system sciences*, 63(1):21–41, 2001.

[28] Sharon Feigon and Colin Murphy. *Shared mobility and the transformation of public transit*. Transportation Research Board, 2016. TCRP Research Report, Issue Number: 188.

[29] Andreas Emil Feldmann, Jochen Könemann, Neil Olver, and Laura Sanità. On the equivalence of the bidirected and hypergraphic relaxations for steiner tree. *Mathematical programming*, 160(1):379–406, 2016.

[30] Alan M Frieze, Giulia Galbiati, and Francesco Maffioli. On the worst-case performance of some algorithms for the asymmetric traveling salesman problem. *Networks*, 12(1):23–39, 1982.

[31] Philine Gattermann, Jonas Harbering, and Anita Schöbel. Line pool generation. *Public Transport*, 9(1):7–32, 2017.

[32] Michel X Goemans. Smallest compact formulation for the permutahedron. *Mathematical Programming*, 153(1):5–11, 2015.

[33] Michel X Goemans and Dimitris J Bertsimas. Survivable networks, linear programming relaxations and the parsimonious property. *Mathematical programming*, 60(1-3):145–166, 1993.

[34] Michel X Goemans and Young-Soo Myung. A catalog of steiner tree formulations. *Networks*, 23(1):19–28, 1993.

[35] Michel X Goemans, Neil Olver, Thomas Rothvoß, and Rico Zenklusen. Matroids and integrality gaps for hypergraphic steiner tree relaxations. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 1161–1176, 2012.

[36] Michel X Goemans and David P Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24(2):296–317, 1995.

[37] Aric Hagberg, Pieter Swart, and Daniel S Chult. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008.

[38] Jonathan D Hall, Craig Palsson, and Joseph Price. Is uber a substitute or complement for public transit? *Journal of Urban Economics*, 108:36–50, 2018.

[39] Pamela E Harris, Brian M Kamau, J. Carlos Martínez Mori, and Roger Tian. On the outcome map of mvp parking functions: Permutations avoiding 321 and 3412, and motzkin paths. *Enumerative Combinatorics and Applications*, 3(2):Article #S2R11, 2023.

[40] Pamela E Harris, Jan Kretschmann, and J Carlos Martínez Mori. Lucky cars and the quicksort algorithm. *arXiv preprint arXiv:2306.13065*, 2023.

[41] Kamal Jain. A factor 2 approximation algorithm for the generalized steiner network problem. *Combinatorica*, 21(1):39–60, 2001.

[42] Anna R Karlin, Nathan Klein, and Shayan Oveis Gharan. A (slightly) improved approximation algorithm for metric tsp. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 32–45, 2021.

[43] Richard M Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

[44] Anton J Kleywegt, Alexander Shapiro, and Tito Homem-de Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502, 2002.

[45] Jochen Könemann, David Pritchard, and Kunlun Tan. A partition-based relaxation for steiner trees. *Mathematical Programming*, 127(2):345–370, 2011.

[46] Alan G. Konheim and Benjamin Weiss. An occupancy discipline and applications. *Society for Industrial and Applied Mathematics Journal on Applied Mathematics*, 14(6):1266–1274, 1966.

[47] Lawrence Kou, George Markowsky, and Leonard Berman. A fast algorithm for steiner trees. *Acta informatica*, 15(2):141–145, 1981.

[48] Songhua Li, Minming Li, and Victor CS Lee. Trip-vehicle assignment algorithms for ride-sharing. In *International Conference on Combinatorial Optimization and Applications*, pages 681–696. Springer, 2020.

[49] Yining Liu and Yanfeng Ouyang. Mobility service design via joint optimization of transit networks and demand-responsive services. *Transportation Research Part B: Methodological*, 151:22–41, 2021.

[50] Rémi Louf and Marc Barthelemy. A typology of street patterns. *Journal of The Royal Society Interface*, 11(101):20140924, 2014.

[51] Meghna Lowalekar, Pradeep Varakantham, and Patrick Jaillet. Zac: A zone path construction approach for effective real-time ridesharing. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 29, pages 528–538, 2019.

[52] Meghna Lowalekar, Pradeep Varakantham, and Patrick Jaillet. Competitive ratios for online multi-capacity ridesharing. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, pages 771–779, 2020.

[53] Kelin Luo and Frits CR Spieksma. Approximation algorithms for carsharing problems. In *International Computing and Combinatorics Conference*, pages 262–273. Springer, 2020.

[54] Qi Luo, Samitha Samaranayake, and Siddhartha Banerjee. Multimodal mobility systems: joint optimization of transit network design and pricing. In *Proceedings of the ACM/IEEE 12th International Conference on Cyber-Physical Systems*, pages 121–131, 2021.

[55] J. Carlos Martínez Mori and Samitha Samaranayake. Bounded asymmetry in road networks. *Scientific reports*, 9(1):11951, 2019.

[56] J Carlos Martínez Mori and Samitha Samaranayake. Permutatorial optimization via the permutahedron. *Operations Research Letters*, 50(5):441–445, 2022.

[57] J Carlos Martínez Mori, M Grazia Speranza, and Samitha Samaranayake. On the value of dynamism in transit networks. *Transportation Science*, 2023.

[58] J Carlos Martínez Martínez Mori and Samitha Samaranayake. On the request-trip-vehicle assignment problem. In *SIAM Conference on Applied and Computational Discrete Algorithms (ACDA21)*, pages 228–239. SIAM, 2021.

[59] Louis A Merlin. Transportation sustainability follows from more people in fewer vehicles, not necessarily automation. *Journal of the American Planning Association*, 85(4):501–510, 2019.

[60] NYC Taxi and Limousine Commission. TLC trip record data. https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page, 2021. Accessed: February 27, 2021.

[61] Masayo Ota, Huy Vo, Claudio Silva, and Juliana Freire. Stars: Simulating taxi ride sharing at scale. *IEEE Transactions on Big Data*, 3(3):349–361, 2016.

[62] Tobias Polzin and Siavash Vahdati Daneshmand. On steiner trees and minimum spanning trees in hypergraphs. *Operations Research Letters*, 31(1):12–20, 2003.

[63] Richard Rado. An inequality. *Journal of the London Mathematical Society*, 1(1):1–6, 1952.

[64] Prabhakar Raghavan and Clark D Thompson. Randomized rounding: a technique for provably good algorithms and algorithmic proofs. *Combinatorica*, 7(4):365–374, 1987.

[65] Lisa Rayle, Danielle Dai, Nelson Chan, Robert Cervero, and Susan Shaheen. Just a better taxi? a survey-based comparison of taxis, transit, and ridesourcing services in san francisco. *Transport Policy*, 45:168–178, 2016.

[66] Ronald Read and Robin Wilson. *An Atlas of Graphs*. Oxford: Clarendon, 1998.

[67] Connor Riley, Antoine Legrain, and Pascal Van Hentenryck. Column generation for real-time ride-sharing operations. In *International Conference on Integration of Constraint Programming, Artificial Intelligence, and Operations Research*, pages 472–487. Springer, 2019.

[68] Alexander Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *Journal of Combinatorial Theory, Series B*, 80(2):346–355, 2000.

[69] AI Serdyukov. Some extremal bypasses in graphs. *Upravliaemie systemy*, 17:76–79, 1978.

[70] Susan Shaheen and Nelson Chan. Mobility and the sharing economy: Potential to facilitate the first-and last-mile public transit connections. *Built Environment*, 42(4):573–588, 2016.

[71] Andrea Simonetto, Julien Monteil, and Claudio Gambella. Real-time city-scale ridesharing via linear assignment problems. *Transportation Research Part C: Emerging Technologies*, 101:208–232, 2019.

[72] Mitja Stiglic, Niels Agatz, Martin Savelsbergh, and Mirko Gradisar. Enhancing urban mobility: Integrating ride-sharing and public transit. *Computers & Operations Research*, 90:12–21, 2018.

[73] Ola Svensson, Jakub Tarnawski, and László A Végh. A constant-factor approximation algorithm for the asymmetric traveling salesman problem. *Journal of the ACM (JACM)*, 67(6):1–53, 2020.

[74] Robert Vicari. Simplex based steiner tree instances yield large integrality gaps for the bidirected cut relaxation. *arXiv preprint arXiv:2002.07912*, 2020.

[75] Jarrett Walker. *Human transit: How clearer thinking about public transit can enrich our communities and our lives*. Island Press, 2012.

[76] Jarrett Walker. Microtransit: What I Think We Know. Human Transit, https://humantransit.org/2018/02/microtransit-what-i-think-we-know.html, February 2018. Accessed 2021-06-27.

[77] Marla Westervelt, Emma Huang, Joshua Schank, Nolan Borgman, Tamar Fuhrer, Colin Peppard, and Rani Narula-Woods. Uprouted: Exploring microtransit in the United States. Technical report, Eno Center for Transportation, January 2018.

[78] Richard T Wong. A dual ascent approach for steiner tree problems on a directed graph. *Mathematical programming*, 28(3):271–287, 1984.

[79] Catherine H Yan. Parking functions. In *Handbook of Enumerative Combinatorics*, pages 835–894. Chapman and Hall/CRC, 2015.