Line Planning and Connectivity

vorgelegt von
Dipl.-Math. oec Marika Karbstein
Berlin

Von der Fakultät II – Mathematik und Naturwissenschaften
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften
– Dr. rer. nat. –

genehmigte Dissertation

Promotionsausschuss

Vorsitzender:	Prof. Dr. Volker Mehrmann
Berichter:	Prof. Dr. Ralf Borndörfer
	Prof. Dr. Dr. h.c. mult. Martin Grötschel

Tag der wissenschaftlichen Aussprache: 26. März 2013

Berlin 2013
D 83

Zusammenfassung

Diese Arbeit führt das *Steinerzusammenhangsproblem (Steiner connectivity problem)* ein. Es ist eine Verallgemeinerung des sehr bekannten *Steinerbaumproblems*. Beim Steinerbaumproblem wird in einem Graphen $G = (V, E)$ eine kostenminimale Menge von Kanten gesucht, die eine gegebene Teilmenge $T \subseteq V$ von Knoten verbindet. Beim Steinerzusammenhangsproblem wird statt einer kostenminimalen Menge von Kanten eine kostenminimale Menge von Pfaden gesucht, die alle Knoten in T verbindet. Die Pfade werden dabei aus einer gegebenen Menge von Pfaden ausgewählt. Wir zeigen im ersten Teil dieser Arbeit, dass Hauptresultate über Komplexität, Approximierbarkeit, ganzzahlige Formulierungen und Polyeder für das Steinerbaumproblem auf das Steinerzusammenhangsproblem übertragen bzw. erweitert werden können.

Ein Beispiel für eine direkte Übertragung sind *Steinerpartitionsungleichungen*, eine fundamentale Klasse von facettendefinierenden Ungleichungen für das Steinerbaumproblem. Sie können in ähnlicher Weise für das Steinerzusammenhangsproblem definiert werden. Andere Resultate erfordern mehr Aufwand, z.B., das Finden einer gerichteten Schnittformulierung und das Beweisen, dass diese die kanonische ungerichtete Schnittformulierung mit allen Steinerpartitionsungleichungen dominiert. Im Steinerzusammenhangsfall muss man dafür auf *erweiterte Formulierungen* zurückgreifen, ein „Umweg“, der für das Steinerbaumproblem nicht notwendig ist. Es gibt aber auch Resultate, die sich nicht übertragen lassen, z.B., führt der Fall $T = V$ beim Steinerzusammenhangsproblem zu *Set Covering Problemen* und damit nicht zu einem polynomial lösbaren Fall wie beim Steinerbaum.

Das Steinerzusammenhangsproblem ist nicht nur eine interessante Verallgemeinerung des Steinerbaumproblems, sondern auch das grundlegende Zusammenhangsproblem in der Linienplanung mit integriertem Passagierrouting. Das *integrierte Linienplanungs- und Passagierroutingproblem* ist ein wichtiges Problem in der Angebotsplanung des öffentlichen Nahverkehrs und Gegenstand des zweiten Teils dieser Arbeit. Gegeben ist ein Infrastrukturnetzwerk für den öffentlichen Nahverkehr, d. h. Kanten entsprechen Straßen und Schienen und Knoten entsprechen Haltestellen von Linien. Gesucht werden Wege im Infrastrukturnetzwerk für Linien und Passagiere, so dass die Kapazität der Linien ausreicht, eine gegebene Beförderungsnachfrage zu erfüllen. Existierende Modelle, die ein Passagierrouting integrieren, behandeln entweder das Umsteigen von Passagieren zu ungenau und vernachlässigen einen wichtigen Aspekt in der Routenwahl der Passagiere, oder sie behandeln das Umsteigen zu detailliert und sind für große reale Probleme nicht

berechenbar. Wir stellen ein neues Modell vor, das den Fokus auf Direktverbindungen legt. Durch eine Umsteigestrafe für jede nicht-direkte Verbindung, wird die Attraktivität von umsteigefreien Verbindungen erhöht und das Passagierrouting in Richtung auf mehr umsteigefreie Verbindungen gelenkt.

Für die Berechnung des Modells verwendeten wir Algorithmen, die durch Erkenntnisse über das Steinerzusammenhangsproblem beeinflusst sind. Die Ergebnisse zeigen, dass das Modell sehr gute Lösungen berechnet, die eine gewichtete Summe aus den Kosten eines Linienplans und der Gesamtfahrzeit aller Passagiere minimieren. Im Vergleich zu einem existierenden Ansatz, der Direkttverbindungen nicht gesondert betrachtet, ergibt sich eine deutliche Verbesserung von bis zu 17%. Im Gegensatz zu einem Ansatz, in dem alle Umstiege betrachtet werden und für den wir innerhalb von 10 Stunden nicht einmal das Wurzel-LP für große Instanzen lösen konnten, konnten wir mit dem neuen Modell, in der gleichen Zeit, sehr gute Lösungen (Differenz zum Optimum <1%) oder sogar Optimallösungen für reale Instanzen finden. In einem Projekt mit der Verkehrsgesellschaft in Potsdam GmbH zur Berechnung des Linienplans 2010 konnten wir nachweisen, dass unser Verfahren für die Lösung echter Praxisprobleme angewendet werden kann.

Abstract

This thesis introduces the *Steiner connectivity problem*. It is a generalization of the well known *Steiner tree problem*. Given a graph $G = (V, E)$ and a subset $T \subseteq V$ of the nodes, the Steiner tree problem consists in finding a cost minimal set of edges connecting all nodes in T. The Steiner connectivity problem chooses, instead of edges, from a given set of paths a subset to connect all nodes in T. We show in the first part of this thesis that main results about complexity, approximation, integer programming formulations, and polyhedra can be generalized from the Steiner tree problem to the Steiner connectivity problem.

An example for a straightforward generalization are the *Steiner partition inequalities*, a fundamental class of facet defining inequalities for the Steiner tree problem. They can be defined for the Steiner connectivity problem in an analogous way as for the Steiner tree problem. An example for a generalization that needs more effort is the definition of a directed cut formulation and the proof that this dominates the canonical undirected cut formulation enriched by all Steiner partition inequalities. For the Steiner connectivity problem this directed cut formulation leads to *extended formulations*, a concept that is not necessary for the Steiner tree problem. There are also major differences between both problems. For instance, the case $T = V$ for the Steiner connectivity problem is equivalent to a *set covering problem* and, hence, not a polynomial solvable case as in the Steiner tree problem.

The Steiner connectivity problem is not only an interesting generalization of the Steiner tree problem but also the underlying connectivity problem in line planning with integrated passenger routing. The *integrated line planning and passenger routing problem* is an important planning problem in service design of public transport and the topic of the second part. Given is the infrastructure network of a public transport system where the edges correspond to streets and tracks and the nodes correspond to stations/stops of lines. The task is to find paths in the infrastructure network for lines and passengers such that the capacities of the lines suffice to transport all passengers. Existing models in the literature that integrate a passenger routing in line planning either treat transfers in a rudimentary way and, hence, neglect an important aspect for the choice of the passenger routes, or they treat transfers in a too comprehensive way and cannot be solved for large scale real world problems. We propose a new model that focuses on direct connections. The attractiveness of transfer free connections is increased by introducing

a transfer penalty for each non-direct connection. In this way, a passenger routing is computed that favors direct connections.

For the computation of this model we also implemented algorithms influenced by the results for the Steiner connectivity problem. We can compute with our model good solutions that minimize a weighted sum of line operating costs and passengers travel times. These solutions improve the solutions of an existing approach, that does not consider direct connections, by up to 17%. In contrast to a comprehensive approach, that considers every transfer and for which we could not even solve the root LP within 10 hours for large instances, the solutions of the new model, computed in the same time, are close to optimality (<1%) or even optimal for real world instances. In a project with the Verkehr in Potsdam GmbH to compute the line plan for 2010 we showed that our approach is applicable in practice and can be used to solve real world problems.

Acknowledgments

The research for my thesis was conducted within the project *Service Design in Public Transport* which is supported by the DFG research center MATHEON. I greatly benefited from the MATHEON community; this provided me the opportunity to interact with excellent researchers in business meetings as well as at social get-togethers. I am also very grateful to Martin Grötschel and Ralf Borndörfer for the excellent working condition I had at the Zuse Institute Berlin.

I would like to express my sincere and deep gratitude to my supervisor Ralf Borndörfer for his continuous support of my PhD study and research. This thesis would not have been possible without his guidance, encouragement, and immense knowledge. He always took time for discussions, proposed possible solutions whenever I needed help, and pointed out open questions which required further investigations. All this helped to greatly improve this thesis. My sincere thanks also go to Marc Pfetsch who was an important contact for all my questions, especially in the beginning of my research. He established the basic framework for the line optimization tool which saved me a lot of time to implement and test different models and algorithms.

It is a great pleasure for me to work at the optimization department at the Zuse Institute. My colleagues gave me any help I needed. Thank you all for that! I further wish to thank Gregor Karbstein, Christian Raack, Thomas Schlechte, and Kati Wolter for proof-reading and discussing parts of this thesis.

Finally, I thank my family for their patience, encouragement, and support. My special thanks go to my husband Gregor who coped perfectly well with my ups and downs through the duration of my studies.

Contents

Contents ix

Overview 1

I The Steiner Connectivity Problem 9

1 Introduction and Complexity 11
1.1 Problem Description . . . 12
1.1.1 The Degree Property . . . 14
1.1.2 Interpretation in Hypergraphs . . . 17
1.2 Relation to Steiner Trees . . . 19
1.3 Relation to Set Covering . . . 21
1.4 Approximation Results . . . 27
1.4.1 Approximation by Reduction to a Steiner Tree Problem . . . 27
1.4.2 Approximation by Connecting Sets . . . 29
1.4.3 Primal-Dual Approximation Algorithm . . . 33

2 IP Formulations and Polyhedra 39
2.1 IP Formulations . . . 39
2.1.1 Cut Formulation . . . 40
2.1.2 Directed Cut Formulation . . . 43
2.1.3 Contracted Directed Cut Formulation . . . 47
2.2 Polyhedral Analysis . . . 51
2.2.1 Steiner Partition Inequalities . . . 52
2.2.2 Separating Steiner Partition Inequalities . . . 57
2.2.3 k-Terminal Sets Inequalities . . . 59

3 Connecting Sets and Disconnecting Sets 63
3.1 Blocking Pairs . . . 64
3.2 The Max-Flow-Min-Cut Theorem . . . 66
3.3 A Minimum st-Connecting Set Algorithm and a Companion Theorem . . . 68
3.4 Ideal Matrices . . . 72

3.5 Directed Paths . . . 73
3.6 Connectivity Problems w. r. t. Paths . . . 75

4 Solving the Steiner Connectivity Problem 81
4.1 Cutting Planes for the Undirected Cut Formulation . . . 81
4.1.1 Separating Steiner Partition Inequalities . . . 82
4.1.2 Separating Cuts from the Extended Formulation . . . 83
4.1.3 Separating Cuts by Partial Projection . . . 84
4.2 Primal Heuristics . . . 88
4.3 Computational Analysis . . . 89

5 Concluding Remarks for Part I 97

II The Line Planning Problem 99

6 Line Planning Models 101
6.1 Literature Overview . . . 102
6.2 Notation . . . 105
6.3 Basic Dynamic Model . . . 109
6.3.1 Integer Program . . . 109
6.3.2 Pricing Problem . . . 111
6.4 Change-and-Go Model . . . 112
6.4.1 Integer Program . . . 112
6.4.2 Pricing Problem . . . 114
6.5 Direct Connection Models . . . 114
6.5.1 Exact Model – Direct Line Connections . . . 115
6.5.2 Approximation Model I – Relaxing Direct Line Capacities . . . 118
6.5.3 Approximation Model II – Relaxing Direct Connections . . . 122
6.5.4 Model Extension – Unavoidable Transfers . . . 123
6.5.5 Pricing Problems . . . 126
6.6 Model Discussion . . . 130

7 Polyhedral Aspects 135
7.1 The Line Planning Polytope . . . 136
7.1.1 Basic Properties of the Line Planning Polytope . . . 139
7.1.2 Polytope for One Metric Inequality . . . 140
7.2 Band Inequalities . . . 141
7.3 Improving Inequalities by Mixed Integer Rounding . . . 143
7.4 Cutset Inequalities . . . 144
7.5 Steiner Partition Band Inequalities . . . 146

8 Solving the Line Planning Problem 151
8.1 Data and Preprocessing . . . 151
8.2 Primal Heuristics . . . 155

8.3 Cutting Planes . . . 159
8.4 Branching Rule . . . 164
8.5 Computational Comparison of Line Planning Models . . . 165

9 Potsdam 2010 – Line Optimization in Practice 173
9.1 Data . . . 174
9.2 Model Specification . . . 175
9.3 Results . . . 178
9.3.1 Optimization vs. Simulation . . . 179
9.3.2 Cost vs. Travel Time . . . 181
9.3.3 OPT vs. P10 . . . 182
9.4 Conclusion . . . 187

10 Concluding Remarks for Part II 189

A Matheon *Adventskalender* – Exercises 191
A.1 Die Turboschlitten Rettung (2008) . . . 192
A.2 Linienplanung (2009) . . . 196
A.3 Der lange Weg nach Hause (2010) . . . 199
A.4 Gesprächige Wichtel (2011) . . . 203
A.5 Wer war es? (2012) . . . 205

Bibliography 209

Overview

In the year 2009 the S-Bahn Berlin was in big trouble with its vehicles of the series 481. A broken wheel was followed by an extensive safety inspection. As a consequence, only a quarter of the car pool was available for the service. The S-Bahn Berlin had to shorten lines, reduce frequencies of lines, and even stop the service on some routes. The purpose of this so-called "Notfahrplan" [89] was to offer a minimum of service for a set of important stations. But how can we identify the resource-minimal set of lines that connects a set of important stations?

This question can also be put in a broader, in a graph theoretic context by identifying the infrastructure of a public transport system, i. e., the streets and tracks, with a graph and the lines, operating on these streets/tracks, with paths in the graph. Associating costs with the paths, representing a resource usage, e. g., money or number of vehicles for operation, the problem is then to find a cost minimum set of paths such that a (sub)set of nodes are connected. Here, connected means that one can "travel" from one node in the set to any other node in the set along the chosen paths. We denote this problem by the *Steiner connectivity problem*. A very similar problem is the *Steiner tree problem*, a classical and well investigated combinatorial optimization problem. Given a graph with costs on the edges, the problem is to find a cost minimum set of edges that connects a subset of nodes. Steiner trees are fundamental for network design in transportation and telecommunication; see Dell'Amico, Maffioli, and Martello [4] for an overview. In fact, the Steiner tree problem can be seen as the prototype of all problems where nodes are connected by installing capacities on individual edges or arcs. In the same way, the Steiner connectivity problem can be seen as the prototype of all problems where nodes are connected by installing capacities on *paths* which is exactly the case in line planning. Hence, the significance of the Steiner connectivity problem for line planning is similar to the significance of the Steiner tree problem for telecommunication network design. The relation between all four problems is illustrated in Figure 1. The first comprehensive investigation of the Steiner connectivity problem is the topic of the first part of this thesis.

Although the Steiner tree problem and the Steiner connectivity problem look very similar at first glance, it is not possible to generalize all structural results and algorithms from the Steiner tree problem directly to the Steiner connectivity problem. More precisely, an equivalent directed cut formulation as it is proposed by Chopra and Rao [35] for the

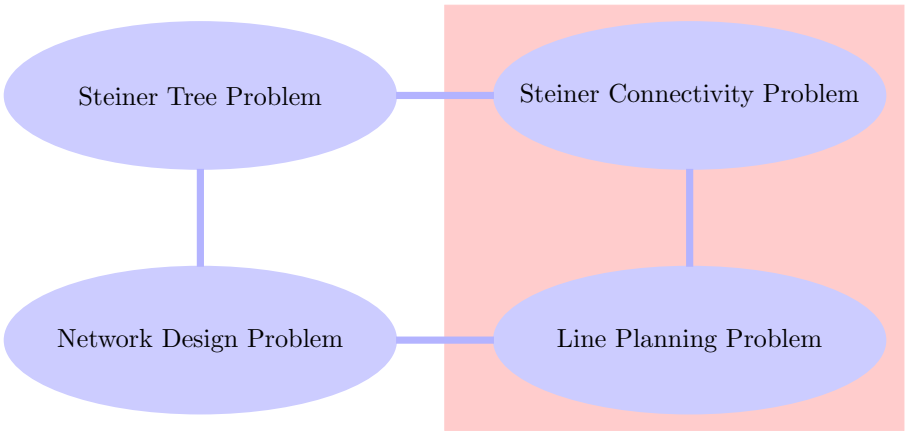


Figure 1: Relations between Steiner tree problem, Steiner connectivity problem, (capacitated) network design problem, and line planning problem. Horizontal edges represent the generalization from edges/arcs (left) to paths (right); vertical edges correspond to the generalization from pure connectivity (top) to connectivity w. r. t. capacity (bottom). The topic of this thesis is the investigation of the Steiner connectivity problem and the line planning problem (with integrated passenger routes).

Steiner tree problem does not exist for the Steiner connectivity problem. Chopra and Rao showed that the LP relaxation of the undirected IP formulation of the Steiner tree problem, including all so-called Steiner partition inequalities, is dominated by a certain family of directed formulations; see also Polzin [79] and Polzin and Daneshmand [82]. For the Steiner connectivity problem, analogous results can instead be derived from an extended formulation based on a suitably constructed directed Steiner tree problem. We show that this formulation is provably strong, including, e. g., a class of facet defining generalized Steiner partition inequalities. It is often too large to be solved directly, however, it can be used to produce a strong relaxation of the Steiner connectivity problem via projection to the original space of variables.

The solution of a Steiner connectivity problem gives a lower bound on the costs of a line plan. Focusing on line paths, it ignores travel times and passenger route choices. The infrastructure of a public transport system usually offers several possibilities to find a route between an origin and a destination, often with similar travel times or lengths, and passengers change their routes not only according to the travel times but also according to the lines operating on these routes. Hence, the passenger routing is an important aspect, that a good line planning method must take into account. Coming back to the S-Bahn Berlin, the "Notfahrplan" of the year 2009 also forced me to use a different route on my daily way to work. Fortunately, the travel time on the new route was similar to that on the old one, and I continued to use the new route when the regular service was reintroduced. But how should a "regular service", a system of lines in public transport, be defined? We will address this question in the second part of the thesis. More precisely, we will consider the *integrated line planning and passenger routing problem.*

An informal description of the line planning problem is as follows: We are given a graph with edges and nodes representing the infrastructure of the public transport system. We

are further given point-to-point demands, i. e., the number of passengers that want to travel from one point in the network to another point. A line is a path in the network, visiting a set of stops/stations in a predefined order. Passengers can travel along these lines and they can change from one line to another line in a stop/station if these lines intersect. Bringing capacities into play, the task is to find a set of lines with associated frequencies of operation such that the capacities of those lines suffice to transport the given travel demand. There are two main objectives for a line plan, namely, minimization of line operation costs and minimization of passenger discomfort measured in, e. g., travel times and number of transfers.

We will introduce and investigate line planning models that integrate a passenger routing, i. e., the passenger routes are computed in dependence of the line plan. Since the number of transfers is an important decision factor for passengers, we will propose a novel *direct connection* approach that encourages transfer free connections. We will show that this approach leads to a computationally tractable line planning method that provides good estimates of transfer times. As far as we know, this is the only line planning model integrating a passenger routing and a transfer handling that allows to solve large scale real world instances. Indeed, in a joint project with the ViP Verkehr in Potsdam GmbH we supported the development of the Potsdam line plan 2010 with mathematical optimization methods. The resulting direct connection line planning model including all practical requirements could be solved to proven optimality. ViP also certified that our computed line plan is indeed practicable. In the end ViP implemented a slightly different variant. The reason was that ViP rated the demand for tram lines higher than for transfer free connections. In fact, our final solution moved some traffic from the tram network to the bus network to offer more transfer free connections which was one important goal for ViP.

Background

The work in this thesis was carried out at the Konrad-Zuse Zentrum für Informationstechnik Berlin (ZIB) in the project "Service Design in Public Transport" [13] supported by the DFG Research Center Matheon *Mathematics for key technologies*. This project deals with public transport planning problems in service design, which are network design, line planning, timetabling, and fare planning, compare with Figure 2. The problem of network design is to specify the infrastructure of a public transport network. Determining the exact arrival and departure times for each line at each station is the problem of timetabling. The fare planning problem investigates the definition of different ticket types and prices. The goal of the project "Service Design in Public Transport" is to advance mathematical optimization methods for service design problems following the example of the subsequent operational planning problems such as vehicle scheduling and duty scheduling. These problems involve the assignment of vehicles and duties/drivers to the line routes.

Indeed, mathematical optimization methods are well established in operational planning

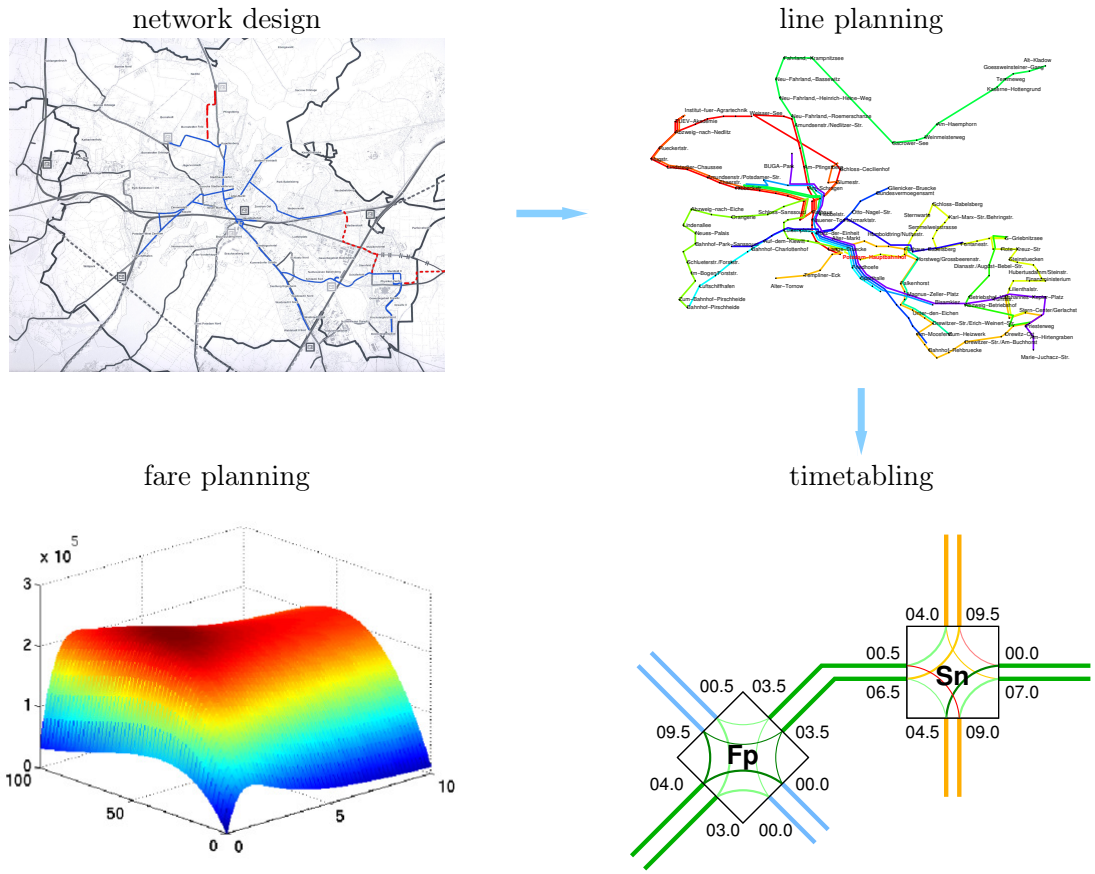


Figure 2: Service design problems in public transport. *Upper left:* Infrastructure network of Potsdam, red are additional streets/tracks that can be established for public transport. *Upper right:* A line plan of Potsdam. *Lower right:* Piece of the Berlin Subway network: two stations and three lines with waiting and transfer times [68]. *Lower left:* Revenue function depending on single ticket and monthly ticket fares for Potsdam; plotted with Matlab [100], see also [19].

and nowadays integrated in planning software for public transport such as IVU.plan [63], see Borndörfer, Grötschel, and Löbel for duty scheduling [14], and Löbel [70] for vehicle scheduling. These problems are even so well understood that approaches to integrate duty scheduling and vehicle scheduling have been successfully analyzed by Borndörfer, Löbel, and Weider [21].

In other areas of traffic planning, the relevance of mathematics also becomes more and more evident. Bussieck, Winter, and Zimmermann [32] discussed the use of mathematical programming methods in public rail transport. Borndörfer, Grötschel, and Jäger [11, 12] describe "the state and the relevance of mathematics in planning and operating public transit" not only in urban traffic but also in rail and air traffic. Two particular examples are given by Borndörfer, Dovica, Nowak, and Schickinger [9] on delay management in airline industries and by Schlechte [90] on track allocation problems in rail traffic.

In public transport service design, however, mathematical optimization methods are

currently rarely used in practice. One reason for this backlog is that the problems in service design are very complex. In contrast to operational planning, in which the objective is mainly cost minimization, service design also has to cover passenger interests like short travel and transfer times, i. e., service design leads naturally to bi- or even multi-criteria optimization problems. Another challenge is the consideration of passenger behavior. Changes in the line plan or timetable surely influence the choice of the route that passengers will use to travel from their origin to their destination in the public transport network. Likewise, changes in the ticket prices may influence the passengers in using public transport or not, i. e., there is a behavioral component in service design. Such aspects could not be handled well until very recently. Three of four contributions have been made within the project "Service Design in Public Transport" and its two predecessor projects "Strategic Planning in Public Transport" at ZIB and "Line Planning and Periodic Timetabling in Railway Traffic" at TU Berlin, where we could show that optimization applications to real world problems are becoming possible also in service design [25]. Liebchen [67] optimized the Berlin subway timetable in 2005; this was the first optimized timetable that has been implemented. Moreover, he extended his timetabling model by vehicle and duty scheduling aspects in a project together with the ViP Verkehr in Potsdam GmbH (ViP) [96]; this was the first integrated treatment of these planning steps [69]. A comprehensive investigation of timetabling can be found in the dissertation of Liebchen [68]. Passenger behavior plays a prominent role in fare planning. We introduced a family of optimization models to compute fares that maximize revenue, demand, or social welfare in [19] and tested them in a pilot project with data of ViP Verkehr in Potsdam GmbH. These models integrate a discrete choice model for the passenger behavior. Although, or maybe because, fares are subject to economic, social, and political interests, we believe that flexible models for mathematical fare optimization can be a valuable decision support tool. We optimized the line plan for the city of Potsdam in 2010 with mathematical optimization methods [24]. To our knowledge, no mathematically computed line plans had been implemented in practice before. In a project with ViP we showed that such an implementation is possible [10].

Main Contribution and Outline of this Thesis

This thesis consists of two parts. The first part investigates the Steiner connectivity problem while the second part covers integrated line planning and passenger routing problems. An overview of the content of the two parts is as follows.

The first part generalizes and extends complexity, approximation, and polyhedral results for the Steiner tree problem to the Steiner connectivity setting. In Chapter 1, we introduce the *Steiner connectivity problem* (SCP) and two special cases, the *minimum spanning set problem* and the *minimum st-connecting set problem*. We point out the relation to the undirected and the directed Steiner tree problem and show that the minimum spanning set problem can also be stated in terms of a (submodular) set covering problem. These relations allow to carry over some complexity results to the Steiner connectivity

problem. In particular, the greedy algorithm gives a logarithmic bound for the minimum spanning set problem, compare with Wolsey [105], which is asymptotically optimal, see Feige [46]. We give a direct and elementary proof for this approximation guarantee. Further results have to be extended to the Steiner connectivity problem, e. g., we show that the primal-dual approximation technique of Goemans and Williams [54] can be applied to the Steiner connectivity problem. The proof of the approximation guarantee is based on a degree property for minimum connected hypergraphs which, as far as we know, was not proven before. Chapter 2 investigates integer programming formulations and polyhedral results for the Steiner connectivity problem. We propose a canonical undirected cut formulation as well as an extended directed cut formulation which is based on the relation of the Steiner connectivity problem to the directed Steiner tree problem shown in the preceding chapter. We generalize the Steiner partition inequalities, a fundamental class of facet defining inequalities for the Steiner tree problem, see Chopra and Rao [35], to the Steiner connectivity problem. Then we state necessary and sufficient conditions for the Steiner partition inequalities to be facet defining. We show that a super class of the Steiner partition inequalities can be separated in polynomial time. In particular, we show that the undirected cut formulation enriched by all Steiner partition inequalities is dominated by the extended directed cut formulation. This shows that extended formulations provide tight relaxations for the SCP. Indeed the extended directed cut formulation for the SCP plays a similar role as the directed cut formulation for the Steiner tree problem, see, e. g., Chopra and Rao [35], Polzin [79] and Polzin and Daneshmand [82], with the exception that the extended formulation for the SCP is not suitable for computations in the same way as the directed cut formulation for the Steiner tree problem, compare with Koch and Martin [66] and the computational results in Chapter 4. In Chapter 3 we emphasize the similarity of the minimum *st*-connecting set problem with the common shortest *st*-path problem and show that properties and duality results on paths and cuts in a graph can be generalized to *connecting* and *disconnecting sets*. More precisely, we show that connecting and disconnecting sets give rise to a blocking pair of ideal matrices just like paths and cuts. Using a relation of the Steiner connectivity problem to hypergraphs this implies that the blocking properties of paths and cuts can be extended to hypergraphs. In this way, it turns out that Menger's theorem and the associated companion theorem also hold for hypergraphs. While the first theorem is folklore, we prove the second theorem by showing that the LP relaxation of the cut formulation for the minimum *st*-connecting set problem is TDI. In this context, we propose an algorithm to solve the minimum *st*-connecting set problem. In the graph case, the duality results and the blocking properties of paths and cuts are also valid if paths and cuts are directed, i. e., for directed graphs. This is not true for connecting and disconnecting sets. We will show this by proving that the directed generalization of the minimum *st*-connecting set problem even becomes $\mathcal{NP}$-hard. We present computational investigations and results for the Steiner connectivity problem in Chapter 4. This involves, e. g., a *partial projection method* to derive strong inequalities including facet defining cuts from a combinatorially motivated subsystem of the extended formulation. We provide a computational comparison of this approach with a shrinking heuristic, which gives rise to a very effective way to improve the canonical undirected cut formulation using Steiner partition inequalities.

The comparison gives evidence that these inequalities close most of the gap between the canonical undirected and the extended directed cut formulation. Chapter 5 concludes the first part by summarizing complexity and polyhedral results for the Steiner connectivity problem including a comparison to the Steiner tree problem.

The second part investigates models and algorithms for the integrated line planning and passenger routing problem. Chapter 6 starts with a literature overview on the line planning problem. We discuss two existing models that consider an integrated line planning and passenger routing, the column generation approach of Borndörfer, Grötschel, and Pfetsch [15] and the change-and-go approach of Schöbel and Scholl [93]. While the column generation model is computable but lacks a sufficient handling of transfers, the change-and-go model considers a detailed treatment of transfers but is of enormous size and computationally hard to handle for real world instances. We propose a novel *direct connection* approach that combines the advantages of both. The idea of this model is to increase the attractiveness of direct connections by penalizing non-direct connections. We first define a model that implements this idea in an exact way. Then we relax and compress the model in order to reduce the number of variables and constraints. Finally, we propose an extension of the direct connection model by incorporating the number of *unavoidable* transfers, i. e., the number of transfers a passenger has to do in any final line plan. The relaxed direct connection model and the extended direct connection model can be seen as a computationally tractable "first order approximation" to the change-and-go model or as a "transfer improvement" of the column generation model. In all cases, a strong LP formulation improves the computation. Therefore, we want to identify additional cutting planes. To this purpose, we analyze the *line planning polytope* in Chapter 7. This polytope is fundamental for all integrated line planning and passenger routing models that were considered in Chapter 6. We investigate dimension, valid inequalities, and a number of facet defining inequalities for this polytope. Here, some results for the Steiner connectivity polytope, that can be seen as an uncapacitated variant of the line planning polytope, can be generalized. In Chapter 8, we provide a computational comparison of the column generation model, the change-and-go model, the relaxed direct connection model, and the extended direct connection model. To this purpose, we have implemented the corresponding branch-and-cut algorithms on the basis of the constraint integer programming framework SCIP [2, 95]. These algorithms include heuristics to find violated inequalities defined in Chapter 7 as well as primal heuristics to find valid solutions. It turns out that the direct connection model can be solved quite as efficiently as the column generation model. The number of direct travelers can be overestimated by the direct connection model. However, our computational results show that the model works well in practice and estimates the number of direct travelers in a quite accurate way. The further improvement via the extended direct connection model is only little while the extended model is harder to solve. On the other hand, it turns out that already the LP relaxation of the change-and-go model cannot be solved within 10 hours for half of our test instances. We, finally, present our most important success, the optimization of a real world line plan, in Chapter 9. In particular, we report on a project in cooperation with the ViP Verkehr in Potsdam GmbH to compute the line plan 2010

for Potsdam. We computed a solution that covered all requirements of ViP and produced a cost reduction of around 4% and a reduction of the perceived travel time of around 6%. ViP confirmed the practicability of our solution but established a slightly different version that shifts some of the traffic from the bus network to the tram network. To our knowledge, this is the first line plan used in practice that was optimized using integer programming methods that integrate line planning and passenger routing. Chapter 10 ends this thesis with concluding remarks for the second part.

We assume that the reader is familiar with basic concepts in polyhedral theory, graph theory, combinatorial optimization, and integer programming. We recommend the book of Grötschel, Lovász, and Schrijver [56] for an introduction to combinatorial optimization. The notation in this thesis mainly follows the notation of this book. Certain special definitions needed in this thesis are stated at the point where they are used.

Part I

The Steiner Connectivity Problem

Chapter 1

Introduction and Complexity

In the first part of this thesis we introduce and analyze the Steiner connectivity problem, a fundamental problem to connect a set of nodes in a graph using a given set of paths. The Steiner connectivity problem generalizes the Steiner tree problem, in which all paths consist of a single edge, in fact, it can be interpreted as a hypergraph version of this problem. It has applications in traffic and telecommunication network design, where the paths correspond to bus, subway, or railway lines and fibers of different technical characteristics, respectively. Although the problem comes up naturally in these ways, it has not been studied much up to now. In essence, all that was known is that some basic Steiner connectivity questions such as the shortest st-connecting set problem can be reduced to the graph theoretic case by simple transformations. But there is a beautiful mathematical structure beyond that. With its own combinatorial flavor, the Steiner connectivity problem allows to generalize almost all results on Steiner trees to its much broader setting. The wider perspective can even lead to new results on well investigated topics, such as the discovery of a companion theorem to Menger's theorem for hypergraphs. Not everything works, though! In contrast to the Steiner tree problem, results do not carry over to the directed case.

In this chapter, we clarify the notation and definitions that are relevant for the first part. We point out relations of the Steiner connectivity problem to the Steiner tree problem, to the set covering problem, and to hypergraphs. These relations yield complexity results and lead to approximation algorithms that can be, in some cases, applied in a straight forward way and, in other cases, generalized or extended to the Steiner connectivity problem and its special cases. Some results of this chapter are published in [20, 26].

The structure of this chapter is as follows. In Section 1.1, we define the Steiner connectivity problem and two special cases including all relevant notation such as connecting sets and disconnecting sets. This section also includes a proof of an important degree lemma and an interpretation of the Steiner connectivity problem in hypergraphs. We point out the relation to Steiner trees in Section 1.2 and the relation to set covering in Section 1.3. Both sections include a complexity discussion of the problem in general

and for some special cases. We end this chapter by presenting approximation results in Section 1.4.

1.1 Problem Description

We consider the following setting. We are given an undirected graph $G = (V, E)$ with no loops, i. e., for each edge $e = (u, v)$ we have $u \neq v$, $u, v \in V$. We denote by $T \subseteq V$ a set of *terminal nodes*. We are further given a set of paths $\mathcal{P}$ in G, where $p \in \mathcal{P}$ is defined as $p = (v_0, e_1, \ldots, e_k, v_k)$ with $v_0, \ldots, v_k \in V$ and $e_i = \{v_{i-1}, v_i\} \in E$, $i = 1, \ldots, k$, for some $k \in \mathbb{N}$. We denote by $V(p) = \{v_0, \ldots, v_k\} \subseteq V$ the set of nodes of p and by $E(p) = \{e_1, \ldots, e_k\} \subseteq E$ the set of edges of p; $V(\mathcal{P}') = \cup_{p \in \mathcal{P}'} V(p)$ and $E(\mathcal{P}') = \cup_{p \in \mathcal{P}'} E(p)$, $\mathcal{P}' \subseteq \mathcal{P}$. We say that an edge $e \in E(p)$ is *covered* by $p \in \mathcal{P}$ and that p *covers* e. It is perfectly possible that e is covered by more than one path. We assume that each edge is covered by at least one path $p \in \mathcal{P}$; otherwise the (uncovered) edge can be removed. If there is no danger of confusion, i. e., if G has no parallel edges, we also define a path according to its nodes, i. e., $p = (v_0, \ldots, v_k)$ and $\{v_{i-1}, v_i\} \in E$, $i = 1, \ldots, k$.

A set $\mathcal{P}' \subseteq \mathcal{P}$ is *T-connecting* if every two nodes in T are connected in the subgraph $H = (V, E(\mathcal{P}'))$. By assumption $\mathcal{P}$ is V-connecting if $H = G$ is connected. The graph $G = (V, E)$ is *path-connected* w. r. t. $\mathcal{P}$ if $\mathcal{P}$ is V-connecting. If the set $\mathcal{P}' \subseteq \mathcal{P}$ is V-connecting we also denote it as a *spanning set*. In the following, we assume that G is connected and, hence, by assumption path-connected.

A set $\mathcal{P}' \subseteq \mathcal{P}$ is *T-disconnecting* if $\mathcal{P} \setminus \mathcal{P}'$ is not a T-connecting set. Let $W \subset V$ with $T \cap W \neq \emptyset$ and $(V \setminus W) \cap T \neq \emptyset$. We call the set $\mathcal{P}_{\delta(W)} := \{p \in \mathcal{P} : \delta(W) \cap E(p) \neq \emptyset\}$ of all paths that cross the cut $\delta(W) = \{e \in E \mid |e \cap W| = 1\}$ at least once a *T-path cut* or a *Steiner path cut*. If $T = \{s, t\}$ we also speak of an *st*-connecting set, an *st*-disconnecting set, and an *st*-path cut. A T-connecting set (T-disconnecting set/T-path cut) $\mathcal{P}' \subseteq \mathcal{P}$ is minimal if there does not exist a T-connecting set (T-disconnecting set/ T-path cut) $\mathcal{P}'' \subseteq \mathcal{P}$ with $\mathcal{P}'' \subsetneq \mathcal{P}'$.

Let $\mathcal{P}_1, \mathcal{P}_2 \subseteq \mathcal{P}$. The sets $\mathcal{P}_1$ and $\mathcal{P}_2$ are *path-disjoint* if $\mathcal{P}_1 \cap \mathcal{P}_2 = \emptyset$, i. e., there exists no $p \in \mathcal{P}$ with $p \in \mathcal{P}_1$ and $p \in \mathcal{P}_2$. The sets $\mathcal{P}_1, \ldots, \mathcal{P}_k \subseteq \mathcal{P}$, $k > 2$, are *path-disjoint* if every two sets $\mathcal{P}_i, \mathcal{P}_j$, $i, j = 1, \ldots, k$, $i \neq j$, are path disjoint.

The *length* $|p| = |E(p)|$ of a path is the number of edges it contains. We denote by $\deg_{\mathcal{P}}(v) = |\{p \in \mathcal{P} : v \in V(p)\}|$ the *path-degree* w. r. t. $\mathcal{P}$ of node $v \in V$. We skip "w. r. t. $\mathcal{P}$" in the notation if there is no danger of confusion.

Figure 1.1 illustrates these definitions.

The following lemma relates disconnecting sets and path cuts.

Lemma 1.1. *Minimal T-disconnecting sets are minimal T-path cuts and vice versa.*

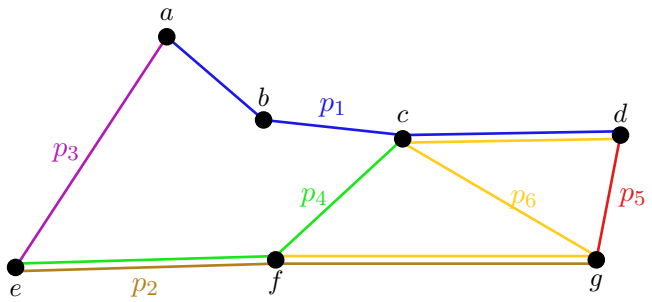


Figure 1.1: Example of a graph and a set of six paths $\mathcal{P} = \{p_1 = (a, b, c, d), p_2 = (e, f, g), p_3 = (a, e), p_4 = (e, f, c), p_5 = (g, d), p_6 = (f, g, c, d)\}$. An example of a de-connecting set is $\{p_1, p_4\}$; a de-disconnecting set is $\{p_2, p_3, p_4\}$. The de-path cut defined by the set $W = \{e\}$ also yields the set $\{p_2, p_3, p_4\}$. The de-connecting sets $\{p_1, p_3\}$ and $\{p_4, p_6\}$ are path-disjoint. Considering the path-degree, we have, e. g., $\deg_{\mathcal{P}}(a) = 2$, $\deg_{\mathcal{P}}(b) = 1$, and $\deg_{\mathcal{P}}(c) = 3$.

Proof. "⇒": Let $\mathcal{P}' \subseteq \mathcal{P}$ be a minimal T-disconnecting set, and let $s, t \in T$ be two terminal nodes that are disconnected. Define W to be the set of nodes reachable from t via $\mathcal{P} \setminus \mathcal{P}'$. Note that $s \notin W$ and $t \in W$, and hence $\mathcal{P}_{\delta(W)}$ is a T-path cut. We claim that $\mathcal{P}_{\delta(W)} = \mathcal{P}'$.

- Assume $p \in \mathcal{P}_{\delta(W)} \setminus \mathcal{P}'$. Hence, p connects some node u in $V \setminus W$ to some node $v \in W$. By definition of W, $\mathcal{P} \setminus \mathcal{P}'$ connects v and t, and since $p \in \mathcal{P} \setminus \mathcal{P}'$ connects u and v, $\mathcal{P} \setminus \mathcal{P}'$ connects u and t. It follows that $u \in W$, a contradiction. Hence, $\mathcal{P}_{\delta(W)} \subseteq \mathcal{P}'$.
- Conversely, assume $p \in \mathcal{P}' \setminus \mathcal{P}_{\delta(W)}$. Since $\mathcal{P}_{\delta(W)} \subseteq \mathcal{P}'$ is a T-disconnecting set, it follows that $\mathcal{P}'$ is not minimal, another contradiction.

Finally, $\mathcal{P}_{\delta(W)}$ is minimal, because otherwise $\mathcal{P}' = \mathcal{P}_{\delta(W)}$ would not be a minimally T-disconnecting set.

"⇐": Let $W \subseteq V$ with $\emptyset \neq W \cap T \neq T$, such that $\mathcal{P}_{\delta(W)}$ is minimal. Then $\mathcal{P}_{\delta(W)}$ is a T-disconnecting set, because no terminal in W is connected to a terminal in $V \setminus W$ via $\mathcal{P} \setminus \mathcal{P}_{\delta(W)}$. We claim that $\mathcal{P}_{\delta(W)}$ is also a minimal T-disconnecting set. Suppose not; then there is some smaller T-disconnecting set $\mathcal{P}' \subsetneq \mathcal{P}_{\delta(W)}$, which we can assume to be minimal. By the forward direction of the proof, $\mathcal{P}' = \mathcal{P}_{\delta(W')}$ for some set $W' \subseteq V$, $\emptyset \neq W' \cap T \neq T$. It follows that $\mathcal{P}_{\delta(W')} = \mathcal{P}' \subsetneq \mathcal{P}_{\delta(W)}$, i. e., $\mathcal{P}_{\delta(W)}$ was not minimal, a contradiction. □

In addition to the graph $G = (V, E)$, the terminal nodes $T \subseteq V$, and the set of paths $\mathcal{P}$, we now consider nonnegative costs $c \in \mathbb{R}^{\mathcal{P}}_{\geq 0}$ for the paths. This allows to optimize over T-connecting sets.

Definition 1.2. *The* Steiner connectivity problem *(SCP) is to find a T-connecting set $\mathcal{P}' \subseteq \mathcal{P}$ of minimum cost, i. e., $c(\mathcal{P}') = \min_{\tilde{\mathcal{P}} \subseteq \mathcal{P}} \sum_{p \in \tilde{\mathcal{P}}} c_p$.*
For the special cases $T = V$ and $T = \{s, t\}$, we denote the Steiner connectivity problem also as the minimum spanning set problem *and the* minimum *st*-connecting set problem, *respectively.*

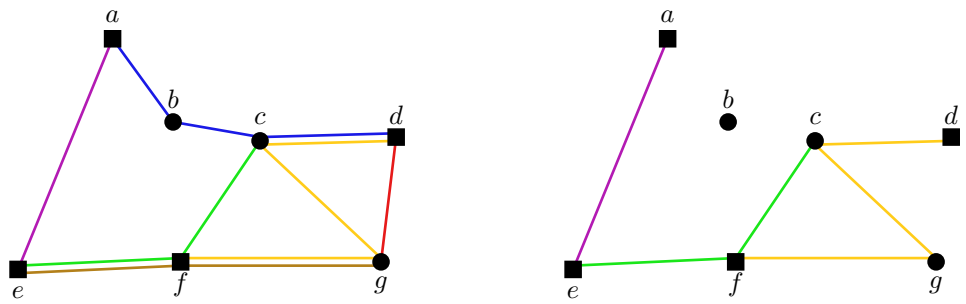


Figure 1.2: *Left:* Example of an SCP with four terminal nodes $T = \{a, d, e, f\}$ and six paths $(\mathcal{P} = \{p_1 = (a, b, c, d), p_2 = (e, f, g), p_3 = (a, e), p_4 = (e, f, c), p_5 = (g, d), p_6 = (f, g, c, d)\})$. *Right:* An inclusion wise minimal solution.

An instance of the Steiner connectivity problem and an inclusion wise minimal solution are shown in Figure 1.2. Note that the graph induced by an inclusion wise minimal solution must not be a tree.

The Steiner connectivity problem generalizes the Steiner tree problem since the Steiner tree problem is the special case where all paths have length one, i. e., correspond to exactly one edge. On the other hand, the Steiner connectivity problem can also be transformed into a directed Steiner tree problem. We consider this transformation in Section 1.2. This relation yields a polynomial approximation result for a fixed number of terminal nodes, i. e., for $|T| = k$. The minimum spanning set problem, i. e., the SCP in which all nodes are terminal nodes, has a strong relation to the (submodular) set covering problem. This relation implies an $\mathcal{NP}$-hardness result and a harmonic approximation guarantee for a Greedy algorithm. We will investigate this relation in more detail in Section 1.3. In Section 1.4 we consider approximation algorithms for the general case.

In the remainder of this section we will prove a degree property which is needed to show the approximation guarantee for a primal-dual approximation algorithm to be developed in Section 1.4 and we point out relations of the Steiner connectivity problem to hypergraphs.

1.1.1 The Degree Property

If the length of all paths in $\mathcal{P}$ is 1, i. e., the paths correspond to edges, a minimal V-connecting set is a spanning tree in G. A tree has $|V|$ nodes and $|V| - 1$ edges, and each edge is incident to exactly two nodes. The average node degree in a tree is therefore $\frac{2(|V|-1)}{|V|} \leq 2 - \frac{2}{|V|} \leq 2$, i. e., at most 2. It is well known that this bound also holds for the average degree of a terminal node in an inclusion wise minimal Steiner tree, because each non-terminal node has degree at least 2. This basic property of Steiner trees generalizes to minimal T-connecting sets.

Lemma 1.3 (Degree Lemma). *The average path-degree of a terminal node w. r. t. an inclusion wise minimal T-connecting set $\mathcal{P}'$ is at most $(k + 1)$, where k denotes the minimum of*

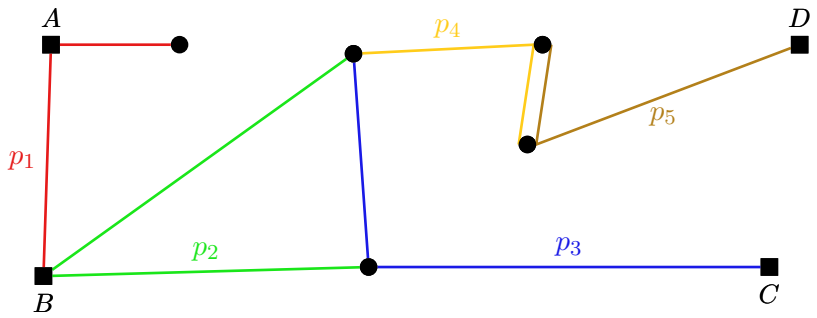


Figure 1.3: Given is an inclusion wise minimal T-connecting set for $T = \{A, B, C, D\}$. The proof of Lemma 1.3 considers only p_1, p_2, p_3, p_5. A starting order can be $\mathcal{P}'(T) = \{(p_1, p_2, p_3), (p_5)\}$ which is also a final order. We have $T_1 = T_2 = \{A, B\}$, $T_3 = \{A, B, C\}$, $T_4 = \{A, B, C, D\}$. The path p_1 gives rise to case 1., while p_2 comes in case 2. Paths p_2 and p_3 constitute a pair.

(a) the maximal number of edges in a path,
(b) the maximal number of terminal nodes in a path.

More precisely, we have

$$\sum_{t \in T} \deg_{\mathcal{P}'}(t) \leq (k+1)(|T|-1), \qquad k = \min\{\max_{p \in \mathcal{P}} |p|, \max_{p \in \mathcal{P}} |T \cap V(p)|\}.$$

Proof. We only consider paths that contain at least one terminal node since these are the only paths that contribute to the path-degree of the terminal nodes. Denote the set of these paths by $\mathcal{P}'(T)$. The idea of the proof is to consider these paths in such a sequence that either each path or a pair of paths establishes a connection to some new terminal. Reaching all terminals then requires at most $|T|-1$ such paths or pairs of paths. This gives rise to a sum of path-degrees at the terminal nodes of at most $(|T|-1)(k+1)$. The details of this argument are as follows. Define a starting order on $\mathcal{P}'(T)$ as follows.

$$\mathcal{P}'(T) = \{p_1, \ldots, p_n\} = \{p_1^1, \ldots, p_{s_1}^1, \ldots, p_1^\ell, \ldots, p_{s_\ell}^\ell\},$$

where

- $V(p_i^j) \cap (\cup_{r=1}^{i-1} V(p_r^j)) \neq \emptyset$, $j = 1, \ldots, \ell$, $i = 2, \ldots, s_j$, and
- $p_i^j \cap p_{\tilde{i}}^{\tilde{j}} = \emptyset$, for all $j \neq \tilde{j}$, $i = 1, \ldots, s_j$, $\tilde{i} = 1, \ldots, s_{\tilde{j}}$,

i. e., the graph induced by the paths $p_1^j, \ldots, p_i^j$, $i \leq s_j$, $j \in \{1, \ldots, \ell\}$, is connected and there is no connection to the graph induced by the paths $p_1^{\tilde{j}}, \ldots, p_{\tilde{i}}^{\tilde{j}}$, $\tilde{i} \leq s_{\tilde{j}}$, $\tilde{j} \in \{1, \ldots, \ell\}$, $j \neq \tilde{j}$.

We define $T_i = \cup_{j=1}^{i} (V(p_i) \cap T)$ to be the set of terminal nodes that are covered by $p_1, \ldots, p_i$. We have $T_i \subseteq T_{i+1}$, $i = 1, \ldots, n-1$, and $T_i = T_{i+1}$ is also possible. Figure 1.3 shows the notation of this proof on an example.

Let $r_1 \geq 1$ be the number of terminal nodes contained in path p_1, i. e., $r_1 = |T_1|$. For $i \geq 2$ let $r_i = |T_i \setminus T_{i-1}|$ be the number of additional terminal nodes contained in p_i, i. e.,

terminal nodes not contained in T_{i-1}. Then we have one of the following two cases:

1. $r_i \geq 1$; then the maximum number of terminal nodes from the set T_{i-1} contained in path p_i is
   - the minimum of $|T_{i-1}|-1 = (\sum_{j=1}^{i-1} r_j)-1$ and $k+1-r_i$, if k is the maximum length of the paths, or
   - the minimum of $|T_{i-1}| - 1 = (\sum_{j=1}^{i-1} r_j) - 1$ and $k - r_i$, if k is the maximum number of terminal nodes.

   In both cases, p_i increases the sum of the path-degrees of all terminal nodes by at most r_i plus the minimum of k and $(\sum_{j=1}^{i-1} r_j) - 1$.
2. $r_i = 0$, i. e., p_i contains a subset of terminal nodes of T_{i-1}. Note that the order of the paths implies that the terminal nodes in p_i are connected by a subset of the paths $p_1, \ldots, p_{i-1}$.
   Due to minimality, there has to exist a path p_h, $h > i$, with $V(p_i) \cap V(p_h) \neq \emptyset$ such that p_h adds $r_h \geq 1$ new terminal nodes and covers no terminal nodes of T_i, i. e., p_i and p_h have a non-terminal node in common. Move path p_h to position $i+1$. Both paths, p_i and p_h, increase the sum of the path-degree of the terminal nodes by at most r_h plus the minimum of $\{|T_i| - 1 = (\sum_{j=1}^{i-1} r_j) - 1,\ k\}$:
   - If k is the maximum path-length, p_i contains at most k terminal nodes since it has a non-terminal node in common with p_h.
   - If k is the maximum number of terminal nodes in a path, the statement above is also true.

The (final) order of the set $\mathcal{P}'(T)$ yields m paths and pairs of paths, respectively, that increase the path-degree on all terminal nodes by at most $\overline{r}_i + \min\{k, (\sum_{j=1}^{i-1} \overline{r}_j) - 1\}$, $\overline{r}_i \geq 1$, $i = 1, \ldots, m \leq n$. Let $1 \leq j \leq m$ be an index such that (i) and (ii) below are satisfied (if $\sum_{i=1}^{m} \overline{r}_i \leq k$, the Lemma holds trivially); clearly (iii) – (v) also hold.

(i) $\sum_{i=1}^{j} \overline{r}_i \geq k+1$,
(ii) $\sum_{i=1}^{j-1} \overline{r}_i \leq k$,
(iii) $\overline{r}_1 + \ldots + \overline{r}_m = |T|$,
(iv) $m \leq |T|$,
(v) $j \leq k+1$.

We can then bound the sum of the path-degrees on all terminal nodes as follows:

$$
\begin{aligned}
\sum_{t \in T} \deg_{\mathcal{P}'}(t) &= \sum_{t \in T} \deg_{\mathcal{P}'(T)}(t) \\
&\leq \overline{r}_1 + (\overline{r}_2 + \min\{k, \overline{r}_1 - 1\}) + \ldots + (\overline{r}_m + \min\{k, (\sum_{i=1}^{m-1} \overline{r}_i) - 1\}) \\
&\leq \overline{r}_1 + \overline{r}_2 + (\overline{r}_1 - 1) + \ldots + \overline{r}_j + (\overline{r}_1 + \ldots \overline{r}_{j-1} - 1) + \overline{r}_{j+1} + k + \ldots \overline{r}_m + k \\
&\overset{\text{(i)-(iii)}}{\leq} |T| + (\overline{r}_1 - 1) + \ldots + (\overline{r}_1 + \ldots + \overline{r}_{j-1} - 1) + (m-j)k \qquad (1.1) \\
&\overset{\text{(ii),(iv)}}{\leq} |T| + k(j-1) - 1 \cdot (j-1) + (|T| - j)k
\end{aligned}
$$

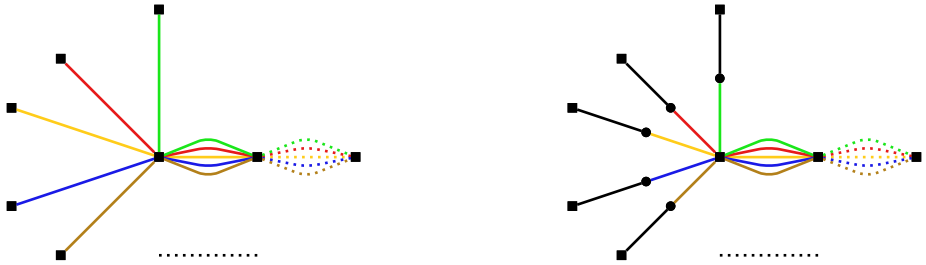


Figure 1.4: Worst case example for the Degree Lemma 1.3, case (a) (*left*) and case (b) (*right*).

$$\begin{aligned} &= && |T| + jk - k - j + 1 + |T|k - jk \\ &\text{if } j \geq 2 \\ &\leq && |T| - 1 + |T|k - k = (|T| - 1)(k + 1). \end{aligned}$$

For $j = 1$ we have $m - 1 \leq (|T| - \overline{r}_1) \leq (|T| - 2)$ since $\overline{r}_1 = k + 1 \geq 2$ and, therefore,

$$\begin{aligned} (1.1) = |T| + (m-1)k &\leq |T| + (|T| - 2)k \\ &\leq |T| + |T|k - k - 1 = (|T| - 1)(k + 1). \end{aligned}$$

□

We briefly show that this bound is tight for case (a) and (b) of Lemma 1.3. Consider the instance in the left of Figure 1.4. All nodes are terminal nodes. We have n nodes in the rim and k nodes in the middle. Suppose each path contains one node of the rim and all nodes in the middle. All paths together form a minimal V-connecting set. We have n nodes with path-degree 1 and k nodes with path-degree n, i. e., the total degree is $n(k+1)$, which gives an average path-degree of $\frac{n(k+1)}{(n+k)}$. This is arbitrarily close to $(k+1)$ as n goes to infinity. This instance can be slightly modified to get a worst case example for case (b), compare with the right of Figure 1.4. We have n additional non-terminal nodes in the inner rim and each path contains a non-terminal node in the inner rim and all nodes in the middle. We further have n paths that connect the outer rim with the inner rim. The maximal number of terminal nodes in a path is k and we have the same path degree for each terminal node as for the case (a) above.

1.1.2 Interpretation in Hypergraphs

Path-connectivity can also be studied in terms of hypergraphs by interpreting each path $p \in \mathcal{P}$ as a hyperedge in a suitably constructed hypergraph $\mathcal{H}$, namely the undirected *hypergraph* $\mathcal{H} = (V, \mathcal{E})$ with the same node set as G and hyperedge set $\mathcal{E} = \{V(p) : p \in \mathcal{P}\}$. Our setting translates as follows. Let s and t be two different vertices of $\mathcal{H}$. An st-path q in $\mathcal{H}$ is an alternating sequence of mutually different nodes v_{i_h}, $h = 0, \ldots, k$, and mutually distinct hyperedges e_{j_h}, $h = 1, \ldots, k$, such that $v_{i_{h-1}}, v_{i_h} \in e_{j_h}$ for all $h = 1, \ldots, k$, $v_{i_0} = s$, and $v_{i_k} = t$. If such a path exists, we call s and t connected in $\mathcal{H}$. We call the set of edges $\mathcal{E} \cap q$ an *st-hyperpath*. A set of hyperedges $\mathcal{E}' \subseteq \mathcal{E}$ is called an *st-hypercut* if s and t are connected in $\mathcal{H} = (V, \mathcal{E})$, but not in $\mathcal{H}' = (V, \mathcal{E} \setminus \mathcal{E}')$, see, e. g.,

Frank [51]. It is easy to see that an st-connecting set in G corresponds to an st-hyperpath in $\mathcal{H}$ while an st-disconnecting set in G corresponds to an st-hypercut in $\mathcal{H}$. Let $T \subseteq V$ and $c_e \geq 0$ for all $e \in \mathcal{E}$. The problem to find a cost minimal set of hyperedges $\mathcal{E}' \subseteq \mathcal{E}$ such that every two nodes $s, t \in T$ are connected in $\mathcal{H}' = (V, \mathcal{E}')$ is then equivalent to the Steiner connectivity problem.

To our knowledge, there is no systematic investigation of the Steiner connectivity problem in the hypergraph literature. We are only aware of results for the special cases $|T| = 2$ and $T = V$.

The case $|T| = 2$ corresponds to an undirected shortest path problem in hypergraphs. This problem has not received much attention, probably because it can be easily transformed to a shortest path problem in common graphs: Each hyperedge is substituted by a complete graph with edge cost corresponding to the cost of the hyperedge. While this is true, it it not everything that can be said. We will, e. g., present in Chapter 3 a shortest path algorithm in the original graph which considers each path exactly once. This algorithm is more efficient than the above described transformation method. Moreover, a primal-dual version of this algorithm can be used to prove a companion theorem to Menger's theorem for hypergraphs. These results have not been mentioned in the literature before. The directed version of the shortest path problem for hypergraphs has been studied extensively, e. g., by Nguyen and Pallottino [75] and Gallo, Longo, and Pallottino [53]. However, the definitions for directed hypergraphs differ in the literature and do not fit into our context, e. g., Gallo, Longo, and Pallottino [53] define a directed hyperedge as a set of source nodes that are connected to a set of tail nodes.

The case $|T| = V$ corresponds to what is called a *minimum spanning set problem in hypergraphs* by Baudis et al. [6]. It is closely related to the set covering problem, see Section 1.3. The spanning tree problem in hypergraphs, however, is defined a little bit differently. Here, the task is to find a set of hyperedges $\mathcal{E}'$ that connects all nodes such that every two hyperedges $e_1, e_2 \in \mathcal{E}'$ can have at most one node in common, i. e., $|e_1 \cap e_2| \leq 1$. This makes the problem hard, namely, even the question whether a hypergraph contains a spanning tree is $\mathcal{NP}$-hard, see, e. g., Warme [103]. We assume that a Steiner tree in hypergraphs would be defined similarly, i. e., every two hyperedges can only intersect in at most one node, although we could not find any references in the literature. The closest hits are the papers by Warme [103] and Polzin and Daneshmand [81] who consider spanning trees in hypergraphs in the context of geometric Steiner trees in the plane.

The degree property, Lemma 1.3, can also be interpreted in a straight forward way in the context of hypergraphs. Case (a) of Lemma 1.3 has been mentioned in a paper of Takeshita, Fujito, and Watanabe [64], written in Japanese, who used this property to derive an approximation result for a primal-dual algorithm, compare also with Section 1.4.3. We could, however, not find a proof for this claim. In fact, an inquiry with the authors and several other persons in the hypergraph community revealed that there is none published. Case (b) of the Degree Lemma is new and extends the result. We have therefore added a self-contained proof of the Degree Lemma in Subsection 1.1.1.

In what follows, we will not use a hypergraph notation, but stick to the notation of Section 1.1, since this fits better with our line planning application.

1.2 Relation to Steiner Trees

We study in this section the complexity of the Steiner connectivity problem. The SCP is a generalization of the Steiner tree problem and therefore $\mathcal{NP}$-hard in general. However, we will show that it is also equivalent to a suitably constructed directed Steiner tree problem. This relation exhibits a number of polynomially solvable cases.

Assume we are given a graph $G = (V, E)$, terminal nodes $T \subseteq V$, and costs on the edges $c \in \mathbb{R}^E_{\geq 0}$. Then the (undirected) *Steiner tree problem* is to find a set of edges $E' \subseteq E$ with minimum cost such that all terminal nodes T are connected by E'. The Steiner tree problem is strongly $\mathcal{NP}$-hard, see Prömel and Steger [82].

Observation 1.4. *If $|p| = 1$ for all $p \in \mathcal{P}$, the Steiner connectivity problem is exactly the Steiner tree problem, i. e., the Steiner connectivity problem is strongly $\mathcal{NP}$-hard in general.*

The *directed Steiner tree problem* (DSTP) is the following: Given is a directed graph with costs on the arcs, a set of terminal nodes T, and a root node $r \in T$. We have to find a minimum cost set B of arcs that connects the root node to each other terminal node $t \in T\backslash\{r\}$, i. e., there exists a directed path from r to t in B. If the costs of the arcs are nonnegative, which we assume, there exists a solution that is a directed tree (an arborescence).

Consider an SCP with undirected graph $G = (V, E)$, a set of paths $\mathcal{P}$, terminal nodes $T \subseteq V$, and nonnegative costs $c \in \mathbb{R}^{\mathcal{P}}_{\geq 0}$. Define nodes v_p, w_p for each path $p \in \mathcal{P}$ and a digraph $D' = (V', A')$, which we call *Steiner connectivity digraph*. Its node set is

$$V' := T \cup \{v_p, w_p : p \in \mathcal{P}\}.$$

We choose some terminal node $r \in T$ as a root node and define the following arcs $a \in A'$ and costs c'_a:

$$\begin{array}{lll}
a = (r, v_p), & c'_a := 0, & \forall\, p \in \mathcal{P} \text{ with } r \in V(p), \\
a = (v_p, w_p), & c'_a := c_p, & \forall\, p \in \mathcal{P}, \\
a = (w_{\tilde{p}}, v_p), & c'_a := 0, & \forall\, p, \tilde{p} \in \mathcal{P},\ r \notin p,\ p \neq \tilde{p},\ \ p \text{ and } \tilde{p} \text{ have} \\
 & & \quad \text{a node } v \in V \text{ in common}, \\
a = (w_p, t), & c'_a := 0, & \forall\, p \in \mathcal{P},\ \forall\, t \in T\backslash\{r\} \text{ with } t \in V(p).
\end{array}$$

Figure 1.5 illustrates our construction. Note that choosing different root nodes results in different Steiner connectivity digraphs and hence different associated DSTPs. However, we will show in Proposition 1.6 that the solutions of an SCP and any associated DSTP are all equivalent, independent of the choice of the root node. For ease of notation, we

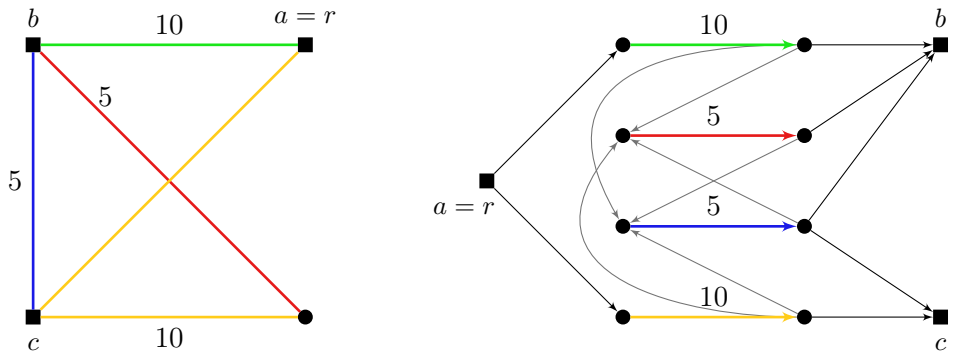


Figure 1.5: A Steiner connectivity problem and an associated directed Steiner tree problem. *Left:* Graph G with four paths and three terminal nodes. The numbers on the paths indicate costs. *Right:* Associated Steiner connectivity digraph D' for root node $r = a$. The numbers on the arcs are the costs; the default value is zero.

will therefore omit the root node from the notation whenever the results are independent of r. Polyhedral results can depend on the choice of the root node, see Remark 2.9 in Chapter 2. In such cases we will include the root node in the notation.

Observation 1.5. *The Steiner connectivity digraph $D' = (V', A')$ has the following properties:*

1. *The only arc with target node w_p is (v_p, w_p), for all $p \in \mathcal{P}$.*
2. *The only arc with source node v_p is (v_p, w_p), for all $p \in \mathcal{P}$.*
3. *If $r \in p$ then the only arc with target node v_p is (r, v_p).*
4. *Each simple directed st-path has the form $(s, v_{p_1}, w_{p_1}, \ldots, v_{p_k}, w_{p_k}, t)$, $k \geq 1$.*

Proposition 1.6. *The following holds for an SCP and an associated DSTP: For each solution of one problem there exists a solution of the other problem with the same objective value. In particular, the optimal objective value of an associated DSTP is independent of the choice of the root node.*

Proof. Assume $\tilde{\mathcal{P}}$ is a solution of SCP. Then let

$$\tilde{A} := A' \setminus \{(v_p, w_p) : p \notin \tilde{\mathcal{P}}\}.$$

The arcs in $\tilde{A}$ connect the root r with each terminal $t \in T \setminus \{r\}$ via a directed path. Moreover, $\sum_{a \in \tilde{A}} c'_a = \sum_{p \in \tilde{\mathcal{P}}} c'_{v_p w_p} = \sum_{p \in \tilde{\mathcal{P}}} c_p$.

For the converse, assume that $\tilde{A}$ is a solution of the DSTP. We show that

$$\tilde{\mathcal{P}} := \{p \in \mathcal{P} : (v_p, w_p) \in \tilde{A}\}$$

is a solution of the corresponding SCP with the same cost. To this purpose, consider the root node r and some terminal $t \in T \setminus \{r\}$; these nodes are connected by a simple directed path in D' using only arcs in $\tilde{A}$. Each such path has the form $(r, v_{p_1}, w_{p_1}, \ldots, v_{p_k}, w_{p_k}, t)$, $k \geq 1$ (see Observation 1.5), with $(v_{p_i}, w_{p_i}) \in \tilde{A}$, $i = 1, \ldots, k$, that is, $p_i \in \tilde{\mathcal{P}}$, $i = 1, \ldots, k$.

Due to the construction of D', p_1 contains r, p_i and p_{i+1}, $i = 1, \ldots, k-1$, have at least one node in common, and p_k contains t. Hence, we can find a path from r to t in G that is covered by $p_1, \ldots, p_k \in \tilde{\mathcal{P}}$. Since the paths are undirected, every two terminal nodes $t_1, t_2 \in T$, $t_1, t_2 \neq r$, can be connected via r, i. e., $\tilde{\mathcal{P}}$ connects T. Furthermore, $\sum_{p \in \tilde{\mathcal{P}}} c_p = \sum_{p \in \tilde{\mathcal{P}}} c'_{v_p w_p} = \sum_{a \in \tilde{A}} c'_a$.

These arguments hold for every root node. □

Corollary 1.7. *The SCP is solvable in polynomial time for $|T| = k$, k constant.*

Proof. This follows from the complexity results for the directed Steiner tree problem, see Feldman and Ruhl [47]. □

Corollary 1.8. *The minimum st-connecting set problem is solvable in polynomial time.*

In fact, the minimum *st*-connecting set problem can be solved by a directed shortest path computation in the Steiner connectivity digraph. As an alternative, we will consider a shortest *st*-connecting set algorithm in the original graph, i. e., without transforming the problem, in Section 3.3.

1.3 Relation to Set Covering

We have seen that the Steiner connectivity problem has a strong relation to the (directed) Steiner tree problem. However, a main difference is the complexity of these problems if all nodes are terminal nodes. The Steiner tree problem then becomes a minimum spanning tree problem which is polynomially solvable. In contrast to that the minimum spanning set problem is $\mathcal{NP}$-hard.

Proposition 1.9. *The minimum spanning set problem, i. e., the SCP for $T = V$, is $\mathcal{NP}$-hard, even for unit costs.*

Proof. We reduce the set covering problem to the minimum spanning set problem. In a set covering problem we are given a finite set S and a set $\mathcal{M} \subseteq 2^S$. The problem is to find a subset $\mathcal{M}' \subseteq \mathcal{M}$ of minimal cardinality $|\mathcal{M}'|$, such that for all $s \in S$ there exists an $M \in \mathcal{M}'$ with $s \in M$.

Given a set covering instance, we define a minimum spanning set instance in a graph $G = (V, E)$ as follows: The nodes are $V = S \cup \{v\} = T$ with v being one extra node. Let us write $V = \{s_0, s_1, s_2, \ldots\}$, where $v = s_0$. All nodes are terminal nodes. We first assume that G is a complete graph and later remove all edges that are not covered by paths after their construction. For each set $M \in \mathcal{M}$ order the elements in M arbitrarily and construct a path beginning in node v and passing through all nodes of M in the given order. The cost of each such path is 1. Figure 1.6 illustrates the construction.

It is easy to see that a cover $\mathcal{M}'$ with at most k elements exists if and only if a set of paths exists that is V-connecting with cost at most k, $k \geq 0$. □

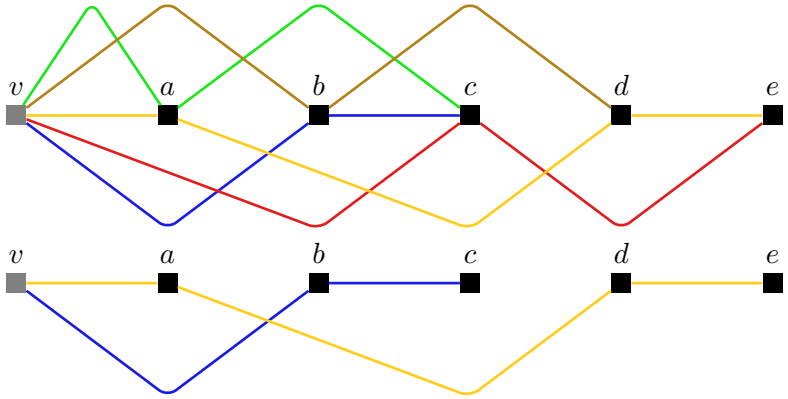


Figure 1.6: *Top:* A Steiner connectivity instance associated with a set covering instance with $S = \{a, b, c, d, e\}$ and $\mathcal{M} = (\{a, c\}, \{b, d\}, \{b, c\}, \{c, e\}, \{a, d, e\})$. *Bottom:* A minimal solution for the Steiner connectivity problem corresponding to the minimal cover $\mathcal{M}' = (\{b, c\}, \{a, d, e\})$.

Corollary 1.10. *The SCP is strongly $\mathcal{NP}$-hard for $|T| = |V| - k$, k constant.*

Proof. We add k isolated nodes to the graph G in the proof of Proposition 1.9. □

Proposition 1.11. *Unless $\mathcal{P} = \mathcal{NP}$, there exists no polynomial time α-approximation algorithm for the SCP with $\alpha = \gamma \cdot \log |V|$, $\gamma \leq 1$.*

Proof. The transformation in Proposition 1.9 is approximation preserving, since there exists a cost preserving bijection between the solutions of a set covering instance and its corresponding Steiner connectivity instance. It has been shown that the set covering problem is not approximable in the sense that there exists no polynomial time approximation algorithm with approximation factor smaller than logarithmic (in the number of nodes) unless $\mathcal{P} = \mathcal{NP}$, see Feige [46]. □

The proof of Proposition 1.9 shows that the set covering problem can be transformed to the minimum spanning set problem. On the other hand, the minimum spanning set problem can be interpreted as a submodular set covering problem.

Definition 1.12. *Let $N = \{1, \ldots, n\}$ and $z : 2^N \to \mathbb{R}$ be a nondecreasing, submodular set function, i. e.,*

$$\begin{aligned} z(A) &\leq z(B) && \forall\, A \subseteq B \subseteq N && \textit{(nondecreasing)} \\ z(A) + z(B) &\geq z(A \cup B) + z(A \cap B) && \forall\, A, B \subseteq N && \textit{(submodular).} \end{aligned}$$

Then

$$\min_{S \subseteq N} \{ \sum_{j \in S} c_j \ : \ z(S) = z(N) \}$$

is a submodular set covering problem. *We call the submodular set covering problem* integer-valued *if $z : 2^N \to \mathbb{Z}$.*

Theorem 1.13 (Wolsey [105], 1982). *There exists a greedy heuristic that gives an* $H(k) = \sum_{i=1}^{k} \frac{1}{i}$ *approximation guarantee for integer-valued submodular set covering problems with* $k = \max_{j \in N} z(\{j\}) - z(\emptyset)$.

Let $N = \mathcal{P}$ and define for $\mathcal{P}' \subseteq \mathcal{P}$

$$z(\mathcal{P}') = |V| - \text{number of connected components in } (V, E(\mathcal{P}'));$$

$z(\mathcal{P}')$ can be interpreted as the maximum number of edges in $(V, E(\mathcal{P}'))$ containing no cycle. Note that this definition corresponds to the rank function for an edge set in a graphical matroid, i. e., $z(\mathcal{P}') = \text{rank}(E(\mathcal{P}'))$, see, e. g., Oxley [77]. The function z is, therefore, a nondecreasing, integer-valued, submodular set function; this follows since $E(\mathcal{P}') \subseteq E(\mathcal{P}'')$ for $\mathcal{P}' \subseteq \mathcal{P}''$. Note that $z(\mathcal{P}') = z(N) = z(\mathcal{P}) = |V| - 1$ means that $\mathcal{P}'$ connects V. Hence, the Steiner connectivity problem can be seen as an integer valued submodular set covering problem. We have $z(p) = |p|$ for $p \in \mathcal{P}$ and $z(\emptyset) = 0$. Therefore, Wolsey's greedy algorithm gives an approximation guarantee of $H(k) = \sum_{i=1}^{k} \frac{1}{i}$ for the minimum spanning set problem if all paths contain at most k edges. This logarithmic bound is also asymptotically optimal, see Feige [46] and compare with Proposition 1.11. In the following, we will consider the greedy algorithm for the minimum spanning set problem in detail and give a direct and constructive proof of Theorem 1.13 for our case. This proof is inspired by the one of Chvátal [36] who showed that a greedy algorithm gives an $H(k) = \sum_{i=1}^{k} \frac{1}{i}$ approximation guarantee for the set covering problem, where k is the largest column sum.

The proof analyzes the greedy Algorithm 1.1. This procedure starts in an initial state in which each single node forms a smallest possible *connected component*. The algorithm then chooses in each iteration a path that minimizes the ratio of cost over the number of components that are connected by the path minus one. These connected components are merged into a new connected component. The algorithm terminates when all nodes have been merged into a single connected component.

We use the following notation. Let B^i be the set of connected components and $\mathcal{P}^i$ the set of chosen paths after iteration i of Algorithm 1.1. Note that in each iteration at least two connected components are merged, i. e., $|B^i|$ decreases strictly with increasing i. Let us further denote by

$$\begin{aligned} N(p,i) &= z(\mathcal{P}^{i-1} \cup \{p\}) - z(\mathcal{P}^{i-1}) = \text{rank}(E(\mathcal{P}^{i-1} \cup \{p\})) - \text{rank}(E(\mathcal{P}^{i-1})) \\ &= \text{no. of conn. comp. in } (V, E(\mathcal{P}^{i-1})) - \text{ no. of conn. comp. in } (V, E(\mathcal{P}^{i-1} \cup \{p\})) \end{aligned}$$

the *component reduction number* of path p and iteration i, i. e., if p were chosen in iteration i, the total number of connected components would reduce by $N(p,i)$. Note that $N(p,i)$ is nonincreasing for increasing i, i. e., $N(p,1) \geq \ldots \geq N(p,n)$ where n is the last iteration of Algorithm 1.1. Let $\mathcal{P}' = \{p(1), \ldots, p(n)\}$. Algorithm 1.1 then computes a solution of cost $c(\mathcal{P}') = \sum_{i=1}^{n} c_{p(i)}$. Let, further, $\mathcal{P}_{\text{opt}} = \{o_1, \ldots, o_m\}$ be an optimal V-connecting set. Finally, we denote by $H(k) = \sum_{i=1}^{k} \frac{1}{i}$ the sum of the first k terms of the harmonic series.

Algorithm 1.1: Greedy heuristic for the SCP

Input : A connected graph $G = (V, E)$, a set of paths $\mathcal{P}$ with costs $c \in \mathbb{R}^{\mathcal{P}}_{\geq 0}$.
Output: A set of paths $\mathcal{P}' \subseteq \mathcal{P}$ that connects all nodes.

$B^0 := \{\{v\} \mid v \in V\}$, $\mathcal{P}^0 := \emptyset$, $i := 1$
while $|B^{i-1}| > 1$ **do**
  $p(i) := \operatorname{argmin}_{p \in \mathcal{P}} \{\frac{c_p}{N(p,i)} : N(p,i) > 0\}$
  $\mathcal{P}' := \mathcal{P}^i := \mathcal{P}^{i-1} \cup \{p(i)\}$
  $B^i := (B^{i-1} \setminus \{b_1, \ldots, b_j\}) \cup \{b_1 \cup \ldots \cup b_j\}$ with $\{b_1, \ldots, b_j\} := \{b \in B^{i-1} : p(i) \in \mathcal{P}_{\delta(b)}\}$
  $i := i + 1$
end

In order to analyze the greedy algorithm, we derive a lemma concerning the sum of the component reduction numbers of the optimal paths in iteration $i \in \{1, \ldots, n\}$. This number is always greater or equal to the sum of the component reduction numbers of the paths that are chosen by the greedy algorithm.

Lemma 1.14. *In Algorithm 1.1 holds*

$$\sum_{o \in \mathcal{P}_{\text{opt}}} N(o, i) \geq \sum_{j=i}^{n} N(p(j), j) \quad \forall\, i = 1, \ldots, n. \tag{1.2}$$

Proof. Consider the right hand side of inequality (1.2). We get

$$\begin{aligned}\sum_{j=i}^{n} N(p(j), j) &= z(\mathcal{P}^{i-1} \cup \{p(i)\}) - z(\mathcal{P}^{i-1}) + z(\mathcal{P}^{i} \cup \{p(i+1)\}) - z(\mathcal{P}^{i}) \\ &\quad + \ldots + z(\mathcal{P}^{n-1} \cup \{p(n)\}) - z(\mathcal{P}^{n-1}) \\ &= z(\mathcal{P}^{n}) - z(\mathcal{P}^{i-1}) = |V| - 1 - z(\mathcal{P}^{i-1}) \\ &= \text{no. of conn. comp. in } (V, E(\mathcal{P}^{i-1})) - 1\end{aligned}$$

The claim then follows since each V-connecting set has to connect all connected components in $(V, E(\mathcal{P}^{i-1}))$. □

Proposition 1.15. *The greedy Algorithm 1.1 gives an $H(k)$ approximation guarantee for Steiner connectivity problems, where $k = \max_{p \in \mathcal{P}} |p|$ is the maximum path length, i. e.,*

$$c(\mathcal{P}') \leq \sum_{p \in \mathcal{P}_{\text{opt}}} H(|p|) c_p \leq H(k)\, c(\mathcal{P}_{\text{opt}}).$$

Proof. The idea of the proof is as follows. In a first step (assignment), we assign the path $p(i) \in \mathcal{P}'$ (added to $\mathcal{P}'$ in iteration $i = 1, \ldots, n$ in Algorithm 1.1) to a subset of optimal paths $O(i) \subseteq \mathcal{P}_{\text{opt}}$. In a second step (bounding), we show that the cost of path $p(i)$ can be bounded from above by the cost of the paths in $O(i)$. In a third step (summation),

Algorithm 1.2: Assigning optimal paths to the paths of the greedy algorithm.

$\upsilon(o,i) := 0, \forall\, o \in \mathcal{P}_{\text{opt}}, \forall\, i = 1,\dots,n$

for $i = n$ to 1 **do**
  $O(i) := \emptyset$, $z := 0$
  while $z < N(p(i),i)$ **do**
    choose $o \in \mathcal{P}_{\text{opt}} \setminus O(i)$ with $N(o,i) - \sum_{j=i}^{n} \upsilon(o,j) > 0$
    $\upsilon(o,i) := \min\{N(o,i) - \sum_{j=i}^{n} \upsilon(o,j), N(p(i),i) - z\}$
    $z := z + \upsilon(o,i)$
    $O(i) := O(i) \cup \{o\}$
  end
end

we show that the cost of each path of the optimal solution $\mathcal{P}_{\text{opt}}$ is used at most $H(k)$ times in the bounding step.

1. Step: Assignment. Consider Algorithm 1.2. It assigns with each path $p(i)$, $i = 1,\dots,n$, of the greedy algorithm, passed in reverse order, a set $O(i) \subseteq \mathcal{P}_{\text{opt}}$ of optimal paths. The component reduction value $N(p(i),i)$ for each path $p(i)$, $i = 1,\dots,n$, is distributed to the paths $o \in O(i)$. To this purpose values $\upsilon(o,i)$ are computed such that $\upsilon(o,i) > 0 \Leftrightarrow o \in O(i)$. More precisely, in each iteration i a set $O(i) \subseteq \mathcal{P}_{\text{opt}}$ is chosen such that

$$\sum_{o \in O(i)} \upsilon(o,i) = N(p(i),i) \quad \forall\, i = 1,\dots,n. \tag{1.3}$$

Here, the values $\upsilon(o,i)$, $o \in O$, $i = 1,\dots,n$, satisfy the following condition.

$$\sum_{j=i}^{n} \upsilon(o,j) \le N(o,i) \quad \forall\, o \in O(i),\ i = 1,\dots,n. \tag{1.4}$$

Lemma 1.14 ensures that these values $\upsilon(o,i)$, $i = 1,\dots,n$, exist.

2. Step: Bounding. Consider the path $p(i)$, $i \in \{1,\dots,n\}$, in iteration i of Algorithm 1.1 and the corresponding set $O(i) = \{o_1,\dots,o_h\}$ defined in Algorithm 1.2. Path $p(i)$ achieves the minimum in the ratio test in Step 3 of Algorithm 1.1. Using this fact and equation (1.3), the cost of $p(i)$ can be bounded as follows

$$\left.\begin{array}{l}
\left.\begin{array}{l}
\frac{c_{p(i)}}{N(p(i),i)} \le \frac{c_{o_1}}{N(o_1,i)} \\
\vdots \\
\frac{c_{p(i)}}{N(p(i),i)} \le \frac{c_{o_1}}{N(o_1,i)}
\end{array}\right\} \upsilon(o_1,i) \text{ times} \\
\vdots \\
\left.\begin{array}{l}
\frac{c_{p(i)}}{N(p(i),i)} \le \frac{c_{o_h}}{N(o_h,i)} \\
\vdots \\
\frac{c_{p(i)}}{N(p(i),i)} \le \frac{c_{o_h}}{N(o_h,i)}
\end{array}\right\} \upsilon(o_h,i) \text{ times}
\end{array}\right\} N(p(i),i) \text{ times.}$$

Hence, we have $c_{p(i)} \leq \sum_{o \in O(i)} \frac{c_o}{N(o,i)} v(o,i)$.

3. Step: Summation. We finally consider how often the costs of a path $o \in \mathcal{P}_{\text{opt}}$ are used in the bounding step. We have $N(p,i) \in \{0,1,2,\ldots,|p|\}$, $\forall p \in \mathcal{P}$, $i = 1,\ldots,n$, hence, the total cost for a path $o \in O \subseteq \mathcal{P}$ in the bounding step can be rewritten as

$$\sum_{i=1}^{n} \frac{c_o}{N(o,i)} v(o,i) = \frac{c_o}{1} a_1 + \frac{c_o}{2} a_2 + \ldots + \frac{c_o}{|o|} a_{|o|}. \tag{1.5}$$

Here, the coefficients

$$a_k = \sum_{\substack{i=1 \\ N(o,i)=k}}^{n} v(o,i), \qquad k = 1,\ldots,|o|$$

are sums of the values $v(o,i)$. Note that $N(o,i)$ is nonincreasing for increasing i. Let $s_k \in \{1,\ldots,n\}$ be the smallest iteration index of Algorithm 1.1 such that $N(o,s_k) = k$, $k = 1,\ldots,|o|$, (for $k = |o|$ we have $s_k = 1$), if such index exists. Then equation (1.4) implies

$$a_k \leq \sum_{j=1}^{k} a_j \leq k, \quad k = 1,\ldots,|o|. \tag{1.6}$$

This follows immediately if $a_k = 0$, i. e., if $N(o,i) \neq k$ for all $i = 1,\ldots,n$. Otherwise

$$a_k \leq \sum_{j=1}^{k} a_j = \sum_{j=1}^{k} \sum_{\substack{i=1 \\ N(o,i)=k}}^{n} v(o,i) = \sum_{i=s_k}^{n} v(o,i) \leq N(o,s_k) = k.$$

The term $\frac{c_o}{k}$ decreases with increasing k and is maximal for $k = 1$, and (1.6) implies $a_1 \leq 1$. This means that the sum (1.5) is maximal for $a_1 = 1$. Repeating this argument for a_2, etc., the sum (1.5) is maximal if all coefficients a_k, $k = 1,\ldots,|o|$, are 1. We then get

$$\sum_{i=1}^{n} \frac{c_o}{N(o,i)} v(o,i) \leq \frac{c_o}{1} + \frac{c_o}{2} + \ldots + \frac{c_o}{|o|} \leq c_o H(|o|).$$

Putting everything together, we get

$$\begin{aligned} c(\mathcal{P}') = \sum_{i=1}^{n} c_{p(i)} &\leq \sum_{i=1}^{n} \sum_{o \in O(i)} \frac{c_o}{N(o,i)} v(o,i) \\ &\overset{v(o,i)=0 \text{ if } o \notin O(i)}{=} \sum_{o \in \mathcal{P}_{\text{opt}}} \sum_{i=1}^{n} \frac{c_o}{N(o,i)} v(o,i) \leq \sum_{o \in \mathcal{P}_{\text{opt}}} H(|o|) c_o \leq H(k) c(\mathcal{P}_{\text{opt}}). \end{aligned}$$

□

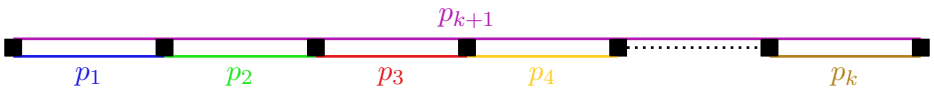


Figure 1.7: Worst case example for the greedy algorithm.

Figure 1.7 shows a worst case example for the greedy heuristic. We have k paths p_i consisting of one edge with cost $c_{p_i} = \frac{1}{i}$, $i = 1, \ldots, k$, and one path consisting of k edges with cost $c_{p_{k+1}} = 1 + \epsilon$, $\epsilon > 0$. The greedy algorithm takes the paths $p_1, \ldots, p_k$ in reverse order producing a total cost of $H(k)$. The optimal solution contains only path p_{k+1} with a cost of $1 + \epsilon$ which can be arbitrarily close to 1.

1.4 Approximation Results

In this section, we will investigate approximation algorithms for the (general) Steiner connectivity problem. A natural idea is to adapt approximation algorithms for the Steiner tree problem to the Steiner connectivity problem. We will pursue this idea in two ways: In Subsection 1.4.1, we will transform a Steiner connectivity problem into a (not equivalent) Steiner tree problem which gives a lower bound on the optimal solution value of the SCP. An upper bound on the optimal solution can then be obtained by applying an approximation algorithm for the Steiner tree problem and transforming the solution back into a solution for the Steiner connectivity problem. This turns an α-approximation algorithm for the Steiner tree problem into a $k\alpha$-approximation algorithm for the SCP, where k is the maximum path length. In Subsections 1.4.2 and 1.4.3, we will generalize approximation algorithms for the Steiner tree problem to the Steiner connectivity problem. More precisely, in Subsection 1.4.2 we will construct a "minimum connecting set tree" heuristic for the Steiner connectivity problem similar to a "minimum length spanning tree" heuristic for the Steiner tree problem. This yields a $(k + 1)$-approximation algorithm, again for a maximum path length of k. In Subsection 1.4.3 we will apply a general primal dual approximation technique of Goemans and Williamson [54] to the Steiner connectivity problem. This yields a $(k+1)$-approximation algorithm for the case that k is the maximum path length and for the case that k is the maximum number of terminal nodes per path.

1.4.1 Approximation by Reduction to a Steiner Tree Problem

The idea is to transform the Steiner connectivity problem into a Steiner tree problem, construct a Steiner tree, and to transform this solution back into a solution for the original Steiner connectivity problem.

Algorithm 1.3 gives a detailed description of this idea. A Steiner tree instance is constructed by setting an edge cost ω_e for each edge in the network as follows: For all paths containing this edge take the minimum ratio of path cost divided by path length. The

Algorithm 1.3: An approximation algorithm for the Steiner connectivity problem using an algorithm for the Steiner tree problem.

Input : A connected graph $G = (V, E)$, a set of paths $\mathcal{P}$ with costs $c \in \mathbb{R}^{\mathcal{P}}_{\geq 0}$, a set of terminal nodes $T \subseteq V$, an STP algorithm AlgoSTP.

Output: A T-connecting set $\mathcal{P}' \subseteq \mathcal{P}$.

`// define edge cost` ω_e `for all` $e \in E$

for all $e \in E$ **do**
  $p = \operatorname{argmin} \{\frac{c_p}{|p|} \mid p \in \mathcal{P}, e \in E(p)\}$
  $\omega_e := \frac{c_p}{|p|}$
  $P(e) := p$
end

`// compute Steiner tree`

$E' \leftarrow \text{AlgoSTP}(G, T, \omega)$

$\mathcal{P}' = \{P(e) : e \in E'\}$

path for which the minimum ratio is achieved is associated with the edge, ties are broken arbitrarily. Then a Steiner tree is computed by the routine AlgoSTP(G, T, ω). All paths associated with the edges of the Steiner tree form a T-connecting set, i. e., a solution for the Steiner connectivity problem.

Proposition 1.16. *Given a Steiner connectivity instance, let $\mathcal{P}_{\text{opt}}$ be a minimum cost T-connecting set and $\mathcal{P}'$ a T-connecting set computed by Algorithm 1.3. Let further k be the maximum path length for all $p \in \mathcal{P}$ and AlgoSTP(g, T, ω) be an α-approximation algorithm for the Steiner tree problem with $\alpha \geq 1$. Then*

$$c(\mathcal{P}') \leq \alpha k \, c(\mathcal{P}_{\text{opt}}).$$

Proof. Let E_{opt} be the edge set of an optimal Steiner tree on the graph $G = (V, E)$ with terminal nodes $T \subseteq V$ and edge costs $\omega_e \geq 0$, $e \in E$. Then we get

$$c(\mathcal{P}_{\text{opt}}) = \sum_{p \in \mathcal{P}_{\text{opt}}} c_p = \sum_{p \in \mathcal{P}_{\text{opt}}} \sum_{e \in E(p)} \frac{c_p}{|p|} \geq \sum_{e \in E(P_{\text{opt}})} \omega_e \geq \sum_{e \in E_{\text{opt}}} \omega_e = \omega(E_{\text{opt}}).$$

Note that $E(P_{\text{opt}})$ induces a Steiner tree.

Since AlgoSTP(g, T, ω) is an α-approximation algorithm for the Steiner tree problem, we have $\omega(E') \leq \alpha\omega(E_{\text{opt}})$. We finally get

$$c(\mathcal{P}') \leq \sum_{e \in E'} c(P(e)) = \sum_{e \in E'} \omega_e |P(e)| \leq \sum_{e \in E'} k\omega_e = k \, \omega(E') \leq \alpha k \, \omega(E_{\text{opt}}) \leq \alpha k \, c(\mathcal{P}_{\text{opt}}).$$

The first inequality is an equality if each $e \in E'$ corresponds to a different $p \in \mathcal{P}'$. □

Corollary 1.17. *The routine AlgoSTP(g, T, ω) in Algorithm 1.3 can be defined such that Algorithm 1.3 yields the following approximation results*

$$\begin{aligned} c(\mathcal{P}') &\leq k \, c(\mathcal{P}_{\text{opt}}) && \text{if } T = V, \\ c(\mathcal{P}') &\leq 1.55 \, k \, c(\mathcal{P}_{\text{opt}}) && \text{otherwise} \end{aligned}$$

Algorithm 1.4: A connecting set heuristic for the Steiner connectivity problem.

Input : A connected graph $G = (V, E)$, a set of paths $\mathcal{P}$ with costs $c \in \mathbb{R}^{\mathcal{P}}_{\geq 0}$, a set of terminal nodes $T \subseteq V$.
Output: A T-connecting set $\mathcal{P}' \subseteq \mathcal{P}$.

```
1  P^0 := ∅, choose r_0 ∈ T, j := 0.
2  Let V(P) as defined on page 12 with the exception of V(∅) := {r_0}
3  while |T \ V(P^j)| > 0 do
4  |   j := j + 1
5  |   Q_j = argmin {c(Q) : Q ⊆ P is vr-connecting for v ∈ V(P^{j-1}), r ∈ T \ V(P^{j-1})}
6  |   P^j := P^{j-1} ∪ Q_j
7  end
8  P' := P^j
9  for all p ∈ P' do
10 |   if P' \ p is T-connecting then
11 |   |   P' := P' \ p // deleting step
12 |   end
13 end
```

with k being the maximum path length for all $p \in \mathcal{P}$.

Proof. The best known approximation factor for the Steiner tree problem is $1 + \frac{\ln 3}{2} \approx 1.55$, see Robins and Zelikovsky [87]. □

1.4.2 Approximation by Connecting Sets

Corollary 1.8 expounds that the minimum st-connecting set problem is solvable in polynomial time. This result can be used to construct an approximation algorithm that performs a minimum st-connecting set computation in each step. More precisely, a starting graph containing an arbitrarily chosen terminal node is grown in each step by connecting at least one terminal node not covered so far via a connecting set with minimum cost. In this way, a $(k+1)$-approximation result (with k being the maximal length of a path) can be derived with similar arguments as for the analogous heuristic for the Steiner tree problem, see, e. g., Takahashi and Matsuyama [99].

Algorithm 1.4 starts with an empty set of paths $\mathcal{P}^0$ and iteratively adds paths to $\mathcal{P}^0$ until all terminal nodes are covered by these paths. The first terminal node r_0 is chosen arbitrarily. In the first iteration, a terminal node $r_1 \neq r_0$ is chosen such that the r_0r_1-connecting set $\mathcal{Q}_1$ has minimum cost; $\mathcal{Q}_1$ and $\mathcal{P}^0$ are then united to $\mathcal{P}^1$. In each following iteration $j \geq 2$ a terminal node $r_j \notin V(\mathcal{P}^{j-1}) \cap T$ is chosen such that a vr_j-connecting set $\mathcal{Q}_j$ has minimum cost over all nodes $v \in V(\mathcal{P}^{j-1})$; again $\mathcal{Q}_j$ and $\mathcal{P}^{j-1}$ are united to $\mathcal{P}^j$.

Proposition 1.18. *Given a Steiner connectivity instance, let $\mathcal{P}_{\text{opt}}$ be the minimum cost*

T-connecting set and $\mathcal{P}'$ a T-connecting set computed with Algorithm 1.4. Then

$$c(\mathcal{P}') \leq (k+1)\, c(\mathcal{P}_{\text{opt}})(1 - \tfrac{1}{|T|}),$$

i. e., Algorithm 1.5 is a $(k+1)$-approximation algorithm with k being the maximal number of edges in a path.

Proof. Consider an arbitrary inclusion wise minimal T-connecting set $\mathcal{P}^\star \subseteq \mathcal{P}$ and recursively construct a *path-tree* as follows. Define as $G_0 := (V(\mathcal{P}^\star), E(\mathcal{P}^\star))$ the graph that is spanned by $\mathcal{P}^\star$. Let v_0 be the root node in the path-tree, i. e., the node at level 0. Node v_0 represents G_0.

Take any $p_0 \in \mathcal{P}^\star$. Since $\mathcal{P}^\star$ is minimally connected, the graph $(V(\mathcal{P}^\star), E(\mathcal{P}^\star \setminus \{p_0\}))$ decomposes into

1. a number of connected components G_i, $i = 1, \ldots, m_1$, spanned by disjoint path sets $\mathcal{P}_i^\star$, i. e., $G_i = (V(\mathcal{P}_i^\star), E(\mathcal{P}_i^\star))$ and $\dot\cup_{i=1}^{m_1} \mathcal{P}_i^\star \dot\cup \{p_0\} = \mathcal{P}^\star$,
2. a number of isolated terminal nodes t_i, $i = 1, \ldots, m_2$,
3. a number m_3 of isolated non-terminal nodes.

Add nodes v_i, $i = 1, \ldots, m_1 + m_2$, representing the components G_i and t_i to the path-tree. These are the nodes of the first level. Connect each node v_i of the first level to the node v_0 of level 0; denote by $p(v_i) = p_0$ the path that establishes this connection. Apply this procedure recursively to G_i by taking a path $p_i \in \mathcal{P}_i^\star$ for each $i = 1, \ldots, m_1$ such that $V(p_i) \cap V(p(v_i)) \neq \emptyset$. Then the graph $G_i - p_i := (V(\mathcal{P}_i^\star), E(\mathcal{P}_i^\star \setminus \{p_i\}))$ decomposes in the same way as $(V(\mathcal{P}^\star), E(\mathcal{P}^\star \setminus \{p_0\}))$. For each connected component of $G_i - p_i$ we add a node in the second level of the path-tree and connect these nodes to the node representing G_i. Proceed with the nodes (and corresponding connected components) of the second level in the same way as with the nodes of the first level. A visualization of the construction is shown in Figure 1.8.

Note that the decomposition of each graph yields $m_1 + m_2 + m_3 \geq 1$ in the three above defined cases and that only m_1 nodes get children. Since the set of paths is finite the procedure terminates. If all paths in $\mathcal{P}$ have at most k edges, every node in the path-tree has at most $k+1$ children, and the edges connecting a node to its children all correspond to a single path. The leafs of the tree correspond to the terminal nodes.

Traverse the tree in inorder and (re-)number the terminals $t_1, \ldots, t_n$ with $n = |T|$. Connect terminals t_i, t_{i+1} via a path $p(t_i, t_{i+1})$ in the path-tree as illustrated by the gray arcs in Figure 1.8, taking indices $i + 1 \bmod n$. Each path $p(t_i, t_{i+1})$ corresponds to a (not necessarily minimal) $t_i t_{i+1}$-connecting set $\mathcal{P}_{t_i, t_{i+1}}$ in the original graph, compare with Figure 1.8. The path set $\{p(t_i, t_{i+1}), i = 1, \ldots, n\}$ contains each path $p \in \mathcal{P}^\star$ at most $k + 1$ times, hence,

$$\sum_{i=1}^{n} c(\mathcal{P}_{t_i, t_{i+1}}) \leq (k+1) c(\mathcal{P}^\star). \tag{1.7}$$

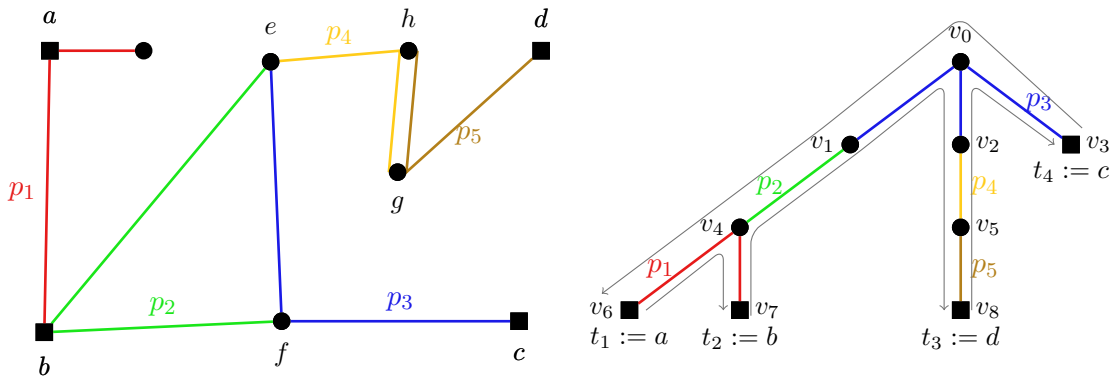


Figure 1.8: Construction of a path-tree as described in the proof of Proposition 1.18. *Left:* A minimal $\{a, b, c, d\}$-connecting set. *Right:* The corresponding path-tree when path p_3 is chosen first. The graph then decomposes into two connecting components (one spanned by p_1 and p_2, the other spanned by p_4 and p_5) and the terminal node c. The choice for all subsequent paths is unique. The path $p(b, d) = \{b, v_4, v_1, v_0, v_2, v_5, d\}$ on the right corresponds to the bd-connecting set $\mathcal{P}_{b,d} = \{p_1, p_2, p_3, p_4, p_5\}$ on the left which is not minimal since $\{p_2, p_4, p_5\}$ is also a bd-connecting set.

This is true for each inclusion wise minimal T-connecting set $\mathcal{P}^\star$ and, therefore, also for $\mathcal{P}_{\text{opt}} = \mathcal{P}^\star$.

Now consider Algorithm 1.4. It starts with terminal r_0 and connects in each iteration at least one terminal node. Let the terminals $r_0, r_1, \dots, r_{n-1}$ be ordered in the same way as they are connected in Algorithm 1.4. If several terminals are connected in one iteration then this subset is ordered arbitrarily. Assume r_i, $i = 1, \dots, n-1$, is connected in iteration j, then we define

$$\mathcal{P}_{r_{i-1},r_i} := \begin{cases} \mathcal{Q}_j & \text{if } r_{i-1} \text{ is connected in iteration } j-1, \\ \emptyset & \text{otherwise, i. e., } r_{i-1} \text{ is also connected in iteration } j. \end{cases}$$

Note that r_0 is "connected in iteration 0". In this way, each set $\mathcal{Q}_j$ in the algorithm is associated with exactly one $\mathcal{P}_{r_{i-1},r_i}$ for $i \in \{1, \dots, n-1\}$. We define the distance between two (terminal) nodes or a set of (terminal) nodes as follows

$$\text{dist}_{\mathcal{P}}(\{r_0, \dots, r_{i-1}\}, r_i) := \text{argmin } \{c(\mathcal{Q}) : \mathcal{Q} \subseteq \mathcal{P} \text{ is } vr_i\text{-connecting for } v \in \{r_0, \dots, r_{i-1}\}\}.$$

Then we have (compare with line 5 of Algorithm 1.4)

$$c(\mathcal{P}_{r_{i-1},r_i}) \leq \text{dist}_{\mathcal{P}}(\{r_0, \dots, r_{i-1}\}, r_i), \quad i = 1, \dots, n-1. \tag{1.8}$$

Moreover, the following holds for the terminal nodes of Algorithm 1.4

$$\text{dist}_{\mathcal{P}}(\{r_0, \dots, r_{i-1}\}, r_i) = \text{dist}_{\mathcal{P}}(\{r_0, \dots, r_{i-1}\}, \{r_i, \dots, r_{n-1}\}). \tag{1.9}$$

By the method of Rosenkrantz, Stearns, and Lewis II [88], one can construct a bijection

$$\begin{aligned} &\sigma : \{1, \dots, n-1\} \to \{1, \dots, n-1\} \text{ such that} \\ &\{t_{\sigma(i)}, t_{\sigma(i)+1}\} \in \{r_0, \dots, r_{i-1}\} \times \{r_i, \dots, r_{n-1}\} \ \forall i = 1, \dots, n-1. \end{aligned} \tag{1.10}$$

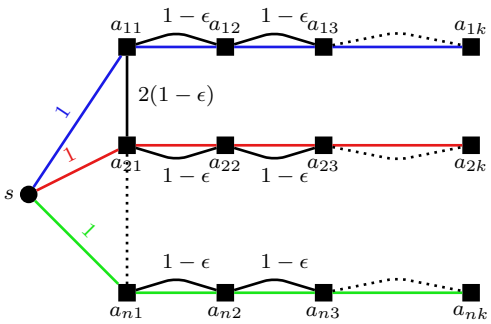


Figure 1.9: Worst case example for the connecting set heuristic given in Algorithm 1.4. All nodes but s are terminal nodes.

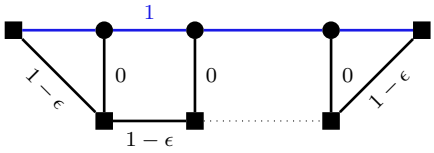


Figure 1.10: Example that Algorithm 1.4 yields no bounded approximation ratio if each path contains at most 2 terminals.

Then

$$c(\mathcal{P}') = \sum_{i=1}^{n-1} c(\mathcal{P}_{r_{i-1},r_i}) \overset{(1.8)}{\leq} \sum_{i=1}^{n-1} \operatorname{dist}_{\mathcal{P}}(\{r_1,\dots,r_{i-1}\},r_i) \overset{(1.9),(1.10)}{\leq} \sum_{i=1}^{n-1} c(\mathcal{P}_{t_{\sigma(i)},t_{\sigma(i)+1}}) \overset{(1.7)}{\leq} (k+1)c(\mathcal{P}_{\text{opt}}).$$

In the above chain of inequalities the summand $c(\mathcal{P}_{t_n,t_1})$ does not appear. If we choose t_1 in such a way that $c(\mathcal{P}_{t_n,t_1}) = \max\{c(\mathcal{P}_{t_i,t_{i+1}}) : i = 1,\dots,n\}$ (taking indices $i+1 \bmod n$), it follows with inequality (1.7) that

$$c(\mathcal{P}') \leq (k+1)\, c(\mathcal{P}_{\text{opt}})(1-\tfrac{1}{n}) = (k+1)\, c(\mathcal{P}_{\text{opt}})(1-\tfrac{1}{|T|}),$$

which proves the claim. □

We show that the approximation guarantee for Algorithm 1.4 is tight. Figure 1.9 illustrates a worst case example. Given is a Steiner connectivity instance with only one non-terminal node s. We have the following paths and costs

$$\begin{array}{lll} o_i = (s, a_{i1}, a_{i2}, \dots, a_{ik}), & c(o_i) = 1, & i = 1,\dots,n, \\ q_i = (a_{i1}, a_{(i+1)1}), & c(q_i) = 2(1-\epsilon), & i = 1,\dots,n-1, \\ p_{ij} = (a_{ij}, a_{i(j+1)}) & c(p_{ij}) = 1-\epsilon, & i = 1,\dots,n,\ j = 1,\dots,k-1. \end{array}$$

The heuristic would choose the paths q_i, $i = 1,\dots,n-1$, and p_{ij}, $i = 1,\dots,n$, $j = 1,\dots,k-1$, with total cost

$$\begin{aligned} c(\text{heur}) &= (n-1)\cdot 2(1-\epsilon) + (k-1)\cdot n\cdot(1-\epsilon) \\ &= n\big((k-1)(1-\epsilon) + 2(1-\epsilon)\big) - 2(1-\epsilon) = n(k+1)(1-\epsilon) - 2(1-\epsilon). \end{aligned}$$

The optimal paths are o_i, $i = 1,\dots,n$, with cost $c(\text{opt}) = n$. We get

$$\frac{c(\text{heur})}{c(\text{opt})} = (k+1)(1-\epsilon) - \frac{2(1-\epsilon)}{n} \xrightarrow[n\to\infty]{\epsilon\to 0} (k+1).$$

Figure 1.10 illustrates an example that the connecting set heuristic given in Algorithm 1.4 yields no bounded approximation ratio if each path contains at most 2 terminal nodes but can be arbitrarily long. We have $n+2$ terminal nodes and n non-terminal nodes. The blue path has cost 1 and contains two terminal nodes and n non-terminal nodes. Each (black) edge is covered by a path with cost as noted next to the edge. The optimal T-connecting set has cost $c(\text{opt}) = 1$, taking the blue path and all 0-cost paths. The T-connecting set computed by the heuristic takes all other paths with cost $c(\text{heur}) = (n+1)(1-\epsilon)$, i. e.,

$$\frac{c(\text{heur})}{c(\text{opt})} = (n+1)(1-\epsilon) \xrightarrow[n\to\infty]{} \infty.$$

The heuristic in Algorithm 1.4 can be modified to a "spanning set heuristic" similar as the minimum spanning tree heuristic for the Steiner tree problem, see, e. g., Prömel and Steger [82] and the references therein. The idea is to construct a complete graph $\bar{G}$ with node set T and edge costs $c_{uv} = \min\{c(\mathcal{P}') : \mathcal{P}' \subseteq \mathcal{P} \text{ is } uv\text{-connecting}\}$. Computing a minimum spanning tree in $\bar{G}$ gives rise to a T-connecting set $\mathcal{P}'$ by uniting the minimal uv-connecting sets for each edge $\{u, v\}$ in the minimum spanning tree.

Corollary 1.19. *The spanning set heuristic is a $(k+1)$-approximation algorithm.*

Proof. This follows with the same arguments as given in the proof of Proposition 1.18. The spanning set heuristic adds a new terminal node t according to a minimum cost st-connecting set with s being any *terminal* node already covered. (In Algorithm 1.4 s could be any node already covered.) □

1.4.3 Primal-Dual Approximation Algorithm

In the following we will construct a primal-dual algorithm to find a T-connecting set. It is analogous to the algorithm of Goemans and Williamson [54] for the Steiner forest problem that is visualized in Figure 1.11. Our application to the Steiner connectivity problem is listed in Algorithm 1.5.

The algorithm constructs a T-connecting set $\mathcal{P}'$. In the beginning $\mathcal{P}' = \emptyset$ and each terminal node is considered to form a connected component that contains only itself. The idea is to extend and merge the connected components along paths until we have only one connected component left. In each iteration *moats* around the connected components are grown until a path *goes tight*. The radii of the moats correspond to the values of the dual variables for the cuts around the connected components, and a *path goes tight* if its associated inequality in the dual program becomes an equality, compare with equation (1.11).

More precisely, let B^i be the set of all connected components in iteration i of Algorithm 1.5; the initial set is $B^0 = \{\{t\} : t \in T\}$, i. e., the set of all terminal nodes. We iterate as long as B^i consists of more than one connected component. Denote by $B^i_p = \{b \in B^i : p \in \mathcal{P}_{\delta(b)}\}$ the set of connected components the path p "cuts" in iteration i. We set $|B^i_p| = 0$ for $B^i_p = \emptyset$. In iteration i we choose, among paths $p \in \mathcal{P}$

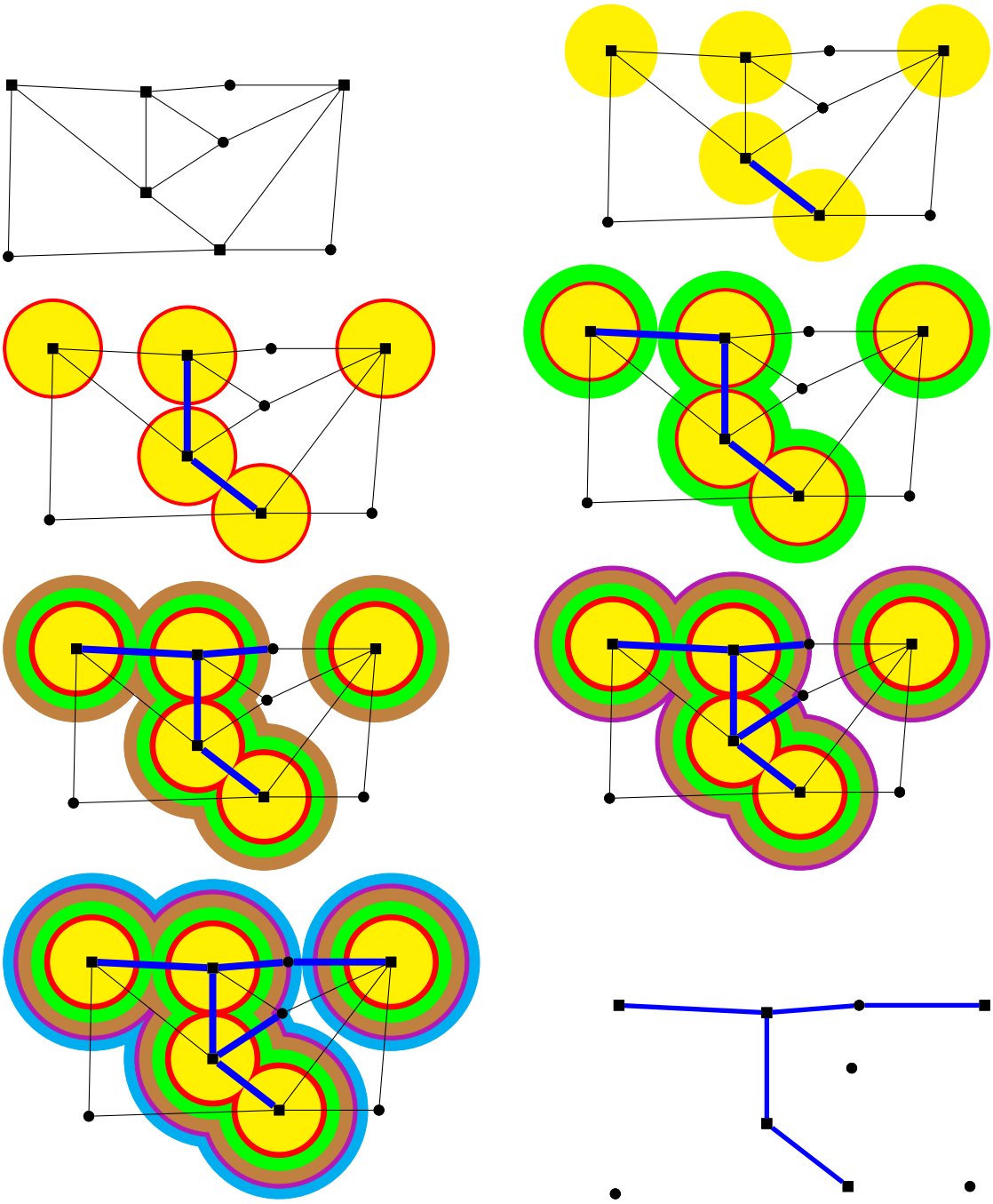


Figure 1.11: The idea of the primal dual algorithm of Goemans and Williamson for the (Euclidean) Steiner tree problem. The upper left figure shows a Steiner tree instance, the square nodes are the terminal nodes, the length of an edge corresponds to the cost of the edge. The colored areas around the connected components $b \in B^i$ (the terminal nodes in the first step) are the moats. If two moats touch each other or if a moat reaches a non-terminal node, the corresponding edge *goes tight* and is added to the Steiner tree, marked blue in the figures. The two connected components or the connected component and the non-terminal node are merged. After the reverse deleting step, the lower right figure shows the final Steiner tree.

Algorithm 1.5: A primal dual heuristic for SCP

Input : A connected graph $G = (V, E)$, a set of paths $\mathcal{P}$ with costs $c \in \mathbb{R}^{\mathcal{P}}_{\geq 0}$, a set of terminal nodes $T \subseteq V$.

Output: A T-connecting set $\mathcal{P}' \subseteq \mathcal{P}$.

1 $B^0 := \{\{t\} \,|\, t \in T\}\}$, $\mathcal{P}' := \emptyset$, $c^0_p := c_p$ for all $p \in \mathcal{P}$, $i := 0$
2 $y_W := 0 \,\forall\, W \subset V$, $W \cap T \neq \emptyset$, $V \setminus W \cap T \neq \emptyset$ `// only set when needed`
3 **while** $|B^i| > 1$ **do**
4 $\quad p = \text{argmin}_{\ q \in \mathcal{P}} \{\frac{c^i_q}{|B^i_q|} : |B^i_q| > 0\}$
5 $\quad a^i := \frac{c^i_p}{|B^i_p|}$
6 $\quad \mathcal{P}' := \mathcal{P}' \cup \{p\}$
7 $\quad$ **for** all $b \in B^i$ **do**
8 $\quad\quad y_b := y_b + a^i$
9 $\quad$ **end**
10 $\quad B^{i+1} := (B^i \setminus B^i_p) \cup \{b_1 \cup \ldots \cup b_k \cup V(p)\}$ with $\{b_1, \ldots, b_k\} := B^i_p$
11 $\quad$ **for** all $q \in \mathcal{P} \setminus \mathcal{P}'$ **do**
12 $\quad\quad c^{i+1}_q := c^i_q - |B^i_q| a^i$
13 $\quad$ **end**
14 $\quad i := i + 1$
15 **end**
16 **for** all $p \in \mathcal{P}'$ **do**
17 $\quad$ **if** $\mathcal{P}' \setminus p$ is T-connecting **then**
18 $\quad\quad \mathcal{P}' := \mathcal{P}' \setminus p$ `// deleting step`
19 $\quad$ **end**
20 **end**

with $|B^i_p| > 0$, a path for which the quotient of reduced cost and number of connected components the path cuts in iteration i is minimal, line 4. Denote by a^i this minimum value, line 5; it gives the maximum amount the dual variables can be increased. The associated path is added to $\mathcal{P}'$ (ties broken arbitrarily), line 6, and the dual variables or moat radii are increased by the value a^i, line 8. If the path contains several connected components, these are merged into one connected component, line 10. All non-terminal nodes of the path are also added to the new connected component. Note that the path contains at least two connected components or at least one connected component and one non-terminal node. Line 12 is an updating step to prepare the computation of the next amount of increase of the dual variables. The final set of chosen paths is T-connecting. In the end of the algorithm, lines 16 to 20, we consecutively remove paths as long as the resulting set is still T-connecting to obtain a minimal T-connecting set $\mathcal{P}'$.

Let $\mathcal{W} = \{W \subset V \,;\, \emptyset \neq W \cap T \neq T\}$. The analysis of Algorithm 1.5 is based on the

consideration of the following dual programs:

$$\begin{array}{lll} \min & \sum_{p\in\mathcal{P}} c_p\, x_p & \\ \text{s.t.} & \sum_{p\in\mathcal{P}_{\delta(W)}} x_p \geq 1 & \forall\, W \in \mathcal{W} \\ & x_p \geq 0 & \forall\, p \in \mathcal{P} \end{array} \qquad \begin{array}{lll} \max & \sum_{W\in\mathcal{W}} y_W & \\ \text{s.t.} & \sum_{W\in\mathcal{W}:p\in\mathcal{P}_{\delta(W)}} y_W \leq c_p & \forall\, p \in \mathcal{P} \\ & y_W \geq 0 & \forall\, W \in \mathcal{W}. \end{array} \tag{1.11}$$

The primal program is the LP relaxation of the undirected cut formulation of the Steiner connectivity problem. It minimizes the cost of a set of paths. This set of paths has to contain at least one path of each T-disconnecting set and is, hence, a T-connecting set. The undirected cut formulation is considered in detail in Chapter 2.

Proposition 1.20. *Setting $x_p = 1$ for all $p \in \mathcal{P}'$, $x_p = 0$ for all $p \in \mathcal{P} \setminus \mathcal{P}'$, and using variables y as defined at the end of Algorithm 1.5 gives solutions for the primal and dual programs (1.11).*

Proof. It is easy to see that $\mathcal{P}'$ is a T-connecting set. Now, consider y_W, $W \in \mathcal{W}$. Clearly, $y_W \geq 0$, $W \in \mathcal{W}$. Let $p \in \mathcal{P}$ and r be the last iteration of the while-loop, i. e., at the end of this last iteration a^r and c_p^{r+1}, $p \in \mathcal{P} \setminus \mathcal{P}'$, are defined. Then we get

$$\sum_{W\in\mathcal{W}:p\in\mathcal{P}_{\delta(W)}} y_W \overset{\text{(i)}}{=} \sum_{i=0}^{r} \sum_{b\in B_p^i} a^i \overset{\text{(ii)}}{=} c_p^0 - c_p^{r+1} \overset{\text{(iii)}}{\leq} c_p. \tag{1.12}$$

(i) Compare with line 8 in Algorithm 1.5.

(ii) We use the following equations, compare with line 12 in the algorithm,

$$c_p^{r+1} = c_p^r - \sum_{b\in B_p^r} a^r = c_p^0 - \sum_{i=0}^{r} \sum_{b\in B_p^i} a^i.$$

(iii) This follows since $0 \leq c_p^{r+1} \leq c_p^0 = c_p$, compare with lines 1, 4, and 5 in Algorithm 1.5.

Hence, y is feasible for the dual program in (1.11). Moreover, we have equality in (1.12) for $p \in \mathcal{P}'$, i. e.,

$$\sum_{W\in\mathcal{W}:p\in\mathcal{P}_{\delta(W)}} y_W = c_p. \tag{1.13}$$

□

Proposition 1.21. *Given a Steiner connectivity instance, let $\mathcal{P}_{\text{opt}}$ be the minimum cost T-connecting set and $\mathcal{P}'$ a T-connecting set computed with Algorithm 1.5. Then*

$$c(\mathcal{P}') \leq (k+1)\, c(\mathcal{P}_{\text{opt}}) \left(1 - \tfrac{1}{|T|}\right),$$

i. e., Algorithm 1.5 is a $(k+1)$-approximation algorithm with k being the minimum of

(a) the maximal number of edges in a path,

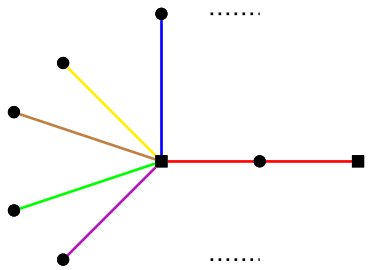


Figure 1.12: Example for the necessity of the deleting step in Algorithm 1.5. We have two terminal nodes, the square ones, and n non-terminal nodes, each connected by a path of cost 1 to one of the terminal nodes. The two terminal nodes are connected via a path (red) of cost 2 which is the optimal solution. The primal-dual Algorithm 1.5 initially adds all paths to the set $\mathcal{P}'$, but the deleting step eliminates the superfluous ones.

(b) the maximal number of terminal nodes in a path.

Proof. Summing up the cost of all paths in $\mathcal{P}'$, we get

$$\sum_{p\in\mathcal{P}'} c_p \overset{(1.13)}{=} \sum_{p\in\mathcal{P}'} \sum_{W\in\mathcal{W}:p\in\mathcal{P}_{\delta(W)}} y_W = \sum_{W\in\mathcal{W}} \sum_{p\in\mathcal{P}_{\delta(W)}\cap\mathcal{P}'} y_W = \sum_{W\in\mathcal{W}} \deg_{\mathcal{P}'}(W) y_W.$$

If we can show the following

$$\sum_{W\in\mathcal{W}} \deg_{\mathcal{P}'}(W) y_W \leq (k+1)\big(1-\tfrac{1}{|T|}\big) \sum_{W\in\mathcal{W}} y_W, \tag{1.14}$$

we are done. Note that this is more general than to require $\deg_{\mathcal{P}'}(W) \leq k+1$. We show inequality (1.14) by induction over the iterations of the algorithm. Initially, $y_W = 0$ for all $W \in \mathcal{W}$, so inequality (1.14) is true. Now, we have to show that the increase on the left hand side is smaller than the increase on the right hand side in every iteration, i. e., for each iteration i we have to show the following

$$\begin{aligned} a^i \sum_{b\in B^i} \deg_{\mathcal{P}'}(b) &\leq (k+1)\big(1-\tfrac{1}{|B^i|}\big) \sum_{b\in B^i} a^i \overset{|B^i|\leq|T|}{\leq} (k+1)\big(1-\tfrac{1}{|T|}\big) \sum_{b\in B^i} a^i \\ \Leftrightarrow \sum_{b\in B^i} \deg_{\mathcal{P}'}(b) &\leq (k+1)(|B^i|-1) = (k+1)\big(1-\tfrac{1}{|B^i|}\big)|B^i|. \end{aligned}$$

Each $b \in B^i$ is connected. Consider the graph $\tilde{G}$ that contains a node for each $b \in B^i$ and all nodes $v \in V$ that are not contained in one of the $b \in B^i$. Then the final set $\mathcal{P}'$ restricted to $\tilde{G}$ is a minimal B^i-connecting set. The rest follows with the Degree Lemma 1.3 since $\mathcal{P}'$ is minimal. □

Case (a) in Proposition 1.21 was also stated by Takeshita, Fujito, and Watanabe [64] in a paper written in Japanese. As far as we could find out, however, they do not give a proof for the Degree-Lemma 1.3 (case (a)); we also do not know of any other reference.

The deleting step of Algorithm 1.5 is important for the approximation guarantee since the degree property only holds for minimal T-connecting sets. Figure 1.12 shows an

example in which the deleting step is necessary: the algorithm initially chooses all paths but only one is necessary to connect the terminal nodes.

Chapter 2

IP Formulations and Polyhedra

In this chapter, we will analyze integer programming formulations and polyhedral aspects of the Steiner connectivity problem. We propose an *extended directed cut formulation* for the problem based on the relation to the directed Steiner tree problem shown in Section 1.2. We show that this formulation is provably strong, including, e. g., a class of facet defining generalized Steiner partition inequalities. It dominates the canonical undirected cut formulation. This generalizes a similar result for the Steiner tree problem. Main parts of this chapter are published in [20, 26].

The structure of this chapter is as follows. In Section 2.1 we propose and compare three integer programming formulations for the SCP: a canonical undirected cut formulation and two extended directed cut formulations that are based on the equivalence of the Steiner connectivity problem and an associated directed Steiner tree problem, compare with Section 1.2. An analysis of the polytope associated with the undirected cut formulation follows in Section 2.2. We state necessary and sufficient conditions for the Steiner partition inequalities to be facet defining. We show that a super class of the Steiner partition inequalities can be separated in polynomial time. This shows that extended formulations provide tight relaxations for the SCP.

2.1 IP Formulations

In this section, we propose three integer programming formulations for the SCP. The first one (SCP_{cut}) is the canonical undirected cut formulation, the second one ($\text{SCP}^r_{\text{arc}^+}$) is a directed cut formulation based on the equivalence between the SCP and its associated DSTP, the third one ($\text{SCP}^r_{\text{con}^+}$) is also a directed cut formulation, but in a smaller space. It will turn out that ($\text{SCP}^r_{\text{arc}^+}$) and ($\text{SCP}^r_{\text{con}^+}$) are equivalent and dominate (SCP_{cut}).

The section uses the following notation. For a vector $x \in \mathbb{R}^n$ and an index set $I \subseteq \{1, \ldots, n\}$, let $x|_I = x_I$ be the restriction of x onto the subspace indexed by I. Let $P_{LP}(F)$ be the polyhedron associated with the LP relaxation of an IP formulation F.

Then $P_{LP}(F)|_I$ is the orthogonal projection of $P_{LP}(F)$ on the subspace of variables indexed by I.

2.1.1 Cut Formulation

The cut formulation is as follows:

$$\begin{aligned}
(\mathrm{SCP}_{\mathrm{cut}}) \quad \min \quad & \sum_{p\in\mathcal{P}} c_p\, x_p & & \\
\text{s.t.} \quad & \sum_{p\in\mathcal{P}_{\delta(W)}} x_p \geq 1 & \forall\, W \subseteq V,\, \emptyset \neq W \cap T \neq T \qquad & (2.1)\\
& x_p \in \{0,1\} & \forall\, p \in \mathcal{P}. &
\end{aligned}$$

Here, x_p is a 0/1-variable that indicates whether path p is chosen ($x_p = 1$) or not ($x_p = 0$). The set $\mathcal{P}_{\delta(W)}$ with $W \subseteq V$, $\emptyset \neq W \cap T \neq T$ is the T-path cut or Steiner path cut as defined in Chapter 1; a Steiner path cut $\mathcal{P}_{\delta(W)}$ with $|\mathcal{P}_{\delta(W)}| = 1$ is a *Steiner path bridge*. For given x, the *capacity of a Steiner path cut* $\mathcal{P}_{\delta(W)}$ is $\sum_{p\in\mathcal{P}_{\delta(W)}} x_p$, and we denote the inequalities (2.1) as *Steiner path cut constraints*; they state that the capacity of each Steiner path cut must be at least one. It is easy to see that ($\mathrm{SCP}_{\mathrm{cut}}$) is a valid formulation for the SCP.

If each path has length 1, i. e., contains only one edge, the sets $\delta(W)$ and $\mathcal{P}_{\delta(W)}$ are equal. In this case the Steiner connectivity problem reduces to a Steiner tree problem, and the Steiner path cut constraints reduce to the so-called *Steiner cut constraints*.

Replacing the Steiner path cut constraints by the inequalities

$$\sum_{e\in\delta(W)} \sum_{p:e\in p} x_p \geq 1 \qquad \forall\, W \subseteq V,\, \emptyset \neq W \cap T \neq T$$

produces the integer program

$$\begin{aligned}
(\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}}) \quad \min \quad & \sum_{p\in\mathcal{P}} c_p\, x_p & & \\
\text{s.t.} \quad & \sum_{e\in\delta(W)} \sum_{p:e\in p} x_p \geq 1 & \forall\, W \subseteq V,\, \emptyset \neq W \cap T \neq T \qquad & (2.2)\\
& x_p \in \{0,1\} & \forall\, p \in \mathcal{P}. &
\end{aligned}$$

This *weak cut formulation* is also a correct IP formulation of the SCP. Note that the left hand side of a *weak Steiner path cut constraint* (2.2) counts how often each path crosses the cut $\delta(W)$. These inequalities can be seen as a direct generalization of the Steiner cut constraints for the STP. However, they are clearly dominated by the Steiner path cut constraints.

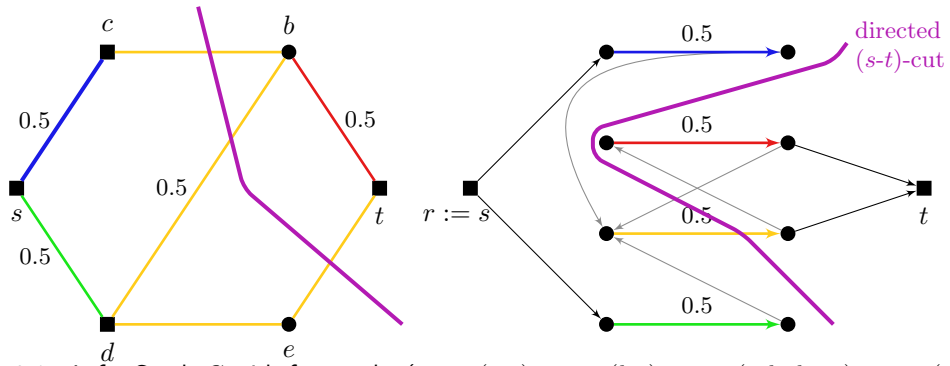


Figure 2.1: *Left:* Graph G with four paths ($p_1 = (s, c)$, $p_2 = (b, t)$, $p_3 = (c, b, d, e, t)$, $p_4 = (s, d)$) with $\hat{x}$-value 0.5 and two terminal nodes s and t. *Right:* Corresponding Steiner connectivity digraph D' for $r := s$. Here, each arc has capacity 0.5. The minimum directed (s, t)-cut has value 0.5 and corresponds to the Steiner path cut $\mathcal{P}' = \{p_3\}$ in G.

Formulation (SCP_{cut}) has $|\mathcal{P}|$ variables and $\mathcal{O}(2^{|V|})$ Steiner path cut constraints, i. e., the number of Steiner path cut constraints can be exponential in the size of the input. However, the associated separation problem, i. e., to decide whether a given point $\hat{x}$ is feasible for the LP relaxation of (SCP_{cut}) or to find a violated Steiner path cut constraint, can be solved in polynomial time. Namely, this problem can be formulated as a family of max flow/min cut problems in the Steiner connectivity digraph $D' = (V', A')$ that was defined in Section 1.2. Consider some nonnegative vector $\hat{x} \in \mathbb{R}^{\mathcal{P}}_{\geq 0}$. We define the following *standard arc capacities* $\kappa = \kappa(\hat{x})$ for D':

$$
\begin{array}{lll}
a = (r, v_p), & \kappa_a := \hat{x}_p, & \forall\, p \in \mathcal{P} \text{ with } r \in p, \\
a = (v_p, w_p), & \kappa_a := \hat{x}_p, & \forall\, p \in \mathcal{P}, \\
a = (w_{\tilde{p}}, v_p), & \kappa_a := \min\{\hat{x}_p, \hat{x}_{\tilde{p}}\}, & \forall\, p, \tilde{p} \in \mathcal{P},\ r \notin p,\ p \neq \tilde{p},\ \ p \text{ and } \tilde{p} \text{ have} \\
 & & \text{a node } v \in V \text{ in common}, \\
a = (w_p, t), & \kappa_a := \hat{x}_p, & \forall\, p \in \mathcal{P},\ \forall\, t \in T \setminus \{r\} \text{ with } t \in p.
\end{array}
$$

Figure 2.1 illustrates this construction. The following holds.

Lemma 2.1. *Let $t \in T \setminus \{r\}$ be a terminal node and $\hat{x} \in \mathbb{R}^{\mathcal{P}}_{\geq 0}$ be a nonnegative vector. If the Steiner connectivity digraph D' has standard capacities $\kappa = \kappa(\hat{x})$, there exists a directed (r, t)-cut with minimum capacity in D' such that all arcs over this cut are of the form (v_p, w_p), $p \in \mathcal{P}$.*

Proof. Let $\delta^-(W)$ be a directed (r, t)-cut with $W \subseteq V \setminus \{r\}$. We show that we can derive an alternative cut set $\tilde{W}$ with smaller or equal capacity where all arcs are of the form (v_p, w_p). Thus, if $\delta^-(W)$ has minimum capacity, then $\delta^-(\tilde{W})$ has minimum capacity as well.

- Assume $(r, v_p) \in \delta^-(W)$, i. e., $v_p \in W$. We set $\tilde{W} = W \setminus \{v_p\} \cup \{w_p\}$ and get $\delta^-(\tilde{W}) = \delta^-(W) \setminus \{(r, v_p)\} \cup \{(v_p, w_p)\}$, because (v_p, w_p) is the only arc with source node v_p and target node w_p, recall statements 1 and 2 of Observation 1.5, and since $r \in p$, $r, v_p)$ is the only arc with target node v_p. Furthermore, $(v_p, w_p) \in \delta^-(\tilde{W})$ and $\kappa_{r v_p} = \kappa_{v_p w_p}$. Hence, $\delta^-(\tilde{W})$ has the same capacity as $\delta^-(W)$.

- If $(w_p, t) \in \delta^-(W)$, we set $\tilde{W} = W \setminus \{v_p\} \cup \{w_p\}$ and argue as above.
- Assume $(w_{\tilde{p}}, v_p) \in \delta^-(W)$, $p \neq \tilde{p}$, and $\hat{x}_p \leq \hat{x}_{\tilde{p}}$. In this case, we set $\tilde{W} = W \setminus \{v_p\} \cup \{w_p\}$ and get $\delta^-(\tilde{W}) \subseteq \delta^-(W) \setminus \{(w_{\tilde{p}}, v_p)\} \cup \{(v_p, w_p)\}$, again because of statements 1 and 2 of Observation 1.5. Furthermore, $(v_p, w_p) \in \delta^-(\tilde{W})$ and $\kappa_{v_p w_p} = \kappa_{w_{\tilde{p}} v_p}$. Hence, $\delta^-(\tilde{W})$ has capacity not larger than $\delta^-(W)$.
- Assume $(w_{\tilde{p}}, v_p) \in \delta^-(W)$, $p \neq \tilde{p}$, and $\hat{x}_{\tilde{p}} \leq \hat{x}_p$. In this case we set $\tilde{W} = W \setminus \{v_{\tilde{p}}\} \cup \{w_{\tilde{p}}\}$ and argue similarly.

In all cases, the set W changes in such a way that nodes w_p enter W and nodes v_p leave W. Hence all steps can be repeated until the cut has the desired form. □

We call a cut of the form stated in Lemma 2.1 a *standard cut*; then Lemma 2.1 can be rephrased as stating that there exists a minimum capacity directed (r, t)-cut in a Steiner connectivity digraph with standard capacities which is a standard cut.

Proposition 2.2. *Let $\kappa \in \mathbb{R}^{A'}_{\geq 0}$ and $\hat{x} \in \mathbb{R}^{\mathcal{P}}_{\geq 0}$ be capacities for D' and G, respectively, such that $\kappa_a = \hat{x}_p$ for all $a = (v_p, w_p) \in A'$, $p \in \mathcal{P}$. Then there is a one-to-one correspondence between minimal directed (r, t)-standard cuts in D' (w. r. t. root node r) and minimal (r, t)-Steiner path cuts in G, and the capacities are equal.*

Proof. "⇒": Consider a directed (r, t)-standard cut $\delta^-(W')$ in D'. We first show that $\delta^-(W')$ gives rise to an (r, t)-disconnecting set

$$\mathcal{P}' = \{p \in \mathcal{P} : (v_p, w_p) \in \delta^-(W')\}$$

in G. Assume there exists a path from r to t in G that is covered only by paths in $\mathcal{P} \setminus \mathcal{P}'$ (i. e., $\mathcal{P}'$ is not a disconnecting set). Let $p_1, \ldots, p_k$ be the paths that are used in this order when traversing the path. Then $(r, v_{p_1}, w_{p_1}, \ldots, v_{p_k}, w_{p_k}, t)$ is a path from r to t in D' that uses only arcs in $A' \setminus \delta^-(W')$. This is a contradiction to the assumption that $\delta^-(W')$ is a directed (r, t)-standard cut in D'.

Now let $\delta^-(W')$ be minimal and suppose $\mathcal{P}'$ is not. Then there exists a smaller (r, t)-disconnecting set $\mathcal{P}'' \subset \mathcal{P}'$. Consider for some path $p \in \mathcal{P}' \setminus \mathcal{P}''$ the arc $(v_p, w_p) \in \delta^-(W')$. As $\delta^-(W')$ is a minimal disconnecting set in D', there exists an (r, t)-path $(r, v_{p_1}, w_{p_1}, \ldots, v_{p_k}, w_{p_k}, t)$ in $A' \setminus \delta^-(W') \cup \{v_p, w_p\}$. But then $p_1, \ldots, p_k$ is a set of paths in $\mathcal{P} \setminus \mathcal{P}' \cup \{p\} \subseteq \mathcal{P} \setminus \mathcal{P}''$ that connect r and t in G, i. e., $\mathcal{P}''$ is not an (r, t)-disconnecting set. This is a contradiction. Therefore $\mathcal{P}'$ is minimally disconnecting and, by Lemma 1.1, $\mathcal{P}'$ is a minimal (r, t)-Steiner path cut.

"⇐": Let $\mathcal{P}'$ be an (r, t)-Steiner path cut. Then $\mathcal{P}'$ is an (r, t)-disconnecting set in G. Define $W' = \{t\} \cup \{w_p : p \in \mathcal{P}'\} \cup W''$, where W'' is the set of nodes from which t can be reached using arcs in the set $A' \setminus \{(v_p, w_p) | p \in \mathcal{P}'\}$. Then we show that $\delta^-(W')$ is a directed (r, t)-standard cut in D', namely, $\delta^-(W') = \{(v_p, w_p) : p \in \mathcal{P}'\}$. It is clear that $\delta^-(W') \supseteq \{(v_p, w_p) : p \in \mathcal{P}'\}$, because the only node that can be reached from v_p is w_p. To show equality, consider the following cases:

- Assume $(r, v_p) \in \delta^-(W')$ for some $p \in \mathcal{P}$. If $p \in \mathcal{P}'$, then $v_p \notin W'$, a contradiction. If $p \notin \mathcal{P}'$ then t can be reached from v_p via arcs in $A' \setminus \{(v_p, w_p) | p \in \mathcal{P}'\}$. Hence, there is an (r,t)-path covered by $p \in \mathcal{P} \setminus \mathcal{P}'$, a contradiction.
- Assume $(w_p, t) \in \delta^-(W')$ for some $p \in \mathcal{P}$. For both cases $p \in \mathcal{P}'$ and $p \notin \mathcal{P}'$ we have $w_p \in W'$, a contradiction.
- Assume $(v_p, w_p) \in \delta^-(W')$ for some $p \in \mathcal{P} \setminus \mathcal{P}'$. Then $w_p \in W'$, i. e., t can be reached from w_p via arcs in $A' \setminus \{(v_p, w_p) | p \in \mathcal{P}'\}$, but $v_p \notin W'$, a contradiction.
- Assume $(w_{\tilde{p}}, v_p) \in \delta^-(W')$ for some $p, \tilde{p} \in \mathcal{P}$. Then $w_{\tilde{p}} \notin W'$ and $v_p \in W'$. This implies that t can be reached from v_p via arcs in $A' \setminus \{(v_p, w_p) | p \in \mathcal{P}'\}$. But then t can also be reached from $w_{\tilde{p}}$ via arcs in $A' \setminus \{(v_p, w_p) | p \in \mathcal{P}'\}$, a contradiction.

Now assume that $\mathcal{P}'$ is a minimal (r,t)-Steiner path cut (i. e., a minimal (r,t)-disconnecting set via Lemma 1.1) and $\delta^-(W')$ is not minimal, i. e., there exists a standard cut $\delta^-(W'') \subset \delta^-(W') = \{(v_p, w_p) : p \in \mathcal{P}'\}$. Then by the forward argument of the proof there exists a disconnecting set $\mathcal{P}'' \subsetneq \mathcal{P}'$, a contradiction.

"$\Leftrightarrow$": It is easy to see that in both cases $\mathcal{P}'$ and $\delta^-(W')$ have the same capacity, and that the constructions in the two directions of the proof pair the same cuts. □

Remark 2.3. *Note that Proposition 2.2 holds for* all *capacities such that $\kappa_a = \hat{x}_p$ for all $a = (v_p, w_p) \in A'$, $p \in \mathcal{P}$, not only for standard capacities.*

Proposition 2.4. *The separation problem for Steiner path cut constraints can be solved in polynomial time.*

Proof. Computing for every two terminals $s, t \in T$ a minimum (s,t)-cut in D' with respect to standard capacities, using s as root node, can be done in polynomial time. If and only if the value of this cut is smaller than 1, we can find a violated Steiner path cut constraint by transforming this cut into a standard cut via Lemma 2.1 and then applying Proposition 2.2. This can also be done in polynomial time. □

2.1.2 Directed Cut Formulation

Our second formulation of the SCP is the well-known *directed cut formulation* for the associated DSTP, compare with Chopra and Rao [35] for the formulation and with Section 1.2 for the definition of the associated DSTP:

$$
\begin{aligned}
(\mathrm{SCP}_{\mathrm{arc}}) \quad \min \quad & \sum_{a \in A'} c'_a \, y_a & & \\
\text{s.t.} \quad & \sum_{a \in \delta^-(W')} y_a \geq 1 & \forall\, W' \subseteq V' \setminus \{r\},\ W' \cap T \neq \emptyset \qquad & (2.3) \\
& y_a \in \{0,1\} & \forall\, a \in A'. &
\end{aligned}
$$

Compared to the undirected cut formulation, the number of variables of $(\mathrm{SCP}_{\mathrm{arc}})$ is quadratic, i. e., $|A'| \in \mathcal{O}(|\mathcal{P}|^2)$, and we have $\mathcal{O}(2^{|V'|}) = \mathcal{O}(2^{2|\mathcal{P}|})$ cut constraints. The

solutions of (SCP_{arc}) are *supersets* of directed Steiner trees for the terminal set T. However, since we minimize a nonnegative objective, there always exists an optimal solution that is a directed Steiner tree. The separation problem for the *directed Steiner cut constraints* (2.3) consists of solving $|T| - 1$ min-cut problems, i. e., for each $t \in T \backslash \{r\}$ one has to find a minimum (r, t)-cut in D'. This can be done in polynomial time.

(SCP_{arc}) can be interpreted as an extended formulation of (SCP_{cut}) by identifying arcs (v_p, w_p) and paths $p \in \mathcal{P}$. We define

$$A'_{\mathcal{P}} = \{(v_p, w_p) \in A' : p \in \mathcal{P}\}$$

and write $y|_{\mathcal{P}} = y|_{A'_{\mathcal{P}}}$ to simplify the notation. Then, Proposition 1.6 states that if y is an integer solution of (SCP_{arc}), its projection on the subspace of *path-arcs* gives rise to a solution $x = y|_{\mathcal{P}}$ of (SCP_{cut}) via $x_p = y_{v_p w_p}$, $p \in \mathcal{P}$, and vice versa. This relation also holds for the LP relaxations of (SCP_{cut}) and (SCP_{arc}).

Lemma 2.5. $P_{LP}(\text{SCP}_{\text{cut}}) = P_{LP}(\text{SCP}_{\text{arc}})|_{\mathcal{P}}$.

Proof. "$\supseteq$": Let $y^* \in P_{LP}(\text{SCP}_{\text{arc}})$, i. e., y^* satisfies all directed (r, t)-Steiner cuts for some root r and every terminal $t \in T \backslash \{r\}$. By Proposition 2.2 and Remark 2.3, the vector $x^* = y^*|_{\mathcal{P}}$ satisfies all (r, t)-Steiner path cuts for every terminal $t \in T \backslash \{r\}$. Since any (s, t)-Steiner path cut is either an (r, s)- or an (r, t)-Steiner path cut, $y^*|_{\mathcal{P}}$ also satisfies the (s, t)-Steiner path cuts for all $s, t \in T \setminus \{r\}$, i. e., $y^*|_{\mathcal{P}} = x^* \in P_{LP}(\text{SCP}_{\text{cut}})$.

"$\subseteq$": Let $x^* \in P_{LP}(\text{SCP}_{\text{cut}})$, in particular, x^* satisfies the (s, t)-Steiner path cuts for all $s, t \in T$ and hence all (r, t)-Steiner path cuts for some fixed root r. We define $y^* \in \mathbb{R}^{A'}$ by setting $y^* = \kappa(x^*)$ according to the standard capacity definition, i. e., in particular, $y^*|_{\mathcal{P}} = x^*$. By Proposition 2.2, the vector y^* satisfies all directed (r, t)-standard cuts, and by Lemma 2.1, all directed (r, t)-cuts, i. e., $y^* \in P_{LP}(\text{SCP}_{\text{arc}})$. □

Corollary 2.6. *The optimal objective values of the LP relaxations of* (SCP_{arc}) *and* (SCP_{cut}) *are equal. In particular, the objective value of the LP relaxation of* (SCP_{arc}) *is independent of the choice of the root node* r.

Proof. This follows from Lemma 2.5, since $c'|_{\mathcal{P}} = c$ and $c'|_{A' \setminus A'_{\mathcal{P}}} = 0$. □

It is known that directed cut formulations for the STP can easily be strengthened by a small number of inequalities that one can write down explicitly. It will turn out that in our case such a strengthening is also possible and dominates a large class of facet defining Steiner partition inequalities for the undirected cut formulation of the SCP, see Section 2.2. The additional inequalities are as follows. Since we assume nonnegative costs, there is always an optimal solution of the associated DSTP that is a directed tree. Each non-terminal node that is contained in such a cost minimal directed Steiner tree has at least one outgoing arc and at most one incoming arc. Therefore, the so-called *flow balance inequalities* can be added to (SCP_{arc}):

$$\sum_{a \in \delta^-(v)} y_a \leq \sum_{a \in \delta^+(v)} y_a \qquad \forall\, v \in V' \backslash T.$$

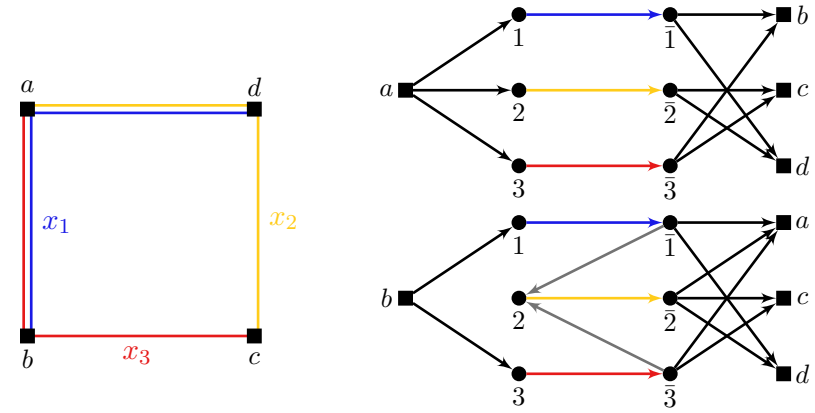


Figure 2.2: An SCP instance showing that choosing different roots leads to different solutions of the LP relaxation of $(\mathrm{SCP}^r_{\mathrm{arc}^+})$. Choosing node a as root allows to set all path values to 0.5 in the LP relaxation of $(\mathrm{SCP}^a_{\mathrm{arc}^+})$. This solution is not possible for the LP relaxation of $(\mathrm{SCP}^b_{\mathrm{arc}^+})$, when b is chosen as root.

In the context of the Steiner tree problem these inequalities were first considered by Duin [45] and later studied by Koch and Martin [66], Polzin [79], and Polzin and Daneshmand [80]. The only non-terminal nodes in D' are the nodes v_p and w_p, $p \in \mathcal{P}$. We will show that the flow balance constraints for nodes v_p, $r \notin p$, and for nodes w_p, $t \notin p$, $\forall\, t \in T$ (note that $r \in T$ is included), can be omitted. Appending these flow balance constraints produces the following *strengthened directed cut formulation* for the SCP:

$$
\begin{aligned}
(\mathrm{SCP}^r_{\mathrm{arc}^+}) \quad \min \quad & \sum_{a \in A'} c'_a\, y_a & & \\
\text{s.t.} \quad & \sum_{a \in \delta^-(W')} y_a \geq 1 & & \forall\, W' \subseteq V' \backslash \{r\},\, W' \cap T \neq \emptyset \\
& y_{v_p w_p} \geq \sum_{a \in \delta^-(v_p)} y_a & & \forall\, v_p \in V'\, (p \in \mathcal{P} : r \notin p) \qquad (2.4) \\
& \sum_{a \in \delta^+(w_p)} y_a \geq y_{v_p w_p} & & \forall\, w_p \in V'\, (p \in \mathcal{P} : t \notin p\, \forall\, t \in T) \qquad (2.5) \\
& y_a \in \{0,1\} & & \forall\, a \in A'. \qquad (2.6)
\end{aligned}
$$

This strengthened directed cut formulation $(\mathrm{SCP}^r_{\mathrm{arc}^+})$ improves an earlier version defined in our paper [20] in two ways. It enlarges the model by including the flow balance constraints for the nodes w_p and simultaneously omits flow balance constraints that have (as we will see) no effect on the objective function. An example in which the additional flow balance constraints for nodes w_p tighten the formulation is discussed in the next section in the context of the contracted directed cut formulation. The lower right of Figure 2.2 shows an example of a typical violation of the flow balance constraints for node v_p: Setting the y-variables associated with the arcs shown in this Steiner connectivity digraph to 0.5 produces a solution that satisfies all directed Steiner cut constraints, but violates the flow balance constraint at node 2.

We will now consider the flow balance constraints that can be omitted in model $(\mathrm{SCP}^r_{\mathrm{arc}^+})$.

Lemma 2.7. *There exists an optimal solution of the LP relaxation of* $(\mathrm{SCP}^r_{\mathrm{arc}^+})$ *that satisfies the flow balance constraints (a) for all nodes* $v_p \in V'$ *with* $r \in p$ *and (b) for all nodes* $w_p \in V'$ *with* $t \in p$ *for a terminal node* $t \in T$.

Proof.

(a) Let $v_p \in V'$, $p \in \mathcal{P}$, with $r \in p$, i. e., $(r, v_p) \in A'$. Assume we are given an optimal LP solution $y^* \in P_{LP}(\mathrm{SCP}^r_{\mathrm{arc}^+})$ with $y^*_{rv_p} > y^*_{v_pw_p}$. Let $\delta^-(W)$ be an (r,t)-cut with $(r, v_p) \in \delta^-(W)$ and $\sum_{a\in\delta^-(W)} y^*_a = 1$. If no such cut exists we can reduce the y^*-value of arc (r, v_p) until $y^*_{rv_p} = y^*_{v_pw_p}$ holds (and we are done) or until it exists. We have $v_p \in W$. Then setting $W' := W \setminus \{v_p\} \cup \{w_p\}$ ($w_p \in W$ is possible) yields an (r,t)-cut $\delta^-(W')$ with $(v_p, w_p) \in \delta^-(W')$. Due to the construction of D', (r, v_p) is the only arc with target node v_p and (v_p, w_p) is the only arc with source node v_p. We get $\delta^-(W) \setminus \{(r, v_p)\} = \delta^-(W') \setminus \{(v_p, w_p)\}$ and therefore

$$\begin{aligned} & 1 = \sum_{a\in\delta^-(W)} y^*_a \le \sum_{a\in\delta^-(W')} y^*_a \\ \Leftrightarrow \quad & \sum_{a\in\delta^-(W)\setminus\{(r,v_p)\}} y^*_a + y^*_{rv_p} \le \sum_{a\in\delta^-(W')\setminus\{(v_p,w_p)\}} y^*_a + y^*_{v_pw_p} \Leftrightarrow y^*_{rv_p} \le y^*_{v_pw_p}, \end{aligned}$$

a contradiction. The construction reduces values $y^*_{rv_p}$ and can be repeated until y^* satisfies all flow balance constraints $\sum_{a\in\delta^-(v_p)} y_a \le y_{v_pw_p}$ for all $p \in \mathcal{P}$.

(b) Let $w_p \in V'$, $p \in \mathcal{P}$, with $t \in p$ for a terminal node $t \in T$. We distinguish two cases: (i) $t \ne r$ and (ii) $t = r$. In case (i) we have $(w_p, t) \in A'$. Then we can increase, if necessary, $y^*_{w_pt}$ until $y^*_{w_pt} \ge y^*_{v_pw_p}$ since $c_{w_pt} = 0$. This construction can be repeated until y^* satisfies all flow balance constraints $\sum_{a\in\delta^+(w_p)} y_a \ge y_{v_pw_p}$ for all $p \in \mathcal{P}$ with $t \in p$ for a $t \in T \setminus \{r\}$.
Consider case (ii). Due to the construction of D' there exists only one path from root node r to node w_p, namely, (r, v_p, w_p). Case (a) ensures that we can assume that $y^*_{rv_p} \le y^*_{v_pw_p}$. Assume further that $y^*_{v_pw_p} > \sum_{a\in\delta^+(w_p)} y^*_a$. The following holds. If $\delta^-(W')$ is an (r,t)-cut with $(v_p, w_p) \in \delta^-(W')$ then $W'' := W' \setminus \{w_p\}$ yields an (r,t)-cut $\delta^-(W'')$ with $\delta^-(W') \setminus \{(v_p, w_p)\} = \delta^-(W'') \setminus \{a \in \delta^+(w_p)\}$. Let $\delta^-(W')$ be an (r,t)-cut with $(v_p, w_p) \in \delta^-(W')$ and $\sum_{a\in\delta^-(W')} y^*_a = 1$. If no such cut exists we can reduce the y^*-value of (v_p, w_p) and if necessary of (r, v_p) as in case (a) until $y^*_{v_pw_p} = \sum_{a\in\delta^+(w_p)} y^*_a$ (and we are done) or until such a cut exists. Suppose there exists an arc $a \in \delta^+(w_p)$ with $a \in \delta^-(W'')$. Otherwise $\delta^-(W') \setminus \{(v_p, w_p)\}$ is also an (r,t)-cut, i. e., $y^*_{v_pw_p}$ must have been 0 or $\sum_{a\in\delta^-(W')\setminus\{(v_p,w_p)\}} y_a < 1$, a contradiction. We get

$$\begin{aligned} & 1 = \sum_{a\in\delta^-(W')} y^*_a \le \sum_{a\in\delta^-(W'')} y^*_a \\ \Leftrightarrow \quad & \sum_{a\in\delta^-(W')\setminus\{(v_p,w_p)\}} y^*_a + y^*_{v_pw_p} \le \sum_{a\in\delta^-(W'')\setminus\{a\in\delta^+(w_p)\}} y^*_a + \sum_{a\in\delta^+(w_p)\cap\delta^-(W'')} y^*_a \\ \Leftrightarrow \quad & y^*_{v_pw_p} \le \sum_{a\in\delta^+(w_p)\cap\delta^-(W'')} y^*_a \le \sum_{a\in\delta^+(w_p)} y^*_a, \end{aligned}$$

a contradiction. The construction possibly reduces values $y^*_{v_p w_p}$ and $y^*_{r v_p}$ and can be repeated until y^* satisfies all flow balance constraints $y_{v_p w_p} \leq \sum_{a \in \delta^+(w_p)} y_a$ for all $p \in \mathcal{P}$ with $r \in p$.

The construction in the proof can be done consecutively for cases (a) and (b) such that all flow balance constraints are satisfied for all v_p, w_p, $p \in \mathcal{P}$. □

Since $(\mathrm{SCP}_{\mathrm{arc}})$ and $(\mathrm{SCP}^r_{\mathrm{arc}^+})$ always have an optimal solution that is a directed Steiner tree, the optimal objective values of $(\mathrm{SCP}^r_{\mathrm{arc}^+})$ and $(\mathrm{SCP}_{\mathrm{arc}})$ are equal but the LP relaxation of the first model might be stronger.

Corollary 2.8. $P_{LP}(\mathrm{SCP}_{\mathrm{cut}}) = P_{LP}(\mathrm{SCP}_{\mathrm{arc}})|_{\mathcal{P}} \supseteq P_{LP}(\mathrm{SCP}^r_{\mathrm{arc}^+})|_{\mathcal{P}}$.

Remark 2.9. *The objective value of the LP relaxation of* $(\mathrm{SCP}^r_{\mathrm{arc}^+})$ *can depend on the choice of the root node, see Figure 2.2.*

2.1.3 Contracted Directed Cut Formulation

A third formulation of the SCP arises from the directed cut formulation by contracting the path-arcs (v_p, w_p), $p \in \mathcal{P}$, i. e., we consider a contracted Steiner connectivity digraph $D'' = (V'', A'') = D'/\{(v_p, w_p) : p \in \mathcal{P}\}$. Let v_p be the node that arises from contracting the arc (v_p, w_p), i. e., $V'' = V' \setminus \{w_p : p \in \mathcal{P}\} = T \cup \{v_p : p \in \mathcal{P}\}$. Analogously, we identify arcs in A'' and A', i. e., $A'' = A' \setminus A'_{\mathcal{P}}$ (here $(w_p, v) \in A'$ corresponds to $(v_p, v) \in A''$). Furthermore, let $c''_a = c_p$ for $a = (u, v_p) \in A''$, $p \in \mathcal{P}$, and 0 otherwise, i. e., the path costs are shifted to the ingoing arcs of a node v_p. D'' can be interpreted as a *terminal and path intersection digraph* more directly than D'. The *strengthened contracted directed cut formulation* reads as follows:

$$
\begin{aligned}
(\mathrm{SCP}^r_{\mathrm{con}^+}) \quad \min \quad & \sum_{a \in A''} c''_a y_a & & \\
\text{s.t.} \quad & \sum_{a \in \delta^-(W'')} y_a \geq 1 & \forall\, W'' \subseteq V'' \setminus \{r\},\ W'' \cap T \neq \emptyset \quad & (2.7) \\
& 1 \geq \sum_{a \in \delta^-(v_p)} y_a & \forall\, v_p \in V''\ (p \in \mathcal{P}) \quad & (2.8) \\
& \sum_{a \in \delta^+(v_p)} y_a \geq \sum_{a \in \delta^-(v_p)} y_a & \forall\, v_p \in V''\ (p \in \mathcal{P} : t \notin p\, \forall t \in T) \quad & (2.9) \\
& y_a \in \{0, 1\} & \forall\, a \in A''. &
\end{aligned}
$$

The constraints (2.9) are the *contracted flow balance constraints*. Constraints (2.8) ensure at most one arc enters each Steiner node. The solutions of $(\mathrm{SCP}^r_{\mathrm{con}^+})$ are also supersets of directed Steiner trees for the terminal set T, and there always exists an optimal solution that is a directed Steiner tree for the terminal set T. We consider a mapping from

$y'' \in \mathbb{R}^{A''}$ to $y' \in \mathbb{R}^{A'}$ to relate formulations ($\text{SCP}^r_{\text{con}^+}$) and ($\text{SCP}^r_{\text{arc}^+}$), namely,

$$y'|_{A''} := y'', \qquad y'_p := \sum_{a \in \delta^-(v_p)} y''_a, \quad p \in \mathcal{P} \tag{2.10}$$

and a projection from $y' \in \mathbb{R}^{A'}$ to $y'' \in \mathbb{R}^{A''}$

$$y'' := y'|_{A''}. \tag{2.11}$$

Lemma 2.10. $P_{LP}(\text{SCP}^r_{\text{con}^+}) = P_{LP}(\text{SCP}^r_{\text{arc}^+})|_{A''}$.

Proof. "$\supseteq$": Let $y' \in P_{LP}(\text{SCP}^r_{\text{arc}^+})$. Then y' satisfies all directed (r,t)-cuts for root r and each terminal $t \in T \backslash \{r\}$ in D'. Let $y'' \in \mathbb{R}^{A''}$ be the projection from y' as defined in equation (2.11). We show that $y' \in P_{LP}(\text{SCP}^r_{\text{con}^+})$.

Consider a directed (r,t)-cut $\delta^-(W'')$ in D''. Let

$$W' := W'' \cup \{w_p : v_p \in W'',\ p \in \mathcal{P}\}.$$

Then $W' \subseteq V' \setminus \{r\}$ and $t \in W' \cap T$, i. e., $\delta^-(W') \subseteq A'$ is an (r,t)-cut in D'. Moreover, identifying A'' and $A' \setminus A'_{\mathcal{P}}$, we have $\delta^-(W') = \delta^-(W'')$. It follows that

$$\sum_{\substack{a \in \delta^-(W') \\ a \in A'}} y'_a \geq 1 \quad \Rightarrow \quad \sum_{\substack{a \in \delta^-(W'') \\ a \in A''}} y''_a \geq 1,$$

i. e., $y'' = y'|_{A''}$ satisfies the directed (r,t)-cut inequality for $\delta^-(W'')$.

Now consider a path $p \in \mathcal{P}$ with $t \notin p$ for all $t \in T$, in particular $r \notin p$. Combining the flow balance constraints in ($\text{SCP}^r_{\text{arc}^+}$) for v_p and w_p, i. e., combining inequalities (2.4) and (2.5), and again identifying A'' and $A' \setminus A'_{\mathcal{P}}$, yields

$$\sum_{\substack{a \in \delta^+(w_p) \\ a \in A'}} y'_a \geq y_{v_p w_p} \geq \sum_{\substack{a \in \delta^-(v_p) \\ a \in A'}} y'_a \quad \Rightarrow \quad \sum_{\substack{a \in \delta^+(v_p) \\ a \in A''}} y''_a \geq \sum_{\substack{a \in \delta^-(v_p) \\ a \in A''}} y''_a,$$

i. e., y'' satisfies the contracted flow balance constraints (2.9).

Again identifying A'' and $A' \setminus A_{\mathcal{P}}$ and using the flow balance constraints for v_p, $r \notin p$, we have

$$1 \geq y_{v_p w_p} \geq \sum_{\substack{a \in \delta^-(v_p) \\ a \in A'}} y'_a \Rightarrow 1 \geq \sum_{\substack{a \in \delta^-(v_p) \\ a \in A''}} y''_a.$$

For $r \in p$, we have $\sum_{a \in \delta^-(v_p)} y''_a = y''_{r v_p} \leq 1$, i. e., y'' satisfies inequalities (2.8). It follows that $y'' = y'|_{A''} \in P_{LP}(\text{SCP}^r_{\text{con}^+})$.

"$\subseteq$": Let $y'' \in P_{LP}(\text{SCP}^r_{\text{con}^+})$. Then y'' satisfies all directed (r,t)-cuts for root r and all $t \in T \setminus \{r\}$ in D''. Define $y' \in \mathbb{R}^{A'}$ from y'' as in equation (2.10). We show that $y' \in P_{LP}(\text{SCP}^r_{\text{arc}^+})$.

Let $\delta^-(W')$ be an (r,t)-cut in D'. We distinguish two cases: $(v_p, w_p) \notin \delta^-(W')$ for all $p \in \mathcal{P}$ or there exists a $p \in \mathcal{P}$ such that $(v_p, w_p) \in \delta^-(W')$.

- Let $(v_p, w_p) \notin \delta^-(W')$ for all $p \in \mathcal{P}$, i. e., we either have $v_p, w_p \in W'$ or $v_p, w_p \notin W'$ for each $p \in \mathcal{P}$. Let
$$W'' := W' \setminus \{w_p : p \in \mathcal{P}\}.$$
Then $W'' \subseteq V'' \setminus \{r\}$ and $t \in W'' \cap T$, i. e., $\delta^-(W'') \subseteq A''$ is an (r,t)-cut in D'' and, analogous to the forward direction, $\delta^-(W'') = \delta^-(W')$, again identifying A'' and $A' \setminus A'_{\mathcal{P}}$. It follows
$$\sum_{\substack{a \in \delta^-(W'') \\ a \in A''}} y''_a \geq 1 \quad \Rightarrow \quad \sum_{\substack{a \in \delta^-(W') \\ a \in A'}} y'_a \geq 1.$$

- Now let $p \in \mathcal{P}$ such that $(v_p, w_p) \in \delta^-(W')$, i. e., $w_p \in W'$ and $v_p \notin W'$. In this case, we set $\hat{W}' = W' \cup \{v_p\}$ and get a new (r,t)-cut with $\delta^-(\hat{W}') \subseteq \delta^-(W') \cup \{(u, v_p) : u \in V'\} \setminus \{(v_p, w_p)\}$. Using $y'_{v_p w_p} = \sum_{a \in \delta^-(v_p)} y'_a$, we get
$$\sum_{a \in \delta^-(W')} y'_a + \sum_{a \in \delta^-(v_p)} y_a - y_{v_p w_p} \geq \sum_{a \in \delta^-(\hat{W}')} y'_a \Rightarrow \sum_{a \in \delta^-(W')} y'_a \geq \sum_{a \in \delta^-(\hat{W}')} y'_a.$$
Note that this operation moved v_p into $\hat{W}'$. Iterating over all $p \in \mathcal{P}$ with $(v_p, w_p) \in \delta^-(W')$, the situation is reduced to the first case.

Now consider a node v_p. Identifying A'' and $A' \setminus A_{\mathcal{P}}$ we get with the mapping from $y'' \in \mathbb{R}^{A''}$ to $y' \in \mathbb{R}^{A'}$ (2.10)
$$\sum_{\substack{a \in \delta^-(v_p) \\ a \in A'}} y'_a = \sum_{\substack{a \in \delta^-(v_p) \\ a \in A''}} y''_a = y'_{v_p w_p},$$
i. e., the flow balance constraints (2.4) for v_p are satisfied. Consider a node w_p with $t \notin p$ for all $t \in T$. Using the contracted flow balance constraints, (2.10), and identifying A'' and $A' \setminus A_{\mathcal{P}}$ we get
$$\sum_{\substack{a \in \delta^+(w_p) \\ a \in A'}} y'_a = \sum_{\substack{a \in \delta^+(v_p) \\ a \in A''}} y''_a \geq \sum_{\substack{a \in \delta^-(v_p) \\ a \in A''}} y''_a = y'_{v_p w_p},$$
i. e., the flow balance constraints (2.5) for w_p are satisfied. This shows the claim. □

Corollary 2.11. *The optimal objective values of the LP relaxations of* $(\mathrm{SCP}^r_{\mathrm{con}^+})$ *and of* $(\mathrm{SCP}^r_{\mathrm{arc}^+})$ *are equal.*

Proof. The proof of Lemma 2.10 shows the following. If y'' is a solution of $(\mathrm{SCP}^r_{\mathrm{con}^+})$ then there exists a solution y' of $(\mathrm{SCP}^r_{\mathrm{arc}^+})$ that satisfies all flow balance constraints with equality and $y'|_{A''} = y''$. Moreover,
$$\sum_{a \in A''} c''_a y''_a = \sum_{p \in \mathcal{P}} \sum_{a \in \delta^-(v_p)} c''_a y''_a = \sum_{p \in \mathcal{P}} c'_{v_p w_p} y'_{v_p w_p} = \sum_{a \in A'} c'_a y'_a.$$

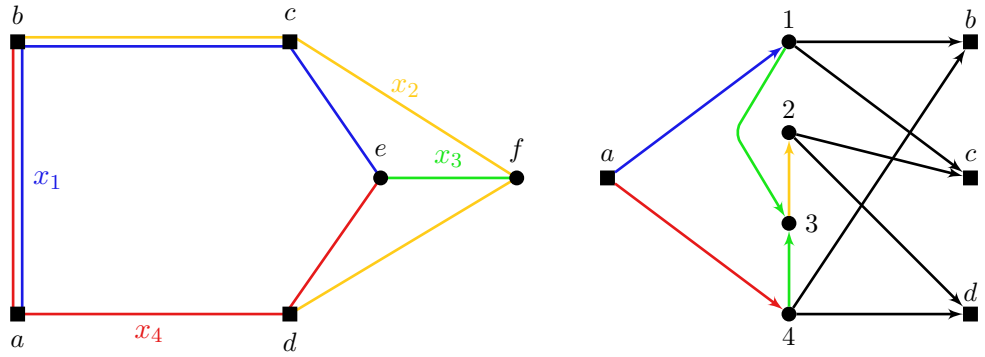


Figure 2.3: An SCP instance showing that the flow balance constraints can improve the LP relaxation of $(\mathrm{SCP}^r_{\mathrm{con}})$, i. e., the contracted directed cut formulation without (contracted) flow balance constraints. The cost of a path corresponds to the number of arcs the path contains, i. e., $c_1 = c_2 = c_4 = 3$, $c_3 = 1$. An optimal solution of the LP relaxation of $(\mathrm{SCP}^r_{\mathrm{con}})$ has cost 5.5. It is visualized on the right for the contracted graph with root $r := a$. All arcs with 0 capacity are omitted in the picture. All other arcs have capacity 0.5, i. e., we get $x_1 = x_2 = x_4 = 0.5$ and $x_3 = y''_{13} + y''_{43} = 1$. All directed (r, t)-cuts are satisfied but the flow balance constraint in node 3 is violated. An optimal solution of the LP relaxation of $(\mathrm{SCP}^r_{\mathrm{con}^+})$ has cost 6 which is equal to the cost of the integer optimal solution.

This shows that the optimal objective value of the LP relaxation of $(\mathrm{SCP}^r_{\mathrm{arc}^+})$ is not larger than the optimal objective value of the LP relaxation of $(\mathrm{SCP}^r_{\mathrm{con}^+})$.

Conversely, if y' is a solution of $(\mathrm{SCP}^r_{\mathrm{arc}^+})$, by Lemma 2.10, $y'' = y'|_{A''}$ is a feasible solution of $(\mathrm{SCP}^r_{\mathrm{con}^+})$ that satisfies

$$\sum_{a \in A'} c'_a \, y'_a = \sum_{p \in \mathcal{P}} c'_{v_p w_p} \, y'_{v_p w_p} \geq \sum_{p \in \mathcal{P}} \sum_{a \in \delta^-(v_p)} c''_a \, y''_a = \sum_{a \in A''} c''_a \, y''_a.$$

This shows the reverse inequality. □

Remark 2.12. *The arguments of this section can be used to show that one can find a directed cut formulation for the Steiner connectivity problem that dominates the canonical undirected cut formulation immediately (without introducing flow balance constraints) similar as for the Steiner tree problem. However, in contrast to the Steiner tree problem, this directed cut formulation is obtained by means of an extended formulation. Let* $(\mathrm{SCP}^r_{\mathrm{con}})$ *be the integer program that is obtained by dropping the contracted flow balance constraints* (2.9) *from formulation* $(\mathrm{SCP}^r_{\mathrm{con}^+})$. *Then each* $y' \in \mathbb{R}^{A'}$ *obtained by equation* (2.10) *from a solution* $y'' \in P_{LP}(\mathrm{SCP}^r_{\mathrm{con}})$ *satisfies inequalities* (2.4), *i. e., the flow balance constraints for node* v_p, *by definition, compare with the proof of Lemma 2.10. It also satisfies all directed* (r, t)*-cuts in* D' *by the same arguments as given in the proof of Lemma 2.10. Hence, we have* $P_{LP}(\mathrm{SCP}^r_{\mathrm{con}}) \subseteq P_{LP}(\mathrm{SCP}^r_{\mathrm{arc}})|_{A''}$ *and the LP relaxation of the first model might be stronger.*

Figure 2.3 gives an example that the contracted flow balance constraints indeed improve the LP relaxation of $(\mathrm{SCP}^r_{\mathrm{con}})$.

We have seen that $(\mathrm{SCP}^r_{\mathrm{arc}^+})$ is a common extended formulation of $(\mathrm{SCP}_{\mathrm{cut}})$ as well as of $(\mathrm{SCP}^r_{\mathrm{con}^+})$. The number of variables of $(\mathrm{SCP}^r_{\mathrm{con}^+})$ is a little smaller than that of

($\mathrm{SCP}^r_{\mathrm{arc}+}$) but still quadratic in $|\mathcal{P}|$, and it is easier to relate ($\mathrm{SCP}^r_{\mathrm{arc}+}$) to ($\mathrm{SCP}_{\mathrm{cut}}$). For this reason, the succeeding sections will investigate the latter relation.

2.2 Polyhedral Analysis

In this section, we investigate the polytope that is associated with the cut formulation of the Steiner connectivity problem. We analyze a class of facet defining Steiner partition inequalities, and discuss the corresponding separation problem. Let

$$P_{\mathrm{SCP}} := \operatorname{conv}\left\{x \in \{0,1\}^{\mathcal{P}} : x \text{ satisfies all Steiner path cut constraints}\right\}$$

be the *Steiner connectivity polytope*. We assume that the Steiner connectivity polytope is nonempty, i. e., the graph G is connected, and each edge is covered by at least one path of $\mathcal{P}$.

In the two-terminal case, a complete description can be given.

Proposition 2.13. *The polytope associated with the LP relaxation of* ($\mathrm{SCP}_{\mathrm{cut}}$)*, i. e.,*

$$\operatorname{conv}\left\{x \in [0,1]^{\mathcal{P}} : x \textit{ satisfies all Steiner path cut constraints}\right\}$$

is integral for $|T| = 2$.

Proof. This follows from Lemma 2.5 and the fact that the polytope associated with the LP relaxation of ($\mathrm{SCP}_{\mathrm{arc}}$) is integral for two terminal nodes (see, e. g., Cornuéjols [38]). □

In general, ($\mathrm{SCP}_{\mathrm{cut}}$) is a special set covering problem. Therefore, the results of Balas and Ng [5] imply the following two lemmas:

Lemma 2.14. *P_{SCP} is full dimensional if and only if there exists no Steiner path bridge.*

Lemma 2.15. *The polytope associated with a Steiner connectivity problem without Steiner path bridges has the following properties:*

1. *The inequality $x_p \geq 0$ defines a facet of P_{SCP} if and only if $|\mathcal{P}_{\delta(W)}| \geq 3$ for all W with $p \in \mathcal{P}_{\delta(W)}$ and $\emptyset \neq W \cap T \neq T$.*
2. *All inequalities $x_p \leq 1$ define facets of P_{SCP}.*
3. *All facet defining inequalities $\alpha^T x \geq \alpha_0$ for P_{SCP} have $\alpha \geq 0$ if $\alpha_0 > 0$.*
4. *A Steiner path cut inequality for $\emptyset \neq W \cap T \neq T$ is facet defining if and only if the following two properties are satisfied:*
   (a) *There exists no W', $\emptyset \neq W' \cap T \neq T$, such that $\mathcal{P}_{\delta(W')} \subsetneq \mathcal{P}_{\delta(W)}$, i. e., $\mathcal{P}_{\delta(W)}$ is not dominated.*
   (b) *For every $p \in \mathcal{P} \setminus \mathcal{P}_{\delta(W)}$ exists a $q \in \mathcal{P}_{\delta(W)}$ such that $q \in \mathcal{P}_{\delta(W')}$ for all $W' \subset V$, $\emptyset \neq W' \cap T \neq T$, with $\mathcal{P}_{\delta(W')} \setminus \mathcal{P}_{\delta(W)} = \{p\}$.*

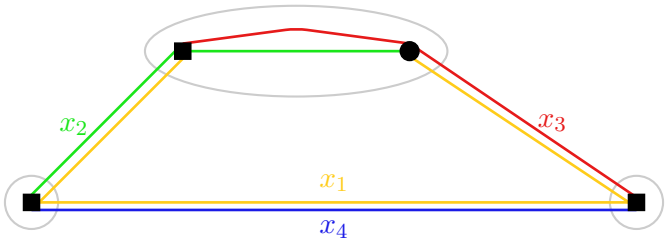


Figure 2.4: The Steiner partition inequality $2x_1 + x_2 + x_3 + x_4 \geq 2$ is facet defining (node sets of the Steiner partition encircled).

5. *The only nontrivial facet defining inequalities for P_{SCP} with integer coefficients and righthand side equal to 1 are Steiner path cut constraints.*

In the following, we assume P_{SCP} to be full dimensional.

2.2.1 Steiner Partition Inequalities

Lemma 2.15 characterizes completely which inequalities of the IP formulation (SCP_{cut}) define facets of the Steiner connectivity polytope. We investigate in this section inequalities arising from node partitions as one important example of an additional class of facets.

Let $P = (V_1, \dots, V_k)$ be a *Steiner partition* of the node set V, i. e., P partitions V and $V_i \cap T \neq \emptyset$ for $i = 1, \dots, k$ and $k \geq 2$. Let $G_P = (V_P, E_P)$ be the graph that arises from contracting each node set $V_i \subseteq V$ to a single node $V_i \in V_P$ (let us denote by V_i a node set in a partition of G as well as a node in the shrunk graph G_P). Note that G_P can have parallel edges but no loops; loops are contracted. Consider a path $p \in \mathcal{P}$: p gives rise to a contracted (not necessarily elementary) path in G_P, which we also denote by p. We say that p *contains* V_i, in formulas $V_i \in p$, if p contains a node of V_i (even if a path $p \in \mathcal{P}$ contains only a single node of G_P). Furthermore, let $\mathcal{P}_P$ denote the set of paths $p \in \mathcal{P}$ that contain at least two distinct shrunk nodes in G_P, in formulas

$$\mathcal{P}_P = \{p \in \mathcal{P} : \exists\, V_i, V_j \in V_P,\ V_i \neq V_j,\ V_i \in p, V_j \in p\},$$

and $\overline{\mathcal{P}} := \mathcal{P} \backslash \mathcal{P}_P$ its complement. Finally, $G[V_i]$ is the graph induced by the nodes V_i, i. e., $G[V_i] = (V_i, E \setminus \{e = \{u, v\} \in E : u \notin V_i \text{ or } v \notin V_i\})$.

Lemma 2.16. *The* Steiner partition inequality

$$\sum_{p \in \mathcal{P}_P} a_p\, x_p \geq k - 1, \tag{2.12}$$

$$a_p := |\{i \in \{1, \dots, k\} : V_i \in V_P,\ V_i \in p\}| - 1$$

is valid for the Steiner connectivity polytope P_{SCP}.

The coefficient a_p, $p \in \mathcal{P}$, counts the number of shrunk nodes that p contains minus one, i. e., a_p is the maximum number of edges that p can contribute to a spanning tree in G_P. The number a_p can be smaller than the number of times that p crosses the multi-cut induced by the Steiner partition.

Note that the inequality can also be stated as $\sum_{p \in \mathcal{P}} a_p x_p \geq k - 1$, because $a_p = 0$ for $p \notin \mathcal{P}_P$. If $k = 2$, the partition inequality is a Steiner path cut constraint. An example of a (facet defining) Steiner partition inequality is illustrated in Figure 2.4.

Proof of Lemma 2.16. We have to show that each 0/1-solution x^* of the Steiner connectivity problem satisfies

$$\sum_{p \in \mathcal{P}_P} a_p x_p^* \geq k - 1.$$

Consider the solution x^* on the shrunk graph G_P. Since each node set V_i, $i = 1, \ldots, k$, contains a terminal node, the shrunk graph G_P has to be connected by the solution x^*, i. e., the (paths of the) support of x^* must contain a spanning tree in G_P. This means that the support of x^* contains at least $k - 1$ edges in G_P. □

The following two propositions give sufficient and necessary conditions for a Steiner partition inequality to be facet defining for the SCP. The sufficient conditions are analogous to those for the Steiner tree polytope, see Grötschel and Monma [57]. Recall $\overline{\mathcal{P}} = \mathcal{P} \setminus \mathcal{P}_P$.

Proposition 2.17. *A Steiner partition inequality is facet defining if the following properties are satisfied.*

1. *$G[V_i]$ is connected by $\overline{\mathcal{P}}$, $i = 1, \ldots, k$.*
2. *$G[V_i]$ contains no Steiner path bridge in $\overline{\mathcal{P}}$, i. e., there is no Steiner path cut $\mathcal{P}_{\delta(W)} \subseteq \overline{\mathcal{P}}$ with $|\mathcal{P}_{\delta(W)}| = 1$ for $W \subseteq V_i$, $\emptyset \neq W \cap T \neq T \cap V_i$, $i = 1, \ldots, k$.*
3. *Each path contains at most two nodes in G_P, i. e., $a_p \in \{0, 1\}$ for all $p \in \mathcal{P}$.*
4. *G_P is 2-node-path-connected, i. e., G_P is path-connected after deleting any node with all incident paths. (compare with Definition 3.10).*

Proof. Let $P = (V_1, \ldots, V_k)$ be a Steiner partition in G and consider the corresponding partition inequality $a^T x = \sum_{p \in \mathcal{P}_P} a_p x_p \geq k - 1$. Assume that properties 1 to 4 are satisfied. Let $b^T x = \beta$ be an equation such that

$$F_a = \{x \in P_{\text{SCP}} : a^T x = k - 1\} \subseteq F_b = \{x \in P_{\text{SCP}} : b^T x = \beta\}$$

and such that F_b is a facet of P_{SCP}.

We first show that $b_p = 0$ for all $p \in \overline{\mathcal{P}}$. Since $p \in \overline{\mathcal{P}}$, p is completely contained in $G[V_j]$ for some $j \in \{1, \ldots, k\}$. Let $\mathcal{P}' \subseteq \mathcal{P}_P$ be a minimal set of paths connecting G_P, i. e., for every two nodes in G_P there exists a path that is completely covered by paths in $\mathcal{P}'$ and if we remove any path of $\mathcal{P}'$ then there are at least two nodes in G_P that are not connected. Since all paths contain at most two different nodes of G_P (property 3), we have $|\mathcal{P}'| = k - 1$. Set $M = \mathcal{P}' \cup \overline{\mathcal{P}}$ and $M' = M \setminus \{p\}$. Since each $G[V_i]$, $i = 1, \ldots, k$,

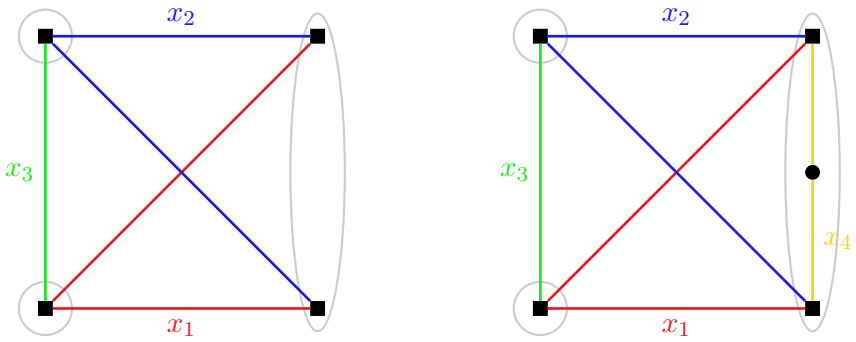


Figure 2.5: Examples of facet defining Steiner partitions that do not satisfy properties 1 (left) and 2 (right) of Proposition 2.17. In both examples the Steiner partition consists of three node sets which are marked gray. The square (terminal) nodes have to be connected.

is connected by paths of $\overline{\mathcal{P}}$ (property 1) and p is not a Steiner path bridge for $G[V_j]$ (property 2), $\chi^M, \chi^{M'} \in P_{\text{SCP}}$ and $a^T\chi^M = a^T\chi^{M'} = k-1$, where χ^M is the incidence vector of M. Thus, $b^T\chi^M = b^T\chi^{M'}$ which implies $b_p = 0$.

Let $p, q \in \mathcal{P}_P$, $p \neq q$. Consider the graph $\hat{G}_P = (V_P, \mathcal{P}_P)$ in which p is an edge between V_i and V_j if it contains V_i and V_j (recall that $p \in \mathcal{P}_P$ contains exactly two nodes, see property 3). Since G_P is 2-node-path-connected, $\hat{G}_P$ is 2-node-connected and there exists a cycle C in $\hat{G}_P$ containing p and q. Let $\mathcal{P}'$ be a tree in $\hat{G}_P$ containing $C\backslash\{p\}$. Then $\mathcal{P}'' = \mathcal{P}' \setminus \{q\} \cup \{p\}$ is also a tree in $\hat{G}_P$. Set $M = \mathcal{P}' \cup \overline{\mathcal{P}}$ and $M' = \mathcal{P}'' \cup \overline{\mathcal{P}}$. Then $\chi^M, \chi^{M'} \in F_a$ and $0 = b^T\chi^M - b^T\chi^{M'} = b_q - b_p$. This implies that $b \in \{0, \lambda\}^{\mathcal{P}}$, $\lambda \geq 0$, using part 3 of Lemma 2.15. Hence, b^Tx is a multiple of a^Tx. This proves that $a^Tx \geq k-1$ defines a facet of P_{SCP}. □

Different from the Steiner tree case (cf. [57]), properties 1 to 3 are not necessary in the Steiner connectivity case, see Figure 2.4 (property 3), Figure 2.5 (left: property 1, right: property 2) for examples. Property 4 is necessary, see Proposition 2.18 below.

We now derive necessary conditions. Let $\Phi_{V_i}(\mathcal{P})$ be the *V_i-contraction* of $\mathcal{P}$, i. e., contract every path $p \in \mathcal{P}$ iteratively in the following way until no reduction is possible anymore:

- If p contains the edges $\{u, v\}$ and $\{v, w\}$, and $v \notin V_i$ then contract $\{u, v\}$ and $\{v, w\}$ to $\{u, w\}$.
- If $p = (\{u_1, u_2\}, \{u_2, u_3\}, \ldots, \{u_{r-1}, u_r\})$, $r \geq 2$, with $u_1 \notin V_i$ then contract p to $p = (\{u_2, u_3\}, \ldots, \{u_{r-1}, u_r\})$.
- If $p = (\{u_1, u_2\}, \{u_2, u_3\}, \ldots, \{u_{r-1}, u_r\})$, $r \geq 2$, with $u_r \notin V_i$ then contract p to $p = (\{u_1, u_2\}, \{u_2, u_3\}, \ldots, \{u_{r-2}, u_{r-1}\})$.

Proposition 2.18. *If the Steiner partition inequality (2.12) is facet defining for a Steiner partition P with at least three partition sets, then the following properties have to be satisfied:*

1. *The shrunk graph G_P is 2-node-path-connected.*

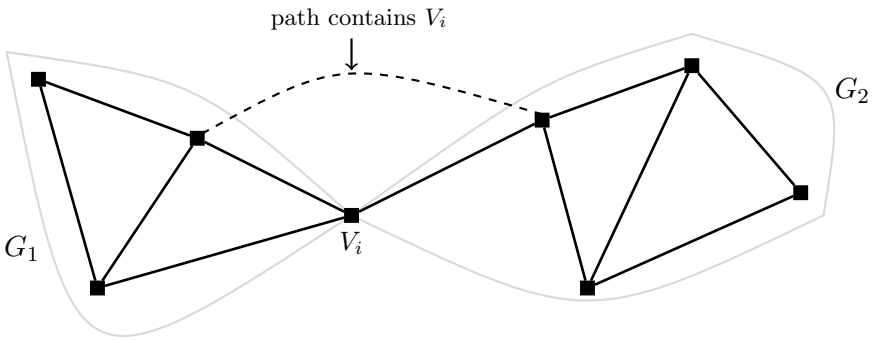


Figure 2.6: The graph G_P in part 1 of the proof of Proposition 2.18 is not 2-node-path-connected and V_i is an articulation node, i. e., each path that connects G_1 and G_2 (dashed in the picture) has to contain V_i.

2. *Either $G[V_i]$ is connected or for every two subsets V_i' and V_i'' of V_i such that $V_i' \dot\cup V_i'' = V_i$ and V_i' is disconnected from V_i'', there exists a path $p \in \mathcal{P}_P$ which contains at least one node of V_i' and one node of V_i'' for all $i = 1, 2, \ldots, k$.*
3. *For each $G[V_i]$ the set of paths $\Phi_{V_i}(\mathcal{P})$ does not contain a Steiner path bridge with respect to $G[V_i]$, i. e., if we remove any $\tilde{p} \in \Phi_{V_i}(\mathcal{P})$ then every two terminal nodes in $G[V_i]$ are still connected by paths of $\Phi_{V_i}(\mathcal{P}) \setminus \{\tilde{p}\}$.*
4. *If two terminal nodes s and t in some $G[V_i]$ are connected by a path $p' \in \mathcal{P}_P$, then these terminals must be also connected by $\overline{\mathcal{P}}$ or we can subdivide V_i into V_i' and V_i'', $V_i = V_i' \dot\cup V_i''$, such that $s \in V_i'$, $t \in V_i''$, and V_i' and V_i'' are not connected by $\overline{\mathcal{P}}$. In the second case for each $V_j \in p'$, $V_j \neq V_i$, there exists a path $p'' \in \mathcal{P}_P$ with $V_j \notin p''$, and $V_i' \in p''$, $V_i'' \in p''$.*

Proof. In the following let $P = (V_1, \ldots, V_k)$, $k \geq 3$, be a Steiner partition with corresponding partition inequality $\sum_{p \in \mathcal{P}_P} a_p x_p \geq k - 1$.

1. Assume G_P is not 2-node-path-connected. In this case there exists a node V_i in G_P which is an articulation node in the following sense: If V_i and all paths incident to V_i are removed from G_P, then the resulting graph is not connected (by the remaining paths). Suppose w.l.o.g. that V_i separates $V_1, \ldots, V_{i-1}$ from $V_{i+1}, \ldots, V_k$. Let $G_1 = G_P[V_1, \ldots, V_i]$ and $G_2 = G_P[V_i, \ldots, V_k]$, see Figure 2.6. Let k_1 be the number of nodes of G_1 and k_2 be the number of nodes of G_2. Recall that the number of nodes of G_P is k. Note that V_i is a node of G_1 and G_2. Therefore we have $k = k_1 + k_2 - 1$.
   We construct a smaller Steiner partition $P' = \{V_1 \cup \ldots \cup V_{i-1} \cup V_i, \ldots, V_k\}$ which contains all nodes of $G_2 \backslash \{V_i\}$ and all nodes of G_1 as a single node. Let the resulting Steiner partition inequality be $\sum_{p \in \mathcal{P}_{P'}} a_p' x_p \geq k_2 - 1$.
   Similarly, we construct a Steiner partition $P'' = \{V_1, \ldots, V_i \cup V_{i+1} \cup \ldots \cup V_k\}$ which contains all nodes of $G_1 \backslash \{V_i\}$ and all nodes of G_2 as a single node. We get the partition inequality $\sum_{p \in \mathcal{P}_{P''}} a_p'' x_p \geq k_1 - 1$.
   The sum of these two partition inequalities is equal to the partition inequality for P. Indeed, $k_1 - 1 + k_2 - 1 = k_1 + k_2 - 2 = k - 1$, and $a_p' + a_p'' = a_p$, see Figure 2.6. Hence, inequality (2.12) does not define a facet.

2. Assume w.l.o.g. $G[V_1]$ is not connected and there exists no path connecting different components of $G[V_1]$. Let $V_1' \subset V_1$ be the node set of one connected component of $G[V_1]$ such that $(V_1 \backslash V_1') \cap T \neq \emptyset$. Since G is connected (and every edge is covered by at least one path) there is a node set V_j, $j \in \{2, \ldots, k\}$, say V_2, such that V_1' and V_2 are connected by a path. We construct a new Steiner partition $P' = (V_1 \backslash V_1', V_1' \cup V_2, V_3, \ldots, V_k)$ and get the partition inequality $\sum_{p \in \mathcal{P}_{P'}} a'_p x_p \geq k - 1$. Let $\hat{\mathcal{P}} = \{p \in \mathcal{P}_P : V_1' \in p,\, V_2 \in p\}$, i. e., $\hat{\mathcal{P}}$ contains all paths that connect V_1' and V_2. One can easily verify that

$$a'_p = \begin{cases} a_p - 1 & \text{if } p \in \hat{\mathcal{P}} \\ a_p & \text{otherwise (since } V_1' \text{ is not connected to } (V_1 \setminus V_1')). \end{cases}$$

   Since $|\hat{\mathcal{P}}| \geq 1$, the partition inequality for P is the sum of the partition inequality for P' and the inequalities $x_p \geq 0$ for all $p \in \hat{\mathcal{P}}$. Therefore, the partition inequality for P is not facet defining.
3. Assume there is a Steiner path bridge $\tilde{p} \in \Phi_{V_i}(\mathcal{P})$ with respect to $G[V_i]$. Let V_i' and $V_i'' := V_i \setminus V_i'$ be two components of $G[V_i]$ that contain terminal nodes which are only connected by $\tilde{p} \in \Phi_{V_i}(\mathcal{P})$. Then $P' = (V_1, \ldots, V_i', V_i'', \ldots, V_k)$ is a Steiner partition. Let the corresponding partition inequality be $\sum_{p \in \mathcal{P}_{P'}} a'_p x_p \geq k$. We claim that this partition inequality plus the upper bound inequality $-x_{\tilde{p}} \geq -1$ of $\tilde{p}$ is equal to the partition inequality for P.
   The partition P' only differs from P in splitting the node set V_i. Because $\tilde{p}$ is the only path that connects V_i' and V_i'', we have $\mathcal{P}_{P'} = \mathcal{P}_P \cup \{\tilde{p}\}$. Furthermore, there is no path in $\mathcal{P}_P$ (except $\tilde{p}$, if $\tilde{p} \in \mathcal{P}_P$) that contains V_i' and V_i''. Therefore the coefficients of all these paths stay the same: $a'_p = a_p$ for all $p \in \mathcal{P}_{P'} \backslash \{\tilde{p}\}$. For $\tilde{p} \in \mathcal{P}_P$ we get $a'_{\tilde{p}} = a_{\tilde{p}} + 1$.
4. Assume w.l.o.g. that there are two terminal nodes s and t in $G[V_1]$ that are connected by a path $p' \in \mathcal{P}_P$ and not connected by paths in $\overline{\mathcal{P}}$. Let V_1' be the nodes reachable from s via paths in $\overline{\mathcal{P}}$ and $V_1'' := V_1 \setminus V_1'$. This shows that the first or the second case of the first part of the statement must hold.
   Furthermore, assume w.l.o.g. that $V_2 \in p'$ and there is no path $p'' \in \mathcal{P}_P$ such that $V_1' \in p''$, $V_1'' \in p''$, and $V_2 \notin p''$. Consider the Steiner partitions $P' := (V_1', V_1'', V_2, \ldots, V_k)$ and $P'' := (V_1 \cup V_2, V_3, \ldots, V_k)$ with corresponding partition inequalities

$$\sum_{p \in \mathcal{P}_{P'}} a'_p x_p \geq k \qquad \text{and} \qquad \sum_{p \in \mathcal{P}_{P''}} a''_p x_p \geq k - 2,$$

   respectively. We show that 2 times the partition inequality for P is dominated by the sum of the partition inequalities for P' and P''. For the right hand side, we obtain:

$$k + k - 2 = 2 \cdot k - 2 = 2 \cdot (k - 1).$$

   For the left hand sides and $p \in \mathcal{P}$, we observe that

$$a'_p = \begin{cases} a_p + 1 & \text{if } V_1' \in p,\ V_1'' \in p \\ a_p & \text{otherwise} \end{cases} \qquad a''_p = \begin{cases} a_p - 1 & \text{if } V_1 \in p,\ V_2 \in p \\ a_p & \text{otherwise.} \end{cases}$$

We claim that $2 \cdot a_p \geq a'_p + a''_p$. Indeed, the only case in which this is not trivially satisfied is when $V'_1 \in p$ and $V''_1 \in p$ (and thus $V_1 \in p$), but $V_2 \notin p$. But this case contradicts our assumptions. □

2.2.2 Separating Steiner Partition Inequalities

Grötschel, Monma, and Stoer [58] showed that separating the Steiner partition inequalities for the Steiner tree problem is $\mathcal{NP}$-hard. This implies that the separation of the Steiner partition inequalities for the Steiner connectivity problem is also $\mathcal{NP}$-hard. However, we show in the following that the Steiner partition inequalities for the SCP are satisfied by all points in $P_{LP}(\mathrm{SCP}^r_{\mathrm{arc}^+})|_{\mathcal{P}}$. This implies that the separation problem for a superclass of Steiner partition inequalities can be solved in polynomial time.

Theorem 2.19. *$P_{LP}(\mathrm{SCP}^r_{\mathrm{arc}^+})|_{\mathcal{P}}$ satisfies all Steiner partition inequalities.*

Proof. Let $y^* \in P_{LP}(\mathrm{SCP}^r_{\mathrm{arc}^+})$. We show that the projection $x^*_p = y^*_{v_p w_p}$ satisfies all Steiner partition inequalities.

Consider an arbitrary Steiner partition $P = (V_1, \ldots, V_k)$ in G and the corresponding partition inequality $\sum_{p \in \mathcal{P}_P} a_p x_p \geq k-1$. W. l. o. g. we assume that $r \in V_k$. Consider the following chain of inequalities

$$\sum_{p \in \mathcal{P}_P} a_p x^*_p \overset{(1)}{\geq} \sum_{\substack{p \in \mathcal{P}_P \\ r \in p}} a_p y^*_{v_p w_p} + \sum_{\substack{p \in \mathcal{P}_P \\ r \notin p}} a_p \sum_{a \in \delta^-(v_p)} y^*_a \overset{(2)}{\geq} \sum_{i=1}^{k-1} \sum_{a \in \delta^-(W_i)} y^*_a \overset{(3)}{\geq} k-1,$$

where $W_i := \{t \in T \backslash \{r\} : t \in V_i\} \cup \{w_p : V_i \in p\} \cup \{v_p : V_i \in p,\, r \notin p\}$, for $i = 1, \ldots, k-1$.

Inequality (1): Identifying $x^*_p = y^*_{v_p w_p}$ and scaling the flow balance constraints $x^*_p = y^*_{v_p w_p} \geq \sum_{a \in \delta^-(v_p)} y^*_a$ by a_p for the paths that do not contain the root node and summing up gives (1).

Inequality (3): Each node set W_i $(i = 1, \ldots, k-1)$ contains at least one terminal node, but not the root node r. Hence, the arc set $\delta^-(W_i)$ is a directed Steiner cut between root r and W_i. Therefore, $\sum_{a \in \delta^-(W_i)} y^*_a \geq 1$ must hold. Summing over all these cuts gives (3).

Inequality (2): The Steiner connectivity digraph D' contains arcs of the form (i) (r, v_p), (ii) (v_p, w_p), $r \in p$, (iii) (v_p, w_p), $r \notin p$, (iv) $(w_{\tilde{p}}, v_p)$, and (v) (w_p, t). We show that all arcs in the cuts $\delta^-(W_i)$, $i = 1, \ldots, k-1$, are of the form (ii) and (iv). Indeed, arcs of the other forms cannot appear in the cuts $\delta^-(W_i)$, $i = 1, \ldots, k-1$:

(i) Arcs of the form (r, v_p) only exist if $r \in p$. But then, $v_p \notin W_i$ due to the definition of W_i, i. e., $(r, v_p) \notin \delta^-(W_i)$.

(iii) The nodes v_p and w_p are either both members or both not members of W_i. In any case, $(v_p, w_p) \notin \delta^-(W_i)$.

(v) Arcs of the form (w_p, t) only exist if $t \in p$. If $t \in W_i$, then $t \in V_i$, and therefore $V_i \in p$. Hence $w_p \in W_i$, i. e., $(w_p, t) \notin \delta^-(W_i)$.

Denote by $\mathcal{V}_p := \{V_i : V_i \in p, i = 1, \dots, k\}$ the set of shrunk nodes contained in p; then $|\mathcal{V}_p| - 1 = a_p$. The proof proceeds by establishing a relation between a_p and the number of times an arc entering v_p appears in the cuts $\delta^-(W_i)$, $i = 1, \dots, k-1$.

Consider an arc $(v_p, w_p) \in A'$. Then the following chain of equations holds:

$$a_p = |\mathcal{V}_p| - 1 = |\mathcal{V}_p \setminus \{V_k\}| = |\{W_i \,|\, (v_p, w_p) \in \delta^-(W_i),\, i = 1, \dots, k-1\}|. \tag{2.13}$$

Here, $(v_p, w_p) \in \delta^-(W_i)$ implies $r \in p$, i. e., $V_k \in p\,(r \in V_k)$ and this yields $|\mathcal{V}_p| - 1 = |\mathcal{V}_p \setminus \{V_k\}|$. Moreover, $(v_p, w_p) \in \delta^-(W_i)$ implies $V_i \in p$. Taking the union for $i = 1, \dots, k-1$ yields $|\mathcal{V}_p \setminus \{V_k\}| = |\{W_i : (v_p, w_p) \in \delta^-(W_i),\, i = 1, \dots, k-1\}|$. Multiplying equation (2.13) with $y^*_{v_p w_p}$ gives

$$\begin{aligned} a_p\, y^*_{v_p w_p} &= |\{W_i \,|\, (v_p, w_p) \in \delta^-(W_i),\, i = 1, \dots, k-1\}| \cdot y^*_{v_p w_p} \\ &= \sum_{i=1}^{k-1} \sum_{(v_p, w_p) \in \delta^-(W_i)} y^*_{v_p w_p}. \end{aligned} \tag{2.14}$$

Consider an arc $(w_{\tilde{p}}, v_p) \in A'$. Then the following chain of equations and inequalities holds

$$a_p = |\mathcal{V}_p| - 1 \geq |\mathcal{V}_p \setminus \mathcal{V}_{\tilde{p}}| \geq |\{W_i \,|\, (w_{\tilde{p}}, v_p) \in \delta^-(W_i),\, i = 1, \dots, k-1\}|. \tag{2.15}$$

Here, $(w_{\tilde{p}}, v_p) \in A'$ implies $\mathcal{V}_p \cap \mathcal{V}_{\tilde{p}} \neq \emptyset$ and this yields $|\mathcal{V}_p| - 1 \geq |\mathcal{V}_p \setminus \mathcal{V}_{\tilde{p}}|$. Moreover, $(w_{\tilde{p}}, v_p) \in \delta^-(W_i)$ implies $V_i \in p$ and $V_i \notin \tilde{p}$. Taking the union for $i = 1, \dots, k-1$ yields $|\mathcal{V}_p \setminus \mathcal{V}_{\tilde{p}}| \geq |\{W_i : (w_{\tilde{p}}, v_p) \in \delta^-(W_i),\, i = 1, \dots, k-1\}|$. Multiplying inequality (2.15) by $y^*_{w_{\tilde{p}} v_p}$ gives

$$\begin{aligned} a_p\, y^*_{w_{\tilde{p}} v_p} &\geq |\{W_i : (w_{\tilde{p}}, v_p) \in \delta^-(W_i),\, i = 1, \dots, k-1\}| \cdot y^*_{w_{\tilde{p}} v_p} \\ &= \sum_{i=1}^{k-1} \sum_{(w_{\tilde{p}}, v_p) \in \delta^-(W_i)} y^*_{w_{\tilde{p}} v_p}. \end{aligned} \tag{2.16}$$

Summing (2.14) and (2.16) over all arcs (v_p, w_p) and $(w_{\tilde{p}}, v_p)$ gives inequality (2):

$$\begin{aligned} &\sum_{\substack{p \in \mathcal{P}_P \\ r \in p}} a_p y^*_{v_p w_p} + \sum_{\substack{p \in \mathcal{P}_P \\ r \notin p}} a_p \sum_{a \in \delta^-(v_p)} y^*_a = \sum_{\substack{p \in \mathcal{P} \\ r \in p}} a_p y^*_{v_p w_p} + \sum_{\substack{p \in \mathcal{P} \\ r \notin p}} a_p \sum_{a \in \delta^-(v_p)} y^*_a \\ &= \sum_{\substack{(v_p, w_p) \in A' \\ r \in p}} a_p\, y^*_{v_p w_p} + \sum_{\substack{(w_{\tilde{p}}, v_p) \in A' \\ r \notin p}} a_p\, y^*_{w_{\tilde{p}} v_p} \overset{(2.14) \text{ and } (2.16)}{\geq} \sum_{i=1}^{k-1} \sum_{a \in \delta^-(W_i)} y^*_a. \end{aligned}$$

This shows the claim. □

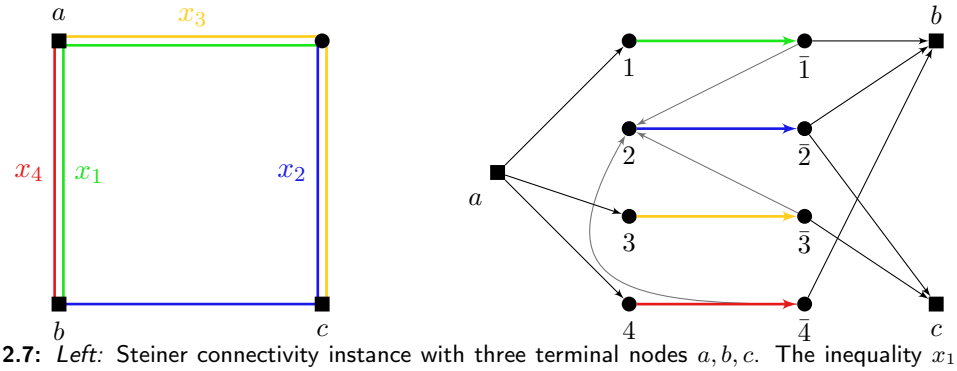


Figure 2.7: *Left:* Steiner connectivity instance with three terminal nodes a, b, c. The inequality $x_1 + x_2 + x_3 + x_4 \geq 2$ is valid but not a Steiner partition inequality. *Right:* Corresponding Steiner connectivity digraph for $r = a$.

Remark 2.20. *Note that the proof of Theorem 2.19 uses only the flow balance constraints for node v_p, $p \in \mathcal{P}$, with $r \notin p$, i. e., the flow balance constraints for node w_p, for all $p \in \mathcal{P}$, are not necessary to derive the Steiner partition inequalities.*

Proposition 2.21. *The separation problem for $P_{LP}(\text{SCP}^r_{\text{arc+}})|_{\mathcal{P}}$ can be solved in polynomial time.*

Proof. Let $P = \{y \in \mathbb{R}^n : Ay \geq b,\ y \geq 0\}$ be a polyhedron, $I \subseteq \{1, \ldots, n\}$, and $x^* \in \mathbb{R}^I$ be a vector. If the optimization problem for P is solvable in polynomial time then the separation problem "$x^* \in P|_I$?" for the projection is solvable in polynomial time. This follows from the equivalence of optimization and separation and its consequences, see Grötschel, Lovász, and Schrijver [56] (intersect P with the affine space $y|_I = x^*$). In our case, the LP relaxation of $(\text{SCP}^r_{\text{arc+}})$ can be solved in polynomial time. This implies the claim. □

A direct method to solve the separation problem for $P_{LP}(\text{SCP}^r_{\text{arc+}})|_{\mathcal{P}}$ will be discussed in Chapter 4 (Subsection 4.1.2).

Corollary 2.22. *If $x^* \in P_{LP}(\text{SCP}_{\text{cut}})$ does not satisfy all Steiner partition inequalities, one can construct a cutting plane that separates x^* from the Steiner connectivity polytope in polynomial time.*

2.2.3 k-Terminal Sets Inequalities

The projected directed cut formulation $(\text{SCP}^r_{arc+})|_{\mathcal{P}}$ implicitly contains other constraints that do not correspond to partition inequalities. Consider the following example.

Example 2.23. *Figure 2.7 shows a facet defining inequality which is not a Steiner partition inequality. Consider the inequality $x_1 + x_2 + x_3 + x_4 \geq 2$. Because the right hand side is 2, a Steiner partition would consist of three node sets, each of which must include at least one terminal node. However, in every possible partition at least one path contains all three partition nodes.*

We show that this inequality is valid for $P_{LP}(\mathrm{SCP}^r_{\mathrm{arc}^+})|_{\mathcal{P}}$. *Choose* a *as root of the associated directed Steiner tree problem, cf. Figure 2.7. (Choosing* b *or* c *as root node would also produce the inequality.) Consider the node sets* $W_b = \{b, 1, \bar{1}, 2, \bar{2}, 4, \bar{4}\}$ *and* $W_c = \{c, 2, \bar{2}, 3, \bar{3}\}$. *The corresponding* (a, b)- *and* (a, c)-*cuts, respectively, are*

$$\begin{aligned} \delta^-(W_b) &= \{(a,1), (a,4), (\bar{3},2)\} \quad \text{and} \\ \delta^-(W_c) &= \{(a,3), (\bar{1},2), (\bar{4},2)\}. \end{aligned}$$

We then get

$$\begin{aligned} x_1 + x_2 + x_3 + x_4 = \; & y_{1\bar{1}} + y_{2\bar{2}} + y_{3\bar{3}} + y_{4\bar{4}} \geq (y_{a1}) + (y_{\bar{1}2} + y_{\bar{3}2} + y_{\bar{4}2}) + (y_{a3}) + (y_{a4}) \\ = \; & (y_{a1} + y_{a4} + y_{\bar{3}2}) + (y_{a3} + y_{\bar{1}2} + y_{\bar{4}2}) \geq 2, \end{aligned}$$

where the flow balance constraints yield the first inequality and the two cut inequalities for $\delta^-(W_b)$ *and* $\delta^-(W_c)$ *yield the second inequality.*

The above example motivates a class of *k-terminal sets inequalities*. The idea is that the partition inequalities can be extended by considering a node set that does not necessarily have to contain a terminal node. More precisely, let $V_1, V_2, V_3, \ldots, V_k \subset V$ be k pairwise disjoint node sets that each contain at least one terminal node. The k node sets need not to partition the whole node set V, i. e., define $\bar{V}$ as $\bar{V} = V \setminus (\cup_{i=1}^k V_k)$. Consider the Steiner path cut constraints

$$\sum_{p \in \mathcal{P}_{\delta(V_i)}} x_p \geq 1$$

for $i = 1, \ldots, k$ which are valid for the Steiner connectivity polytope. If we sum up the Steiner path cut constraints for all node sets V_i, $i = 1, \ldots, k$, we get

$$\sum_{p \in \mathcal{P}_{\delta(V_1)}} x_p + \ldots + \sum_{p \in \mathcal{P}_{\delta(V_k)}} x_p \geq k. \tag{2.17}$$

Dividing this inequality by two and rounding up the coefficient on the left hand side and the right hand side, we get

$$\sum_{p \in \mathcal{P}} \left\lceil \frac{a_p}{2} \right\rceil x_p \geq \left\lceil \frac{k}{2} \right\rceil, \tag{2.18}$$

where $a_p = |\{V \in \{V_1, \ldots, V_k\} : p \in \mathcal{P}_{\delta(V)}\}|$. We will call inequality (2.18) *k-terminal sets inequality*. It is a $\{0, \frac{1}{2}\}$-Chvátal-Gomory cut [33] of Steiner path cut constraints since it arises from a sum of Steiner path cut constraints, each multiplied by 0 or $\frac{1}{2}$. The left of Figure 2.8 gives an example of a k-terminal sets inequality for $k = 5$.

Remark 2.24. The k-terminal sets inequality has the following properties which are easy to see:

- The 3-terminal sets inequality with $\bar{V} = \emptyset$ is the Steiner partition inequality (2.12) with three node sets.

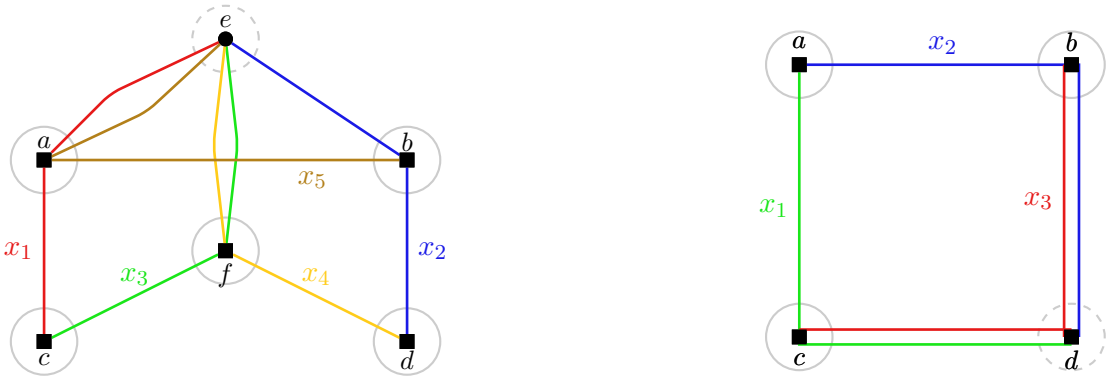


Figure 2.8: *Left:* A Steiner connectivity instance with 5 terminal nodes a, b, c, d, f and five paths. Choosing each terminal node as a node set, the 5-terminal sets inequality is $x_1+x_2+x_3+x_4+x_5 \geq 3$; it is facet defining. *Right:* Choosing the three terminal sets as $\{a\}, \{b\}, \{c\}$ yield the 3-terminal sets inequality $x_1 + x_2 + x_3 \geq 2$. It dominates the Steiner partition inequality for $P = (\{a\}, \{b\}, \{c\}, \{d\})$ which is $2x_1 + 2x_2 + 2x_3 \geq 3$.

- If $\bar{V} \cap T \neq \emptyset$, the k-terminal sets inequality can dominate the Steiner partition inequality for $P = (V_1, \ldots, V_k, \bar{V})$, see the right of Figure 2.8 for an example with $k = 3$.
- The k terminal sets inequality is not facet defining if k is even.

Proposition 2.25. *The k-terminal sets inequality is not facet defining if $\bar{V} \neq \emptyset$ and all paths have length 1, i. e., for the Steiner tree problem.*

Proof. If $\bar{V} \cap T \neq \emptyset$, the k-terminal sets inequality is dominated by the Steiner partition inequality for the partition $(V_1, \ldots, V_k, \bar{V})$: Both inequalities have the same left hand side. Note that an edge/path has coefficient 1 in the k-terminal sets inequality if it has exactly one endpoint in a node set V_i, $i = 1, \ldots, k$. The right hand side of the Steiner partition inequality is k and, hence, bigger than the right hand side of the k-terminal sets inequality which is $\lceil \frac{k}{2} \rceil$.

If $\bar{V} \cap T = \emptyset$, we show that the k-terminal sets inequality is dominated by the Steiner partition inequality for, e. g., the partition $(V_1 \cup \bar{V}, \ldots, V_k)$. It is easy to see that each edge with coefficient 1 in the so-defined Steiner partition inequality has also coefficient 1 in the k-terminal sets inequality. Note that we have only $\{0, 1\}$-coefficients in both inequalities. The right hand side of the Steiner partition inequality is $k - 1$ which is always greater than or equal to $\lceil \frac{k}{2} \rceil$ for $k > 1$. □

Observation 2.26. *The Steiner partition inequality with three node sets is a zero-half Chvátal-Gomory cut of three Steiner path cut constraints.*

We conclude with a small example of a Steiner connectivity instance. This instance has four terminal nodes and six paths, see the upper-left of Figure 2.9. The nontrivial facets of the corresponding Steiner connectivity polytope are Steiner path cut constraints (i) to (iv), Steiner partition inequalities (v) and (vi), and a 3-terminal sets inequality (vii)

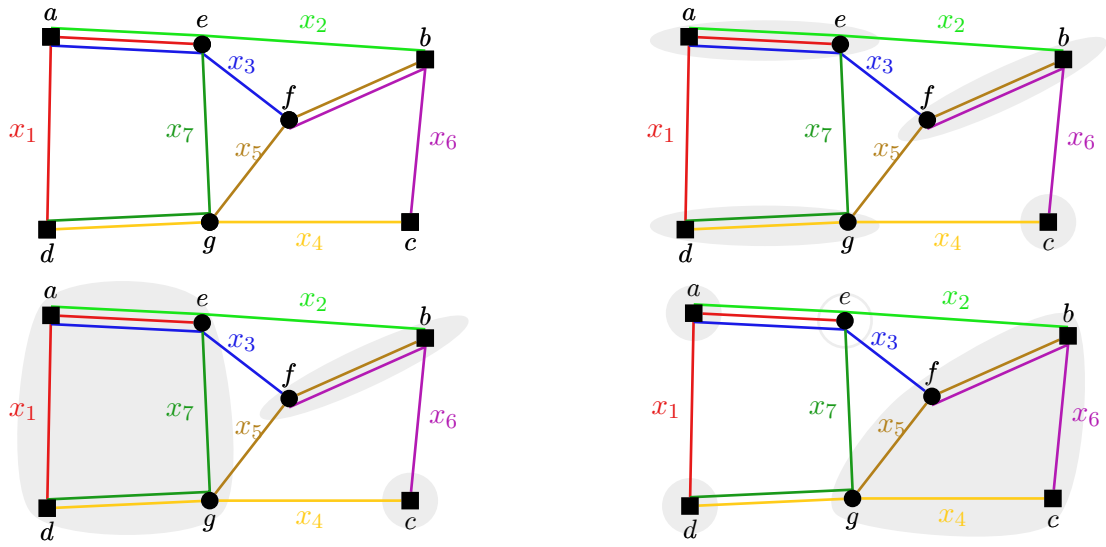


Figure 2.9: *Upper-Left:* Steiner connectivity instance. *Upper-Right:* Steiner partition with four sets. The Steiner partition inequality is $x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \geq 3$. *Lower-Left:* Steiner partition inequality $x_2 + x_3 + x_4 + x_5 + x_6 \geq 2$. *Lower-Right:* Three terminal sets inequality $x_1 + x_2 + x_3 + x_4 + x_7 \geq 2$.

which is not a Steiner partition inequality:

(i)	$x_1 + x_2 + x_3 \geq 1$	$W = \{a\}$
(ii)	$x_2 + x_5 + x_6 \geq 1$	$W = \{b\}$
(iii)	$x_4 + x_6 \geq 1$	$W = \{c\}$
(iv)	$x_1 + x_4 + x_7 \geq 1$	$W = \{d\}$
(v)	$x_1 + \ldots + x_7 \geq 3$	$P = (\{a, e\}, \{b, f\}, \{c\}, \{d, g\})$
(vi)	$x_2 + x_3 + x_4 + x_5 + x_6 \geq 2$	$P = (\{a, d, g, e\}, \{b, f\}, \{c\})$
(vii)	$x_1 + x_2 + x_3 + x_4 + x_7 \geq 2$	$P = (\{a\}, \{b, c, f, g\}, \{d\})$.

The Steiner partition inequalities and the 3-terminal sets inequality are visualized in Figure 2.9.

Chapter 3

Connecting Sets and Disconnecting Sets

An st-connecting set $\mathcal{P}' \subseteq \mathcal{P}$ induces a (not necessarily unique) st-path in the graph $(V(\mathcal{P}'), E(\mathcal{P}'))$. If all paths have length one, i.e., correspond to exactly one edge, then every inclusion wise minimal st-connecting set is an st-path. The same statement holds for st-disconnecting sets and st-cuts. Hence, connecting sets and disconnecting sets generalize paths and cuts. This does not only hold for a single connection, but also for a higher level of connectivity, leading to a Menger type result, and even a max-flow-min-cut theorem. There are, however, also substantial differences to the graph case, most notably, the lack of directed versions of these duality results and the corresponding polynomial time algorithms. In a nutshell, path-connectivity generalizes edge-connectivity but not arc-connectivity.

In this chapter, we show that connecting sets and disconnecting sets form a blocking pair similar as paths and cuts. This involves the validity of two dual theorems, a generalization of Menger's theorem and its companion theorem. While the former one is a well known result of hypergraph theory, we show for the first time that the companion theorem also holds. In fact, we prove that the inequality system of the LP relaxation of the cut formulation for the Steiner connectivity problem with two terminal nodes is TDI. Some parts of this chapter are published in [18].

The structure of this chapter is as follows. In Section 3.1, we briefly introduce the idea of blocking theory on the example of paths and cuts and then apply it to connecting and disconnecting sets, claiming Menger's theorem w. r. t. paths and a Menger companion theorem w. r. t. paths. In Section 3.2, we show that the first theorem follows from hypergraph theory as well as from Hoffman's general max-flow-min-cut theorem. In Section 3.3, we propose an algorithm to find a minimum cost st-connecting set that operates in the original graph. We show its correctness by proving a TDI result. This proof also implies the validity of the Menger companion theorem. In Section 3.4, we show that the incidence matrix of all inclusion wise minimal st-disconnecting sets, indeed, extends the

class of ideal matrices defined by cuts in a graph. We then point out that the results cannot carry over to directed paths in Section 3.5. We end this chapter by considering higher orders of connectivity w. r. t. paths. In particular, we generalize the concepts of k-edge-connected and k-node-connected graphs to our setting in Section 3.6. Again, we will see that not all generalized concepts yield the same complexity results as in the graph case.

3.1 Blocking Pairs

Fulkerson [52] motivates a geometric theory of blocking and anti-blocking pairs of polyhedra by the existence of combinatorial theorems dealing with maximum packing or minimum covering problems that occur in dual pairs. One of the examples he gives is the relation between st-paths and st-cuts. The dual theorems are:

Theorem 3.1. *The minimum number of edges of an st-cut is equal to the maximum number of pairwise edge-disjoint st-paths.*

Theorem 3.2. *The minimum number of edges of an st-path is equal to the maximum number of pairwise edge-disjoint st-cuts.*

That these theorems are a "dual pair" means that one can be obtained from the other by interchanging the roles of paths and cuts. We want to show that there are similar results for connecting and disconnecting sets.

We very briefly recall the basic concepts of blocking theory which we will use in this chapter. For a more detailed description including also the anti-blocking case see Fulkerson [52]. We use similar notations as Borndörfer [8].

Consider a $(0,1)$-matrix A and the associated fractional covering problem (FCP)

$$\begin{array}{lll} \text{(FCP)} & \min & c^T x \\ & \text{s.t.} & Ax \geq 1 \\ & & x \geq 0. \end{array}$$

We are interested in situations in which this LP has an integer optimum; such problems often have a combinatorial meaning. Denote the polytope associated with (FCP) by $Q(A)$. Let $\operatorname{bl} A$ be the matrix that has the incidence vectors of the vertices of $Q(A)$ as its rows. Then

$$Q(A) = \{x \in \mathbb{R}^n_{\geq 0} : Ax \geq 1\} = \operatorname{conv}(\operatorname{bl} A)^T + \mathbb{R}^n_+$$

and $\operatorname{bl} A$ is the *blocker* of A; we also call A and $\operatorname{bl} A$ a *blocking pair* (of matrices). A $(0,1)$ matrix is *ideal* if the polyhedron $\{x : Ax \geq 1, x \geq 0\}$ is integral. The following holds, see also Cornuéjols [38].

Theorem 3.3. *Let A and B be a blocking pair of matrices. A is ideal if and only if B is ideal.*

Coming back to the above defined dual theorems for paths and cuts, Theorem 3.1 is known as the edge-version of Menger's theorem for graphs [73], while Theorem 3.2 is the *companion theorem*. This latter theorem was proven by Robacker [86] but did not find as much attention in the literature as Menger's theorem. A directed version of the companion theorem can be found in Grötschel, Lovász, and Schrijver [56, Theorem 8.3.4]. In the setting of Theorem 3.1 the matrix A in (FCP) corresponds to the incidence matrix of all st-paths while in the setting of Theorem 3.2 A is the incidence matrix of all inclusion wise minimal st-cuts. The two theorems imply that the fractional covering problems for the incidence matrix of all st-paths and for the incidence matrix of all inclusion wise minimal st-cuts are integral. Theorem 3.3 implies that the integrality of one fractional covering problems, i. e., for the incidence matrix of all inclusion wise minimal st-cuts or for the incidence matrix of all inclusion wise minimal st-paths, also follows from the integrality of the other fractional covering problem. Note that Theorem 3.3 does not imply one of the dual theorems from the other.

Considering the incidence matrix of all inclusion wise minimal st-disconnecting sets, (FCP) becomes

$$\begin{aligned} \text{(MCS)} \quad \min \quad & \sum_{p \in \mathcal{P}} c_p \, x_p & & \\ \text{s.t.} \quad & \sum_{p \in \mathcal{P}_{\delta(W)}} x_p \geq 1 & & \forall\, s \in W \subseteq V \setminus \{t\} \\ & x_p \geq 0 & & \forall\, p \in \mathcal{P}. \end{aligned}$$

This is essentially the LP relaxation of the undirected cut formulation for the Steiner connectivity problem (Chapter 2) with $T = \{s, t\}$, except that the upper bounds $x_p \leq 1$, for all $p \in \mathcal{P}$, are missing. This is, however, not a problem since each optimal solution of (MCS) satisfies $x_p \leq 1$ for all $p \in \mathcal{P}$ if $c > 0$, and there always exists an optimal solution with $x_p \leq 1$ for all $p \in \mathcal{P}$ if $c \geq 0$. Hence, (MCS) and the LP relaxation for the undirected cut formulation of the SCP have the same minimal solutions, which is exactly what we are interested in. This leads to the question whether the program (MCS) to find a minimum st-connecting set is integral and whether a similar result as Theorem 3.2 holds for st-disconnecting sets and st-connecting sets.

Theorem 3.4 (Menger companion theorem w. r. t. paths). *The minimum cardinality of an st-connecting set is equal to the maximum number of path-disjoint st-disconnecting sets.*

In the same way, we get the question whether Theorem 3.1 generalizes to st-connecting sets and st-disconnecting sets.

Theorem 3.5 (Menger's theorem w. r. t. paths). *The minimum cardinality of an st-disconnecting set is equal to the maximum number of path-disjoint st-connecting sets.*

The second question is much easier to answer than the first one. In fact, Theorem 3.5 is equivalent to Menger's theorem for hypergraphs which is well known [50, 65]. It

is also implied by Hoffman's general max-flow-min-cut theorem. We will explain this in more detail in Section 3.2. Albeit Theorem 3.5 is widely known and Theorem 3.4 is dual in the sense that the roles of st-connecting sets and st-disconnecting sets are interchanged, we are not aware of any result in this direction. We show the validity of Theorem 3.4 in Section 3.3. In fact, we show a stronger result, namely, that the inequality system of program (MCS) is totally dual integral. The corresponding proof also implies the correctness of an algorithm to find the minimum cost st-connecting set without transforming the problem. Using the equivalence of connecting/disconnecting sets and hyperpaths/hypercuts, it follows that a Menger companion theorem also holds for hypergraphs. This extends the blocking properties of paths and cuts to hyperpaths and hypercuts or to connecting sets and disconnecting sets.

3.2 The Max-Flow-Min-Cut Theorem

In Section 1.1.2, we have seen that st-connecting sets correspond to st-hyperpaths and st-disconnecting sets to st-hypercuts in a hypergraph. This relation implies that Menger's theorem w. r. t. paths, Theorem 3.5, is equivalent to Menger's theorem for hypergraphs which is known to be true [50, 65]. We show now that this theorem also follows by a flow argument. More precisely, the max-flow-min-cut theorem also holds for our setting, i.e., with respect to paths.

A weighted version of Menger's theorem, i.e., interpreting the given costs for the paths as capacities, gives the following dual programs.

$$\begin{aligned} \max \quad & \sum_{\mathcal{P}_{st} \in \mathcal{S}_{st}} y_{\mathcal{P}_{st}} & \qquad \min \quad & \sum_{p \in \mathcal{P}} c_p x_p \\ \text{s.t.} \quad & \sum_{\mathcal{P}_{st} \ni p} y_{\mathcal{P}_{st}} \leq c_p \quad \forall\, p \in \mathcal{P} & \qquad \text{s.t.} \quad & \sum_{p \in \mathcal{P}_{st}} x_p \geq 1 \quad \forall\, \mathcal{P}_{st} \in \mathcal{S}_{st} \\ & y_{\mathcal{P}_{st}} \geq 0 \quad \forall\, \mathcal{P}_{st} \in \mathcal{S}_{st} & & x_p \geq 0 \quad \forall\, p \in \mathcal{P}. \end{aligned} \tag{3.1}$$

Here, we denote by $\mathcal{S}_{st}$ the set of all (inclusion wise minimal) st-connecting sets and by $\mathcal{P}_{st} \in \mathcal{S}_{st}$ one element, i.e., an st-connecting set. The left program of (3.1) searches for a maximum number of st-connecting sets such that each path $p \in \mathcal{P}$ is contained in at most c_p st-connecting sets. The right program of (3.1) searches for the minimum weight of an st-disconnecting set. It is the fractional covering problem for the incidence matrix of all inclusion wise minimal st-connecting sets. The left program of (3.1) can also be interpreted as searching for a maximum flow w. r. t. paths. A *flow w. r. t. paths* is a set of pairs $(\mathcal{P}_{st}, k)$, where $\mathcal{P}_{st}$ is an st-connecting set and $k \geq 0$ represents a flow along $\mathcal{P}_{st}$ from s to t, with the property that for each path the sum of the flow along every connecting set containing the path is less than or equal to the capacity of the path. Indeed, this is the description of a flow given by Ford and Fulkerson [49] applied to our setting. We then get the following result which we will show to be valid in the following.

Theorem 3.6 (max-flow-min-cut theorem w. r. t. paths). *The minimum weight of an st-disconnecting set is equal to the maximum flow w. r. t. paths.*

Setting $c \equiv 1$ in (3.1) and in Theorem 3.6 yields Menger's theorem w. r. t. paths. On the other hand, the max-flow-min-cut theorem w. r. t. paths for $c \in \mathbb{Q}_{\geq 0}$ follows from Menger's theorem w. r. t. paths by multiplying the paths according to their capacities. It also follows with Hoffman's *general* max-flow-min-cut theorem [60]. In fact, Hoffman's general max-flow-min-cut theorem was inspired by the work of Ford and Fulkerson [49]. Hoffman "extracts the essence of the arguments" of Ford and Fulkerson to present the max-flow-min-cut theorem in a more general setting in which results can be applied to directed and undirected graphs without need to reduce one to the other. In the following we restrict ourselves to a special case of this theory that is still more general than the classical max-flow-min-cut theorem, see, e.g., Ahuja, Magnanti, and Orlin [3], but sufficient for our setting.

Let U be a finite set, associate a cost $c_u \geq 0$ to every element $u \in \mathcal{U}$, and let $\mathcal{S} = \{S_1, \dots, S_n\}$ be a set of subsets $S_i \subset U$, $i = 1 \dots, n$. Each set S_i has a linear ordering "$<_i$". The ordering is independent for each set, i.e., it is possible that $\{p, q\} \in S_i \cap S_j$ with $p <_i q$ and $q <_j p$, $S_i, S_j \in \mathcal{S}$. If $p \in S_i \cap S_j$, we define

$$(S_i, p, S_j) = \{q \in S_i : q <_i p\} \cup \{p\} \cup \{r \in S_j : p <_j r\}.$$

We then require the following:

$$\begin{aligned} &\text{If } p \in S_i \cap S_j \text{ then there exist } S', S'' \in \mathcal{S} \text{ with} \\ &S' \subseteq (S_i, p, S_j) \text{ and } S'' \subseteq (S_j, p, S_i). \end{aligned} \tag{3.2}$$

Consider the following linear programs:

$$\begin{aligned} (P)\quad \max \quad & \sum_{S \in \mathcal{S}} y_S & \qquad (D)\quad \min \quad & \sum_{u \in U} c_u x_u \\ \text{s.t.} \quad & \sum_{S \ni u} y_S \leq c_u \quad \forall\, u \in U & \text{s.t.} \quad & \sum_{u \in S} x_u \geq 1 \quad \forall\, S \in \mathcal{S} \\ & y_S \geq 0 \quad \forall\, S \in \mathcal{S} & & x_u \geq 0 \quad \forall\, u \in U \end{aligned}$$

Note that Hoffman attaches to each set $S_i \in \mathcal{S}$ a nonnegative integer r_i such that these values satisfy a supermodularity condition. The setting above follows with $r_i = 1$, $\forall\, S_i \in \mathcal{S}$.

Theorem 3.7 (implied by Hoffman, 1974). *If (3.2) is satisfied and c is a nonnegative integral vector, then the linear programs (P) and (D) have optimal integral solutions.*

We now want to show that we can apply Hoffman's theorem 3.7 to our setting. Let us consider again the graph $G = (V, E)$ and the set of paths $\mathcal{P}$. The set of paths $\mathcal{P}$ can be identified with the set U and $\mathcal{S} =: \mathcal{S}_{st}$ consists of the empty set and all (inclusion wise minimal) st-connecting sets. The ordering of each st-connecting set $\mathcal{P}_{st} \in \mathcal{S}_{st}$ corresponds to the sequence of paths when traversing $\mathcal{P}_{st}$ from s to t. It is easy to see that $\mathcal{S}_{st}$ defined in this way satisfies condition (3.2). Let, finally, c_p be the nonnegative costs associated with each $p \in \mathcal{P}$. Then (P) becomes the left and (D) the right program in (3.1).

Then Theorem 3.7 implies that the two linear programs in (3.1) have optimal integral solutions. Hence, Hoffman's theorem implies the max-flow-min-cut theorem w. r. t. paths and, therefore also Menger's theorem w. r. t. paths.

3.3 A Minimum *st*-Connecting Set Algorithm and a Companion Theorem

There are at least two ways to solve the minimum *st*-connecting set problem by transforming it to another problem. One transformation is to replace each path by a clique of all nodes the path contains; the edge costs in the clique are set to the path cost. The minimum *st*-connecting set problem can then be solved by the common shortest path problem in the resulting undirected graph. Another transformation was given in Subsection 1.2; here, the problem was transformed into a directed shortest path problem in the Steiner connectivity digraph.

These transformations, however, do not mean that the minimum connecting set problem is just a shortest path problem in disguise. To make this argument, we show next that an arc flow reformulation of the shortest path problem does not carry over from the graph case to our path connectivity setting. Indeed, in the graph case, the shortest path problem can be formulated equivalently in terms of cuts and in terms of an arc flow. We show now that an arc flow formulation does not work for the minimum *st*-connecting set problem. To this purpose consider a directed graph (V, A), where each undirected edge of the original graph $G = (V, E)$ is replaced by two directed arcs. We denote by $e(a)$ the undirected arc e corresponding to a and introduce flow variables y_a, for each $a \in A$. An arc flow reformulation of the minimum *st*-connecting set problem would then read:

$$
\begin{aligned}
\min \quad & \sum_{p \in \mathcal{P}} c_p \, x_p & & \\
\text{s.t.} \quad & \sum_{a \in \delta^+(v)} y_a - \sum_{a \in \delta^-(v)} y_a = \begin{cases} 1 & \text{if } v = s \\ -1 & \text{if } v = t \\ 0 & \text{otherwise} \end{cases} & & \forall\, v \in V \\
& \sum_{p \in \mathcal{P} : e(a) \in p} x_p \geq y_a & & \forall\, a \in A \\
& y_a \geq 0 & & \forall\, a \in A \\
& x_p \geq 0 & & \forall\, p \in \mathcal{P}.
\end{aligned}
\tag{3.3}
$$

Indeed, if we require $x_p \in \{0, 1\}$ in (MCS) and (3.3), the solution set of program (3.3) projected onto the space of the x-variables is equal to the solution set of (MCS). However, as an LP, the cut formulation (MCS) dominates the flow formulation (3.3), see Figure 3.1 for an example. Of course, for the case $|p| = 1$, $\forall p \in \mathcal{P}$, both formulations are equal. Another difference to the shortest path case is that the problem of finding a directed minimum cost *st*-connecting set, i.e., $\mathcal{P}$ consists of directed paths which are defined in a directed graph, is $\mathcal{NP}$-hard. We will show this in Subsection 3.5.

In the remainder of this section, we propose an algorithm to find an *st*-connecting set with minimum cost without transforming the problem. Our Algorithm does not only have complexity advantages, it is also instrumental in showing that the inequality system of problem (MCS) is TDI. This implies the validity of the companion Menger theorem w. r. t. paths, Theorem 3.4.

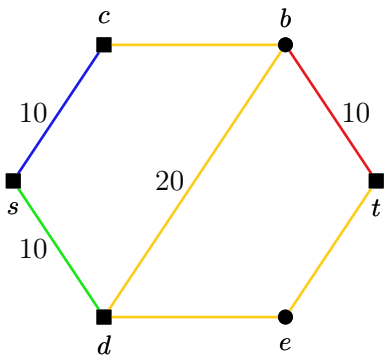


Figure 3.1: An instance of an undirected minimum st-connecting set problem. The minimum st-connecting set has cost 30. The flow formulation 3.3 allows a fractional solution with objective value 25 by setting the path variables to 0.5, i.e., cut and flow formulation are not equal.

A linear system $Ax \geq b$ is *totally dual integral (TDI)* if the linear program $\min\{cx : Ax \geq b\}$ has an integral optimal dual solution y for every integral c for which the linear program has a finite optimum, see, e.g., Cornuéjols [38].

Algorithm 3.6 generalizes Dijkstra's algorithm to our setting. It computes a "minimum path-connected graph for s" with respect to path costs, i.e., the minimum path-connected graph for s contains minimum cost sv-connecting sets for all nodes $v \in V$ considered in line 4. If we replace $t \notin W_i$ by $W_i \neq V$ in the while-loop (line 3), the minimum path-connected graph for s contains minimum cost sv-connecting sets for all nodes $v \in V$. The distances from node s are stored in node labels $d(v)$. The algorithm also computes a dual solution for program (MCS), i.e., a solution for the following program

$$\begin{aligned} \max \quad & \sum_{W \in \mathcal{W}} y_W & & \\ \text{s.t.} \quad & \sum_{W \in \mathcal{W}: p \in \mathcal{P}_{\delta(W)}} y_W \leq c_p & \forall\, p \in \mathcal{P} & \qquad (3.4) \\ & y_W \geq 0 & \forall\, W \in \mathcal{W}, & \end{aligned}$$

where $\mathcal{W} = \{W \subseteq V \backslash \{t\} : s \in W\}$.

More precisely, Algorithm 3.6 works as follows. In an initial step, the distance to s is set to 0 and to infinity for all other nodes; we have $v_0 = s$ and $W_0 = \emptyset$. We assume that all dual variables, i.e., variables of (3.4), are 0 in the initial step. The algorithm iterates as long as the node t is not reached (line 3). Assume that h is the last iteration of the while loop (line 3). In iteration $i = 1, \ldots, h$, the node v_i with the minimum distance is considered (line 4). Note that $v_1 = s$. Then the distances of all nodes of a path containing v are updated, lines 5 to 13. The node v_i is added to W_{i-1} and the dual variable $y_{W_{i-1}}$ is set, lines 15 and 14. The sets W_i, $i = 1, \ldots, h$ produce a sequence of nested cuts $\delta(W_i)$. The distance labels for the nodes v_i, $i = 1, \ldots, h$, are nondecreasing for increasing i.

Algorithm 3.6 is correct. It computes optimal integer solutions for (MCS) and its

Algorithm 3.6: Primal-dual minimum st-connecting set algorithm.

Input : A connected graph $G = (V, E)$, a set of paths $\mathcal{P}$ with costs $c \in \mathbb{R}^{\mathcal{P}}_{\geq 0}$ that covers all edges E, $s, t \in V$.

Output: A minimum cost st-connecting set $\mathcal{P}' \subseteq \mathcal{P}$; solutions for x and y for (MCS) and (3.4).

$d(s) := 0$, $d(v) := \infty \; \forall v \in V \setminus \{s\}$, $n(v) := s$, $p(v) := \emptyset \; \forall v \in V$
$v_0 := s$, $W_0 := \emptyset$, $i := 1$, $y_W := 0 \; \forall W \in \mathcal{W}$, $x_p := 0$, $\forall p \in \mathcal{P}$, $\mathcal{P}' = \emptyset$
while $t \notin W_{i-1}$ **do**
    $v_i = \text{argmin } \{d(u) : u \in V \setminus W_{i-1}\}$
    for all $p \in \mathcal{P} \setminus \mathcal{P}_{\delta(W_{i-1})}$ with $v_i \in p$ **do**
        for all $w \in V(p) \setminus W_{i-1}$ **do**
            if $d(w) > d(v_i) + c_p$ **then**
                $d(w) := d(v_i) + c_p$
                $n(w) := v_i$
                $p(w) := p$
            end
        end
    end
    $y_{W_{i-1}} := d(v_i) - d(v_{i-1})$
    $W_i := W_{i-1} \cup \{v_i\}$
    $i := i + 1$
end
$k := 1$, $u_k := t$
while $u_k \neq s$ **do**
    $\mathcal{P}' := \mathcal{P}' \cup \{p(u_k)\}$
    $x_{p(u_k)} := 1$
    $u_{k+1} := n(u_k)$
    $k := k + 1$
end
return $\mathcal{P}'$, x, y

dual (3.4). We show this by proving that, in particular, the inequality system of (MCS) is TDI.

Proposition 3.8. *The inequality system of (MCS) is TDI.*

Proof. If $c_p < 0$ for a $p \in \mathcal{P}$ then (MCS) has no finite solution since x is not bounded from above; with $x_p \to \infty$ we can improve the objective. This means that we can assume a nonnegative integer cost vector c in the following. We prove the claim by showing that the primal-dual Algorithm 3.6 constructs optimal integral solutions x for (MCS) and y for the linear program (3.4), respectively, with the same objective value.

The algorithm adds nodes v_i to sets $W_i = \{v_1, \ldots, v_i\}$ in the order of increasing distance $d(v_i)$ from $s = v_0 (= v_1)$, i.e., $d(v_{i-1}) \leq d(v_i) \leq \infty$, $i = 1, \ldots, h$, with h being the

last iteration of the while loop 3. This produces a sequence of nested cuts $\delta(W_i)$, $i = 1, \dots, h-1$.

We first show that y is a solution of program (3.4). Lines 2 and 14 imply $y \geq 0$. In fact, the variables y_W can take positive values only for $W \in \{W_1, \dots, W_{h-1}\}$. It remains to show that

$$\sum_{W \in \mathcal{W}: p \in \mathcal{P}_{\delta(W)}} y_W \leq c_p \qquad \forall\, p \in \mathcal{P}. \tag{3.5}$$

Let $p \in \mathcal{P}$. If $v_i \notin p$ for all $i = 1, \dots, h-1$, then $p \notin \mathcal{P}_{\delta(W_i)}$, $i = 1, \dots, h-1$, i.e., $\sum_{W \in \mathcal{W}: p \in \mathcal{P}_{\delta(W)}} y_W = 0 \leq c_p$. Otherwise let $1 \leq i < h$ be the minimal index smaller than h such that $v_i \in p$, i.e., $p \notin \mathcal{P}_{\delta(W_j)}$ for all $0 \leq j < i < h$ but $p \in \mathcal{P}_{\delta(W_i)}$. Let similarly $i \leq \ell < h$ be the maximal index smaller than h such that $V(p) \not\subseteq W_\ell$, i.e., we have $p \in \mathcal{P}_{\delta(W_j)}$ for $i \leq j \leq \ell < h$ and we have $p \notin \mathcal{P}_{\delta(W_j)}$ for $\ell < j < h$. Then equation (3.5) becomes:

$$\begin{aligned}\sum_{W \in \mathcal{W}: p \in \mathcal{P}_{\delta(W)}} y_W = \sum_{j=i}^{\ell} y_{W_j} &= \sum_{j=i}^{\ell} d(v_{j+1}) - d(v_j) \\ &= d(v_{\ell+1}) - d(v_i) \leq d(v_i) + c_p - d(v_i) = c_p.\end{aligned}$$

For the last inequality we distinguish the cases $v_{\ell+1} \in p$ and $v_{\ell+1} \notin p$. The first case follows since $v_i \in p$. In the second case, $v_{\ell+1} = v_h = t$ and there exists a node $w \in p$ with $w \notin W_{h-1}$. Since $d(v_{\ell+1}) = d(t) \leq d(w)$ and $w, v_i \in p$, the second case follows analogously.

We now show that x is a solution of (MCS). Due to the definition of x we have $x \geq 0$. We have to show that

$$\sum_{p \in \mathcal{P}_{\delta(W)}} x_p \geq 1 \qquad \forall\, s \in W \subseteq V \backslash \{t\}. \tag{3.6}$$

Consider the nodes $t = u_1, \dots, u_k = s$ computed in the while loop starting in line 19 and an st-cut $\delta(W)$. Let i be the largest index with $u_i \notin W$ and $u_{i+1} \in W$. This index exists since $u_1 = t \notin W$ and $u_k = s \in W$. Then we have $x_{p(u_i)} = 1$, $p(u_i) \in \mathcal{P}_{\delta(W)}$, and inequality (3.6) is satisfied.

The objective value of program (3.4) is

$$\sum_{i=1}^{h-1} y_{W_i} = \sum_{i=1}^{h-1} d(v_{i+1}) - d(v_i) = d(v_h) - d(v_1) = d(t) - d(s) = d(t).$$

Using lines 18 to 24 and 8 to 10 in Algorithm 3.6 we get

$$\begin{aligned}d(t) = d(u_1) &= d(u_2) + c_{p(u_1)} = d(u_3) + c_{p(u_2)} + c_{p(u_1)} = \dots \\ &= d(u_h) + \sum_{i=1}^{h-1} c_{p(u_i)} = 0 + \sum_{p \in \mathcal{P}'} c_p = \sum_{p \in \mathcal{P}} c_p\, x_p,\end{aligned}$$

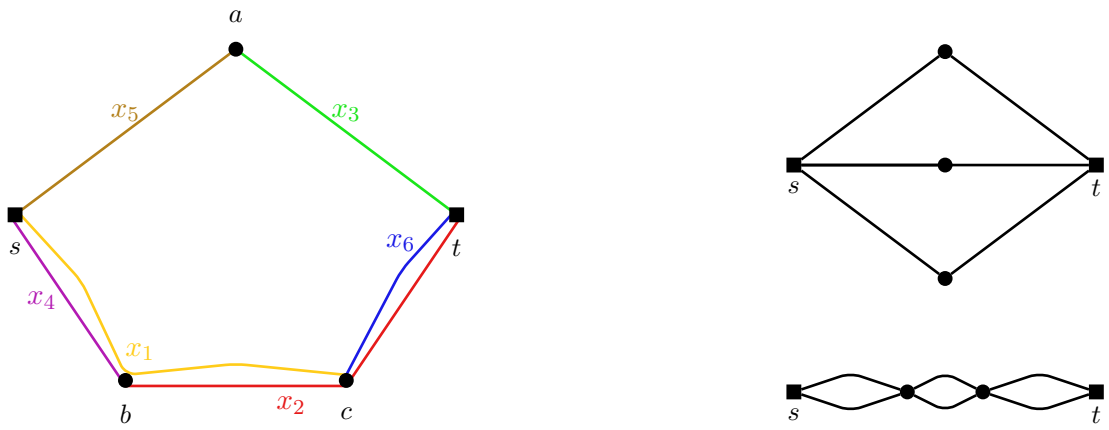


Figure 3.2: *Left: Right:* Example of a graph for which the (minimal) st-disconnecting sets do not produce a totally unimodular incidence matrix. *Upper right:* A graph with six edges in which every minimal st-cut contains exactly three edges. *Lower right:* A graph with six edges and each st-path contains exactly three of them.

i.e., the objective values of (MCS) and (3.4) are equal. The integrality of x is obvious. Since c_p is integral, it follows that $d(v_i)$ is integral for $i = 0, \dots, h-1$. Therefore y_{W_i}, $i = 1, \dots, h-1$, is also integral (line 14). This shows the claim. □

Setting $c \equiv 1$, Proposition 3.8 turns into the companion Menger theorem w. r. t. paths, Theorem 3.4.

3.4 Ideal Matrices

Theorems 3.4 and 3.5 show that the incidence matrix of all inclusion wise minimal st-disconnecting sets $\mathcal{A}_d$ and the incidence matrix of all inclusion wise minimal st-connecting sets $\mathcal{A}_c$ form a blocking pair of ideal $\{0,1\}$-matrices. Note that these matrices are in general not totally unimodular. A matrix is *totally unimodular* if all its square submatrices have a determinant equal to 0, 1, or -1. Consider the left of Figure 3.2. The incidence matrix $\mathcal{A}_d$ whose rows corresponds to all inclusion wise minimal st-disconnecting sets for this example is

$$\begin{array}{c} \\ \{s,a\} \\ \{s,a,b,c\} \\ \{s,b\} \\ \{s\} \\ \{s,a,b\} \\ \{s,b,c\} \end{array} \begin{array}{c} \begin{array}{cccccc} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \end{array} \\ \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array} = \mathcal{A}_d.$$

The 3×3 matrix in the upper left corner has determinant -2 and, therefore, $\mathcal{A}_d$ is not totally unimodular.

The example also shows that the class of incidence matrices of st-disconnecting sets genuinely generalizes the class of incidence matrices of st-cuts, i.e., this class contains more matrices. This can be seen as follows. If $\mathcal{A}_d$ were be an incidence matrix of st-cuts in some graph, the columns of $\mathcal{A}_d$ would have to correspond to the edges of a graph, i.e., such a graph would have to have six edges. Each cut of the graph contains exactly three edges, i.e., the edge-degrees of s and t have to be three. Furthermore, there cannot be an edge connecting s and t since this edge would be contained in every cut. The only possible graph satisfying these conditions is shown on the upper right of Figure 3.2. But this graph has eight minimal st cuts instead of the six in matrix $\mathcal{A}_d$.

The matrix $\mathcal{A}_d$ also cannot be an incidence matrix of st-paths in an undirected graph $G = (V, E)$; this graph would have six edges and cannot contain a Steiner path bridge. Then each (minimal) path from s to t would have to contain exactly three edges. The only possible graph of this type is shown on the lower right of Figure 3.2. But this graph has eight minimal st-paths.

3.5 Directed Paths

In this section, we want to show that, unlike in the graph case, the problem of finding a minimum cost st-connecting set w. r. t. directed paths cannot be handled in the same way as the undirected case. In fact, in contrast to the undirected case, this problem is $\mathcal{NP}$-hard.

Let $G = (V, A)$ be a directed graph with no parallel arcs and loops. We are given a set of directed paths $\vec{\mathcal{P}}$ with nonnegative costs $c_p \geq 0$, $p \in \vec{\mathcal{P}}$. We again assume that each arc of G is covered by a path $p \in \vec{\mathcal{P}}$; otherwise the (uncovered) arc can be removed. Let, furthermore, $s, t \in V$ be two specific nodes of G. We call $\vec{\mathcal{P}}' \subseteq \vec{\mathcal{P}}$ a *directed st-connecting set* if there exists a directed st-path in the subgraph $H = (V, A(\vec{\mathcal{P}}'))$, where $A(\vec{\mathcal{P}}') = \{a \in A : \exists p \in \vec{\mathcal{P}}' \text{ with } a \in p\}$. The *minimum directed st-connecting set problem* is to find a directed st-connecting set $\vec{\mathcal{P}}' \subseteq \mathcal{P}$ with minimum cost. Figure 3.4 shows an example of this natural problem.

Proposition 3.9. *The minimum directed st-connecting set problem is strongly $\mathcal{NP}$-hard, even for unit costs.*

Proof. We reduce the set covering problem to the minimum directed st-connecting set problem. Recall that in a set covering problem we are given a finite set S, a set $\mathcal{M} \subseteq 2^S$, and a positive integer k. The problem is to find a subset $\mathcal{M}' \subseteq \mathcal{M}$, $|\mathcal{M}'| \leq k$, such that for all $s \in S$ there exists an $M \in \mathcal{M}'$ with $s \in M$.

Given a set covering instance, let $S = \{1, \ldots, n\}$ be an arbitrary but fixed order of the elements. We define a directed graph $G = (V, A)$ with node set

$$V = \{1, 2, \ldots, n, n+1\},$$

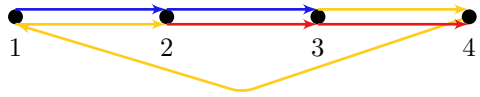


Figure 3.3: Transforming a set covering problem to a direct st-connecting set problem. We are given $S = \{1, 2, 3\}$ and the sets $M_1 = \{1, 2\}$ (corresponding to the blue path), $M_2 = \{1, 3\}$ (corresp. to the yellow path), and $M_3 = \{2, 3\}$ (corresp. to the red path).

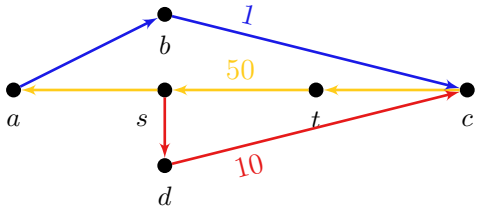


Figure 3.4: An instance of a minimum directed st-connecting set problem. A minimum directed st-connecting set $\mathcal{P}'$ has cost 51 and covers nodes a, b, and c. The minimum (s, c)-connecting set $\vec{\mathcal{P}}''$ has cost 10 and $\vec{\mathcal{P}}' \cap \vec{\mathcal{P}}'' = \emptyset$.

where each element i in S corresponds to node i in V plus one additional node $n + 1$. We define the following arcs

$$\begin{aligned} &(i, i+1) \quad i = 1, \ldots, n \\ &(i+1, j) \quad i, j = 1, \ldots, n,\ i > j. \end{aligned}$$

For each set $M \in \mathcal{M}$ consider the elements in the natural order and group them such that the groups consist of consecutive elements, e.g.,

$$M = \{j_1, j_1 + 1, \ldots, j_1 + k_1, j_2, \ldots, j_2 + k_2, \ldots, j_r, \ldots, j_r + k_r\},$$

with $r \geq 1$, $k_i \in \{0, \ldots, n-1\}$, $i = 1, \ldots, r$.

Construct a path p with cost $c_p = 1$ as follows

$$p = (j_r, \ldots, j_r + k_r + 1, j_{r-1}, \ldots, j_{r_1} + k_{r-1} + 1, j_1, \ldots, j_1 + k_1 + 1)$$

After constructing all paths, we remove all uncovered arcs in G. An example for this construction is illustrated in Figure 3.3.

The construction has the following properties:

1. A set M contains element i, $i \in \{1, \ldots, n\}$ if and only if the corresponding path contains arc $(i, i + 1)$.
2. Let $W_i = \{1, \ldots, i\}$, $i = \{1, \ldots, n\}$. The directed $(1, n + 1)$-cut $\delta^+(W_i)$ contains only arc $(i, i + 1)$. All minimal directed $(1, n + 1)$ cuts are of this form.

We have to show that a cover $\mathcal{M}'$ with at most k elements exists if and only if there exists a directed $(1, n + 1)$-connecting set with cost less than or equal to k.

"$\Rightarrow$": Assume $\mathcal{M}'$ is a cover with at most k elements. Choose

$$\vec{\mathcal{P}}' = \{p \in \vec{\mathcal{P}} : p \text{ corresponds to a set } M \in \mathcal{M}'\}.$$

$\vec{\mathcal{P}}'$ has cost $\leq k$. Since $\mathcal{M}'$ is a cover and using observation 1, each arc $(i, i + 1)$, $i = 1, \ldots, n$, is covered by a path $p \in \vec{\mathcal{P}}'$. Hence, the directed $(1, n + 1)$-path $(1, 2, \ldots, n + 1)$ is covered by $\vec{\mathcal{P}}'$ and $\vec{\mathcal{P}}'$ is a directed st-connecting set.

"⇐": Assume $\vec{\mathcal{P}}'$ is a directed $(1, n+1)$-connecting set with cost less than or equal to k. Choose $\mathcal{M}' = \{M \in \mathcal{M} : M \text{ corresponds to a path } p \in \vec{\mathcal{P}}'\}$; $|\mathcal{M}'| \leq k$. Due to observation 2 each arc $(i, i+1)$, $i = 1, \ldots, n$, is covered by a path $p \in \vec{\mathcal{P}}'$. With observation 1, we get that $\mathcal{M}'$ covers all elements of S. □

Figure 3.4 illustrates that a minimum cost directed st-connecting set $\vec{\mathcal{P}}'$ does not contain a minimum cost directed sc-connecting set for a node c that "lies" on the st-path induced by $\vec{\mathcal{P}}'$. This is in contrast to the undirected case and to the directed graph case; it shows that Algorithm 3.6 cannot be used to compute a minimum directed st-connecting set.

3.6 Connectivity Problems w. r. t. Paths

Considering a graph with nodes and edges, a natural question is to determine how connected it is. The concepts of k-edge-connected and k-node-connected graphs try to answer this question. A graph is k-edge-connected (k-node-connected) if it is still connected after deleting any set of fewer than k edges (nodes). A well-known consequence of Menger's theorem is that a graph is k-edge-connected if and only if there exist k edge-disjoint paths between every two nodes. In this section we will see that we can generalize the k-connectivity concept to our setting to answer the question how "path-connected" a graph is.

Definition 3.10. *A graph G is* k-path-connected *w. r. t. $\mathcal{P}$ if G is path-connected after deleting any set $\mathcal{P}' \subseteq \mathcal{P}$ of fewer than k paths, i.e., $G' = (V, E(\mathcal{P} \setminus \mathcal{P}'))$, $|\mathcal{P}'| < k$, is connected.*
A graph G is k-edge-path-connected *w. r. t. $\mathcal{P}$ if G is path-connected after deleting any set of fewer than k edges E' with all incident paths, i.e., $G' = (V, E(\tilde{\mathcal{P}}))$ with $\tilde{\mathcal{P}} = \mathcal{P} \setminus \{p \in \mathcal{P} : |E' \cap E(p)| \geq 1\}$ is connected.*
A graph G is k-node-path-connected *w. r. t. $\mathcal{P}$ if G is path-connected after deleting any set of fewer than k nodes V' with all incident paths, i.e., $G' = (V \setminus V', E(\tilde{\mathcal{P}}))$ with $\tilde{\mathcal{P}} = \mathcal{P} \setminus \{p \in \mathcal{P} : |V' \cap V(p)| \geq 1\}$ is connected.*

Note that we already needed the definition of a *2-node-path-connected* graph in Chapter 2 and we will also need this definition in Chapter 7.

It is easy to see that the following holds: G is k-edge-path-connected $\Rightarrow$ G is k-path-connected. Definition 3.10 implies that each disconnecting set in a k-path-connected graph has to contain at least k elements. Hence, a consequence of Menger's theorem w. r. t. paths, Theorem 3.5, is the following result which is similar as for the graph case.

Corollary 3.11. *A graph G is k-path-connected if and only if there exist k path-disjoint st-connecting sets between every two nodes $s, t \in V$ ($\Leftrightarrow |\mathcal{P}_{\delta(W)}| \geq k \; \forall\, s \in W \subseteq V \setminus \{t\}$).*

The statement in brackets follows with the equivalence of minimal T-disconnecting sets and T-path cuts (Lemma 1.1).

Corollary 3.12. *The problem to decide whether a graph G is $(k+1)$-path-connected w. r. t. $\mathcal{P}$ can be solved in polynomial time.*

Proof. This can be done by a maximum flow computation in the Steiner connectivity digraph D', compare with Section 1.2. For every pair of nodes $s, t \in V$ we construct D' with $T = \{s, t\}$, setting the root node to $r := s$ and the edge capacities to 1. It is easy to see that every two edge-disjoint st-paths in D' give rise to two path-disjoint st-connecting sets in G and vice versa. The maximum st-flow in the Steiner connectivity digraph corresponds to the maximum number of edge-disjoint st-paths in D'. Hence, if the maximum flow is at least $(k+1)$ in every such graph D', i.e., for all $s, t \in V$, then the graph G is $(k+1)$-path-connected w. r. t. $\mathcal{P}$. Overall, we have to solve $\mathcal{O}(|V|^2)$ maximum flow problems. □

Note for $k = 1$: Since we assume that each edge is covered by a path, a graph is path-connected if and only if the graph is connected. Trivially, k-edge-path-connectivity, k-node-path-connectivity and k-path-connectivity are equivalent for $k = 1$ since for this case nothing is removed.

Remark 3.13. *If we interpret a path p in a graph G as a line in a transportation network, a k-line-connected network guarantees that there still exists a connection between any two nodes when $k-1$ lines cannot be operated. Requiring a k-line-connected network can, hence, be interpreted as a kind of survivability condition on a line plan.*

Proposition 3.14. *The problem to decide whether a graph G is $(k+1)$-edge-path-connected with respect to a set of paths $\mathcal{P}$ is co-$\mathcal{NP}$-hard.*

Proof. When we are given a set of edges E' with $|E'| \leq k$, we can delete these edges with all incident paths and check whether the resulting graph is connected, i.e., we can check a no-instance in $\mathcal{O}(k|\mathcal{P}| + |V| + |E|)$. Hence, the problem is in co-$\mathcal{NP}$.

The idea of this proof is to reduce the minimum vertex cover problem to the $(k+1)$-edge-path-connectivity problem. Let $\overline{G} = (\overline{V}, \overline{E})$ be a graph and $k < |\overline{V}|$ be a positive integer. The minimum vertex cover problem is to decide whether there exists a set $V' \subset \overline{V}$ with $|V'| \leq k$ and each edge has at least one endpoint in V'.

Given an instance of the vertex-cover problem, we construct an instance of the k-edge-path-connectivity problem as follows: Define a graph $G = (V, E)$ with node set

$$V = \{s, t\} \cup \{u_v, w_v : v \in \overline{V}\}$$

and edge set

$$E = \{\{s, u_v\}, \{u_v, w_v\}, \{w_v, t\} : v \in \overline{V}\} \cup \{\{u_v, u_{\tilde{v}}\}, \{w_v, w_{\tilde{v}}\}\} : v, \tilde{v} \in \overline{V},\ v \neq \tilde{v}\}.$$

All edges that are not covered by paths after their construction are deleted afterwards. We have a path of length 1 for each edge $e \in E$, $e \neq \{u_v, w_v\}$, $v \in \overline{V}$. Furthermore, we construct the following paths corresponding to edges in $\overline{G}$:

$$p_{v\tilde{v}} = (w_v, u_v, u_{\tilde{v}}, w_{\tilde{v}}) \quad \text{for all } e = \{v, \tilde{v}\} \in \overline{E} \quad (p_{vv} = (u_v, w_v) \text{ if } \{v, v\} \in \overline{E}).$$

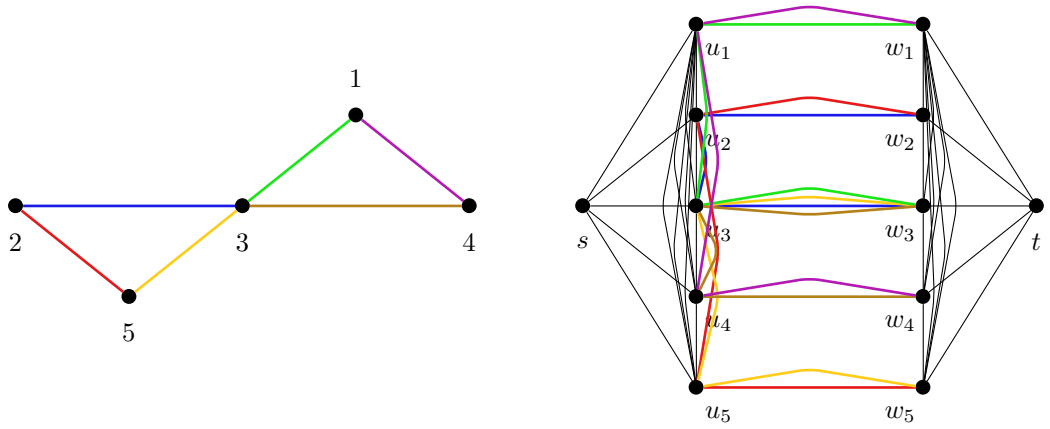


Figure 3.5: Construction of the transformation in the proof of Proposition 3.14. *Left:* A vertex cover instance. *Right:* The corresponding transformation into an edge-path-connectivity instance. There is only one edge $\{u_i, w_i\}$, $i = 1, \ldots, 5$, the paths containing such an edge are plotted on "parallel lines" to identify them.

Figure 3.5 gives an example of this transformation. We claim that G as defined above is not $(k+1)$-edge-path-connected w. r. t. the constructed paths if and only if there exists a vertex cover $V' \subset \overline{V}$ in $\overline{G}$ with $|V'| \leq k$.

"$\Rightarrow$:" Let G be not $(k+1)$-edge-path-connected w. r. t. the constructed paths. Then there exists a set E^* with $|E^*| \leq k$ such that the graph G^*, which results from deleting E^* with all incident paths, is not path-connected. Since the two node sets $\{s\} \cup \{u_v : v \in \overline{V}\} =: U$ and $\{t\} \cup \{w_v : v \in \overline{V}\} =: W$ are cliques, in which each edge is covered by a path of length 1, and $|U|, |W| \geq k+2$, every two nodes in U and every two nodes in W are connected in G^* for any set E^* with $|E^*| \leq k$. This means that E^* has to disconnect the sets U and W, and we can assume w.l.o.g. that E^* contains only edges between U and W. Set $V' = \{v \in \overline{V} : \{u_v, w_v\} \in E^*\}$. Then $|V'| = |E^*| \leq k$. We claim that V' is a vertex cover in $\overline{G}$. Assume it is not. Then there exists an edge $e = \{v, \tilde{v}\} \in \overline{E}$ with $v, \tilde{v} \in \overline{V} \setminus V'$. Due to the construction of V' we have $\{u_v, w_v\}, \{u_{\tilde{v}}, w_{\tilde{v}}\} \notin E^*$. Due to the construction of G the path $(w_v, u_v, u_{\tilde{v}}, w_{\tilde{v}})$ corresponding to $\{v, \tilde{v}\}$ is not incident to an edge in E^* and connects U and W, a contradiction.

"$\Leftarrow$:" Let $V' \subset \overline{V}$ be a vertex cover in $\overline{G}$ with $|V'| \leq k$. Consider the edge set $E^* = \{\{u_v, w_v\} : v \in V'\}$, $|E^*| = |V'| \leq k$. Assume the graph that results from deleting E^* with all incident paths is st-path-connected. Due to the construction of G there has to exist a path $(w_v, u_v, u_{\tilde{v}}, w_{\tilde{v}})$ with $\{u_v, w_v\}, \{u_{\tilde{v}}, w_{\tilde{v}}\} \notin E^*$. This means that $v, \tilde{v} \notin V'$, i.e., the edge $\{v, \tilde{v}\}$ is not covered by V', a contradiction. Hence, G is not k-edge-path connected for the nodes s and t and therefore not k-edge-path connected. □

Proposition 3.15. *The problem to decide whether a graph G is $(k+1)$-node-path-connected with respect to a set of paths $\mathcal{P}$ is co-$\mathcal{NP}$-hard.*

Proof. It is easy to see that the problem is in co-$\mathcal{NP}$. The minimum vertex cover problem can be transformed to the $(k+1)$-node-path-connectivity problem in a similar way as

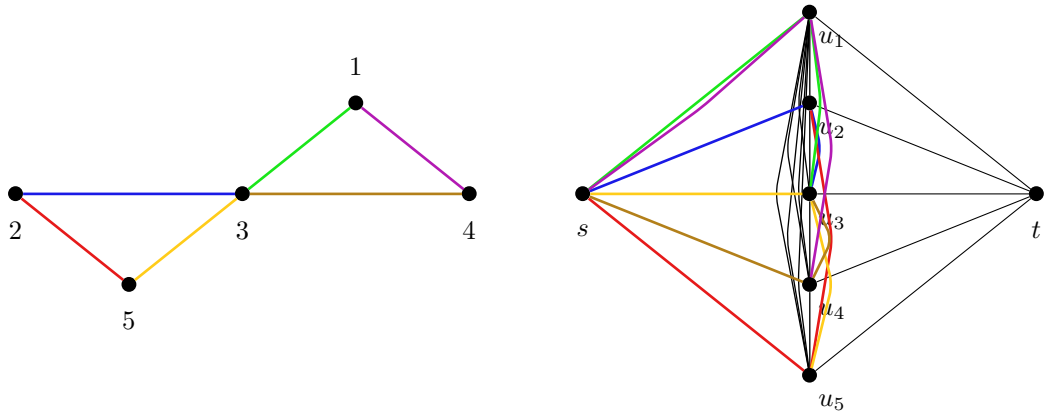


Figure 3.6: Construction of the transformation in the proof of Proposition 3.15. *Left:* A vertex cover instance. *Right:* The corresponding transformation into a node-path-connectivity instance. There is only one edge $\{s, u_1\}$, the two paths containing this edge are plotted on "parallel lines" to identify them.

the $(k+1)$-edge-path-connectivity problem. Given a graph $\overline{G} = (\overline{V}, \overline{E})$ and $k < |\overline{V}|$, define a graph $G = (V, E)$ with node set $V = \{s, t\} \cup \{u_v : v \in \overline{V}\}$ and edge set

$$E = \{\{s, u_v\}, \{u_v, t\}\} : v \in \overline{V}\} \cup \{\{u_v, u_{\tilde{v}}\} : v, \tilde{v} \in \overline{V},\ v \neq \tilde{v}\}.$$

All edges that are not covered by paths after their construction are deleted afterwards. We have a path of length 1 for each edge $e \in E$, $e \neq \{s, u_v\}$, $v \in \overline{V}$. We further construct a path $p_{v\tilde{v}} = (s, u_v, u_{\tilde{v}})$ for each edge $e = \{v, \tilde{v}\} \in \overline{E}$. Figure 3.6 gives an example of this transformation. We claim that the graph G is not $(k+1)$-node-path-connected if and only if there exists a vertex cover for $\overline{G}$ with at most k nodes.

"$\Rightarrow$:" Let G be not $(k+1)$-node-path-connected, i.e., there exists a set of nodes V^*, $|V^*| \leq k$, such that G is not path-connected after deleting V^* with all incident paths. Since the nodes $\{t\} \cup \{u_v : v \in \overline{V}\} =: C$ are a clique with $|C| \geq k+2$, every two nodes $v, w \in C$ with $v, w \notin V^*$ would be still connected after deleting V^* with all incident paths. Hence, the set V^* was chosen such that s is disconnected from a node $v \in C$. With the above argumentation, we get that s hast to be disconnected from all nodes $v \in C = V \setminus \{s\}$, i.e., $s \notin V^*$, and each path incident to s has to contain a node $u \in V^*$. Due to the construction of G, every edge in $\overline{G}$ corresponds to a path incident to s. Choosing $V' = \{v \in \overline{V} : u_v \in V^*\}$ then has to be a vertex cover in $\overline{G}$ with $|V'| = |V^*| \leq k$.

"$\Leftarrow$:" Let $V' \subset \overline{V}$ be a vertex cover in $\overline{G}$ with $|V'| \leq k$. Set $V^* = \{u_v \in V : v \in V'\}$. We show that the graph that results from deleting V^* with all incident paths is not st-path-connected. Assume it is. Then there exists a path $(s, u_v, u_{\tilde{v}})$ with $u_v, u_{\tilde{v}} \notin V^*$, i.e., $v, \tilde{v} \notin V'$, i.e., the edge $\{v, \tilde{v}\}$ is not covered by V', a contradiction. □

Remark 3.16. *Checking a graph for 2-edge-path-connectivity (2-node-path-connectivity) can be done in $\mathcal{O}(|E|^2|\mathcal{P}|)$ ($\mathcal{O}(|V||E||\mathcal{P}|)$. There are $|E|$ ($|V|$) edges (nodes) to check. Checking one edge (node) means to remove this edge (node) and all incident paths. Here, all edges that are not covered by a path any more are also removed. Removing*

all uncovered edges can be done in $\mathcal{O}(|\mathcal{P}||E|)$. *Then one can use depth first search to check connectivity in the resulting graph. This takes* $\mathcal{O}(|V| + |E|)$. *In total we get* $\mathcal{O}(|E|(|E||\mathcal{P}|+|V|+|E|))$ *to check* 2*-edge-path-connectivity and* $\mathcal{O}(|V|(|E||\mathcal{P}|+|V|+|E|))$ *to check* 2*-node-path-connectivity*

Chapter 4

Solving the Steiner Connectivity Problem

In this chapter, we discuss the computational solution of the Steiner connectivity problem. We implemented the undirected cut formulation as well as the (extended) directed cut formulation in the branch-and-cut framework SCIP [2, 95]. The extended directed cut formulation has a quadratic number of variables compared to the undirected one and, as it will turn out, cannot be solved directly for large-scale instances. We, therefore, investigated cutting plane methods for the undirected cut formulation that use the strength of the directed cut formulation and the strength of the Steiner partition inequalities, respectively. These cutting plane methods are explained in Section 4.1 while two primal heuristics are described in Section 4.2. Section 4.3 presents our computational results. Using the methods introduced in this chapter, we show that large Steiner connectivity problems with up to 900 nodes can be solved within reasonable optimality gaps of typically less than five percent.

4.1 Cutting Planes for the Undirected Cut Formulation

We have seen in Chapter 2 that the extended directed cut formulation ($\text{SCP}^r_{\text{arc+}}$) implies strong inequalities for the canonical undirected cut formulation (SCP_{cut}), e. g., the facet defining Steiner partition inequalities. In the following, we will consider two strategies to carry over the strength of the directed cut formulation to the undirected cut formulation. First, we consider a fast heuristic separation method for the Steiner partition inequalities, called *SPI separation*, in Subsection 4.1.1. Encouraged by its effectiveness, we then propose a *partial projection method*: The idea is to lift an LP solution from the space of canonical variables to the extended space, to separate there, and to project the cut back. This algorithm can identify additional cuts and is described in Section 4.1.2 in detail. A problem of this procedure is that the separation step is based on an exponential system

Algorithm 4.7: SPI separation to detect violated Steiner partition inequalities

Input : A connected graph $G = (V, E)$, a set of paths $\mathcal{P}$ with costs $c \in \mathbb{R}^{\mathcal{P}}_{\geq 0}$, a set of terminal nodes $T \subseteq V$, and a fractional solution $x^* \in P_{LP}(\mathrm{SCP}_{\mathrm{cut}})$.

Output: If found, a violated partition inequality.

$\omega_e := \sum_{p \in \mathcal{P}: e \in p} x^*_p$, $\forall\, e \in E$, $R(v) = \{v\}$, $\forall\, v \in V$, $P = \{\{v\} : v \in V\}$
sort edge weights s.t. $\omega_{e_1} \geq \omega_{e_2} \geq \ldots \geq \omega_{e_{|E|}}$, let $\{v_i, w_i\} = e_i$
for $i = 1, \ldots, |E|$ **do**
  if $R(v_i) \cap T = \emptyset$ or $R(w_i) \cap T = \emptyset$ or $\omega_{e_i} \geq 1$ **then**
    $P := P \setminus \{R(v_i), R(w_i)\} \cup \{R(v_i) \cup R(w_i)\}$
    $R(w_i) = R(v_i) := R(w_i) \cup R(v_i)$
  end
end
// P is a Steiner partition
for $i = 1, \ldots, |E|$ **do**
  // if both nodes are in different sets
  if $R(v_i) \cap R(w_i) = \emptyset$ **then**
    if $\sum_{p \in \mathcal{P}} a_p\, x^*_p < |P| - 1,\ a_p := |\{|V(p) \cap W| \geq 1 : W \in P\}| - 1, p \in \mathcal{P}$ **then**
      return violated Steiner partition inequality
    end
    $P := P \setminus \{R(v_i), R(w_i)\} \cup \{R(v_i) \cup R(w_i)\}$
    $R(w_i) = R(v_i) := R(w_i) \cup R(v_i)$
  end
end

of inequalities. This is as difficult as working directly with the extended formulation. We suggest to consider a relaxation to separate cuts from a subsystem of the extended formulation. This subsystem has to be chosen in such a way that it is, on the one hand, of tractable size and, on the other hand, produces strong cuts. We discuss a possible choice of a good subsystem in Section 4.1.3. The resulting partial projection method can be used to approximate the extended formulation. Since the extended formulation includes not only Steiner partition inequalities but also other facet defining inequalities, the partial projection method can in principle find other types of inequalities that cannot be identified by SPI separation, see the upcoming Example 4.2.

4.1.1 Separating Steiner Partition Inequalities

We propose a heuristic, called SPI separation, to find a Steiner partition of the nodes that has a good chance to yield a violated Steiner partition inequality. It is motivated by a graph shrinking procedure that produces a promising Steiner partition of the nodes for the STP, see, e. g., Grötschel et al. [58] and Günlük [59]. It works as follows in our case. Let $x^* \in P_{LP}(\mathrm{SCP}_{\mathrm{cut}})$ be some fractional solution and $\omega_e := \sum_{p \in \mathcal{P}: e \in p} x^*_p$ be edge

weights for all $e \in E$. Sort the edges as $e_1, \dots, e_{|E|}$ such that $\omega_{e_1} \geq \dots \geq \omega_{e_{|E|}}$. We now recursively shrink the graph by contracting edges in the order of decreasing weight. In a first step, we shrink edges as long as the edge weight is greater than or equal to 1 or if one end node of an edge is not a terminal node. If an end node of an edge is a terminal node, the node that arises from shrinking this edge is also defined as a terminal node. After this first shrinking procedure each shrunk node contains at least one terminal node and is, therefore, also a terminal node. If the resulting shrunk graph contains more than one node, the weights of all remaining edges are smaller than 1. Each shrunk node can be considered as a partition. We then compute the resulting Steiner partition inequality. If it is violated, we can add it to the problem. If it is not violated, we shrink the edge with the largest weight in the shrunk graph and consider the Steiner partition inequality associated with the resulting graph. This procedure terminates as soon as we have found a violated cut or when the shrunk graph contains only one node. A detailed description of the SPI separation heuristic is given in Algorithm 4.7.

4.1.2 Separating Cuts from the Extended Formulation

A direct method to use the extended formulation ($\text{SCP}^r_{\text{arc}^+}$) to separate cuts for the canonical undirected cut formulation (SCP_{cut}) is as follows: Let $x^* \in [0,1]^{\mathcal{P}}$ be the point to be separated; denote $\tilde{A} = A'_{\mathcal{P}} = \{(v_p, w_p) \in A' \mid p \in \mathcal{P}\}$ and $A'' = A' \setminus A'_{\mathcal{P}}$. Consider the inequality system associated with ($\text{SCP}^r_{\text{arc}^+}$) for $y|_{\mathcal{P}} = x^* =: y^*$:

$$\begin{aligned}
\sum_{a \in \delta^-(W), a \in A''} y_a &\geq 1 - \sum_{a \in \delta^-(W), a \in \tilde{A}} y^*_a && \forall\, W \in \mathcal{W} \\
-\sum_{a \in \delta^-(v_p)} y_a &\geq -y^*_{v_p w_p} && \forall\, v_p \in V' \,(p \in \mathcal{P} : r \notin p) \\
\sum_{a \in \delta^+(w_p)} y_a &\geq y^*_{v_p w_p} && \forall\, w_p \in V' \,(p \in \mathcal{P} : t \notin p \,\forall t \in T) \\
y_a &\geq 0 && \forall\, a \in A''.
\end{aligned} \tag{4.1}$$

where $\mathcal{W} := \{W \subseteq V' \setminus \{r\} \,:\, W \cap T \neq \emptyset\}$ is the set of all directed Steiner cuts associated with root node r. We want to decide whether there is a vector $y \in P_{LP}(\text{SCP}^r_{\text{arc}^+})$ with $y|_{\mathcal{P}} = x^*$ or to prove that no such vector exists. By the Farkas lemma either inequality system (4.1) or the following inequality system has a solution:

$$\begin{aligned}
\sum_{W \in \mathcal{W}} \Big(1 - \sum_{a \in \delta^-(W), a \in \tilde{A}} y^*_a\Big) \cdot \mu_W - \sum_{p \in \mathcal{P}} y^*_{v_p w_p} (\pi^v_p - \pi^w_p) &> 0 \\
\sum_{W \in \mathcal{W} : a \in \delta^-(W)} \mu_W - \sum_{p \in \mathcal{P} : a \in \delta^-(v_p)} \pi^v_p + \sum_{p \in \mathcal{P} : a \in \delta^+(w_p)} \pi^w_p &\leq 0 \quad \forall\, a \in A'' \\
\mu_W &\geq 0 \quad \forall\, W \in \mathcal{W} \\
\pi^v_p, \pi^w_p &\geq 0 \quad \forall\, p \in \mathcal{P}.
\end{aligned} \tag{4.2}$$

The first inequality of (4.2) gives rise to a violated cut. Namely, if $y \notin P_{LP}(\mathrm{SCP}^r_{\mathrm{arc}^+})$ for $y|_{\mathcal{P}} = y^* = x^*$,

$$\begin{aligned} & \sum_{W \in \mathcal{W}} \Big(1 - \sum_{a \in \delta^-(W), a \in \tilde{A}} y^*_a\Big) \cdot \mu^*_W - \sum_{p \in \mathcal{P}} y^*_{v_p w_p} (\pi^{v*}_p - \pi^{w*}_p) > 0 \\ \iff & \sum_{W \in \mathcal{W}} \Big(1 - \sum_{(v_p, w_p) \in \delta^-(W)} x^*_p\Big) \cdot \mu^*_W - \sum_{p \in \mathcal{P}} x^*_p (\pi^{v*}_p - \pi^{w*}_p) > 0, \end{aligned}$$

and

$$\begin{aligned} & \sum_{W \in \mathcal{W}} \mu^*_W - \sum_{W \in \mathcal{W}} \sum_{(v_p, w_p) \in \delta^-(W)} \mu^*_W x_p - \sum_{p \in \mathcal{P}} (\pi^{v*}_p - \pi^{w*}_p) x_p \leq 0, \\ \iff & \sum_{W \in \mathcal{W}} \mu^*_W \leq \sum_{W \in \mathcal{W}} \sum_{p : (v_p, w_p) \in \delta^-(W)} \mu^*_W \, x_p + \sum_{p \in \mathcal{P}} (\pi^{v*}_p - \pi^{w*}_p) x_p \end{aligned}$$

is a cutting plane that separates x^* from the Steiner connectivity polytope.

The system (4.2) is a feasibility problem that can be solved by minimizing the left-hand-side of the first inequality subject to the remaining system and the additional constraint $||(\mu, \pi^v, \pi^w)|| \leq 1$, where $||\cdot||$ is an arbitrary norm, to bound the variables. There are $2|\mathcal{P}|$ π-variables, which can be treated directly, and $\mathcal{O}(2^{|V'|}) = \mathcal{O}(2^{2|\mathcal{P}|})$ μ-variables, which have to be treated by a column generation procedure. In each iteration, a subset of the cut system $\mathcal{W}$ is considered. When the subset produces a positive objective value, we have found a violated cutting plane. Otherwise, we must increase the subset by generating an improving variable μ_W. Associating dual variables y''_a, $a \in A''$, with the constraints of the feasibility problem (4.2), the pricing problem is to find $W \in \mathcal{W}$ such that

$$\sum_{a \in \delta^-(W), a \in A''} y''_a < 1 - \sum_{a \in \delta^-(W), a \in \tilde{A}} y^*_a$$

or to conclude that no such W exists. This is a minimum directed Steiner cut problem that can be solved in polynomial time, such that the entire separation method has a polynomial complexity. This result is, however, theoretical; a practical version will be discussed in the following subsection.

4.1.3 Separating Cuts by Partial Projection

The system (4.1) is of exponential size and therefore difficult to handle. To make this method practical, we consider a "partial projection" that is based on a relaxation of $(\mathrm{SCP}^r_{\mathrm{arc}^+})$ restricted to a subset of directed (r,t)-Steiner cut inequalities. After extensive computational experiments, the following two types of directed (r,t)-Steiner cuts turned out to be most useful: cuts that are associated with *path neighborhoods* of the terminals and of node sets arising in a shrinking procedure similar to the one described in Section 4.1.1. It turns out that the resulting subsystem is tractable and produces strong cuts. To this purpose we construct a nested family of path neighborhood cuts $\delta^-(W^i_t)$

where W_t^i is of the form of the node sets W_i in the proof of Theorem 2.19. The combinatorial motivation for the choice of these cuts is that they approximate the connectivity requirement of the problem. The path neighborhoods of the first type are iteratively constructed around each terminal node except the root node and are independent of a fractional solution x^*. The path neighborhoods of the second type try to contract the graph in order to identify directed (r,t)-Steiner cuts which likely have small capacity with respect to the LP solution of the undirected formulation.

A formal description of the construction of the path neighborhoods of the first type is as follows. We first choose an arbitrary terminal node as root node $r \in T$. Then we consider for each of the remaining terminal nodes $t \in T \setminus \{r\}$ a node set $V_t^0 = \{t\}$, the set $\mathcal{P}_t^0 \subseteq \mathcal{P}$ of paths that intersect V_t^0 (i. e., contain t), and we define the initial path neighborhood of t as the node set

$$W_t^0 := \{t\} \cup \{w_p \mid p \in \mathcal{P}_t^0\} \cup \{v_p \mid p \in \mathcal{P}_t^0, r \notin p\},$$

i. e., W_t^0 contains t, its neighbors w_p, and for each node w_p its predecessor v_p, provided it is not connected to the root node r. Then $\delta^-(W_t^0)$ forms a directed (r,t)-cut in the Steiner connectivity digraph. In fact, it is easy to see that

$$\delta^-(W_t^0) = \{(v_p, w_p) \in A' : p \in \mathcal{P}_t^0, r \in p\} \cup \{(w_{\tilde{p}}, v_p) \in A' : \tilde{p} \notin \mathcal{P}_t^0, p \in \mathcal{P}_t^0, r \notin p\}.$$

We now grow the sets V_t^0, $\mathcal{P}_t^0$, and W_t^0 iteratively. To this purpose, we define a connected node set $r \notin V_t^1 \supset V_t^0$ such that the set of paths that intersect V_t^1 is minimal, not containing the root node, and increases $\mathcal{P}_t^0$, i. e.,

$$\min_{\substack{V_t^0 \subset V_t^1 \subseteq V \setminus \{r\} \\ V_t^1 \text{ connected}}} |\mathcal{P}_t^1 := \{p \in \mathcal{P} : V_t^1 \cap V(p) \neq \emptyset\}| > |\mathcal{P}_t^0|,$$

and we obtain the first path neighborhood

$$W_t^1 := \{t\} \cup \{w_p \mid p \in \mathcal{P}_t^1\} \cup \{v_p \mid p \in \mathcal{P}_t^1, r \notin p\}.$$

Repeating this construction, i. e., finding an extended connected node set $V_t^i \supset V_t^{i-1}$ such that the set of paths $\mathcal{P}_t^i$ intersecting a node in V_t^i has smallest size and increases $\mathcal{P}_t^{i-1}$, until all nodes are considered, produces a sequence of say $j(t)+2$ path sets and path neighborhoods

$$\mathcal{P}_t^0 \subset \mathcal{P}_t^1 \subset \cdots \subset \mathcal{P}_t^{j(t)+1} = \mathcal{P} \quad \text{and} \quad W_t^0 \subset W_t^1 \subset \cdots \subset W_t^{j(t)+1},$$

with corresponding directed (r,t)-Steiner cuts

$$\delta^-(W_t^i) = \{(v_p, w_p) \in A' : p \in \mathcal{P}_t^i, r \in p\} \cup \{(w_{\tilde{p}}, v_p) \in A' : \tilde{p} \notin \mathcal{P}_t^i, p \in \mathcal{P}_t^i, r \notin p\} \quad (4.3)$$

for each $t \in T \setminus \{r\}$, $i = 0, \ldots, j(t)$. Of course, we do not have to consider all nodes for each terminal, i. e., we can choose $i < j(t)$. This can be done to reduce the time and memory consumption of the partial projection method.

A formal description of the path neighborhoods arising from the shrinking procedure is as follows. Let $x^* \in P_{LP}(\mathrm{SCP}_{\mathrm{cut}})$ be some fractional solution in the original space of variables and $\omega_e := \sum_{p\in\mathcal{P}:e\in p} x^*_p$, be edge weights for all $e \in E$. Then we consider for each node $v \in V$ the set $\mathcal{P}_v$ of paths that contain node v, and define a path neighborhood of v as

$$W^\circ_v := \begin{cases} \{w_p \mid p \in \mathcal{P}_v\} \cup \{v_p \mid p \in \mathcal{P}_v, r \notin p\} \cup \{v\}, & v \in T, \\ \{w_p \mid p \in \mathcal{P}_v\} \cup \{v_p \mid p \in \mathcal{P}_v, r \notin p\}, & v \notin T. \end{cases}$$

Sort the edges as $e_1, \dots, e_{|E|}$ such that $\omega_{e_1} \geq \cdots \geq \omega_{e_{|E|}}$. We now recursively perform two shrinking procedures. We first shrink the graph by contracting edges in the order of decreasing weight until each shrunk node contains a terminal node. Then we shrink the graph by contracting edges in the order of decreasing weight until we arrive at a single node. If nodes u and v are shrunk to a new node w, we define $\mathcal{P}_w := \mathcal{P}_u \cup \mathcal{P}_v$ and $W^\circ_w := W^\circ_u \cup W^\circ_v$. Denote by V° the set of all shrunk nodes (during both shrinking procedures) that contain a terminal node, but not the root node (the path neighborhoods of non-shrunk terminal nodes coincide with their initial path neighborhoods of the first type). Note that we can obtain shrunk nodes that do not contain a terminal node or that contain the root node, but we do not consider them in V°. Each node $v \in V^\circ$ contains a terminal t and therefore defines a directed (r,t)-Steiner cut

$$\begin{aligned} \delta^-(W^\circ_v) = &\{(v_p, w_p) \in A' : p \in \mathcal{P}_v, r \in p\} \\ &\cup \{(w_{\tilde{p}}, v_p) \in A' : \tilde{p} \notin \mathcal{P}_v, p \in \mathcal{P}_v, r \notin p\}. \end{aligned} \tag{4.4}$$

Let $\mathcal{W}' := \{W^i_t \mid t \in T \setminus \{r\},\ i = 0, \dots, j(t)\} \cup \{W^\circ_v \mid v \in V^\circ\}$ be the collection of cut sets arising from these two procedures. We propose to use the resulting two types of path neighborhoods to approximate the inequality system (4.1) by the subsystem

$$\begin{aligned} \sum_{a\in\delta^-(W),a\in A''} y_a &\geq 1 - \sum_{a\in\delta^-(W),a\in\tilde{A}} y^*_a && \forall\, W \in \mathcal{W}' \\ -\sum_{a\in\delta^-(v_p)} y_a &\geq -y^*_{v_p w_p} && \forall\, v_p \in V'\,(p \in \mathcal{P} : r \notin p) \\ y_a &\geq 0 && \forall\, a \in A'', \end{aligned} \tag{4.5}$$

i. e., instead of all directed (r,t)-Steiner cuts, we consider those arising from path neighborhoods, i. e., we replace the set $\mathcal{W}$ by $\mathcal{W}'$. We also omit the flow balance constraints for nodes w_p, $p \in \mathcal{P}$, to further reduce the size. Note that they are not important for the Steiner partition inequalities, compare with Remark 2.20. We additionally bound the variables of the dual of (4.5) by 1 in order to normalize.

The *path neighborhood cut subsystem* (4.5) is tractable because it contains a polynomial number $\sum_{t\in T}(j(t)+1) + |V| + |V| - 1 \leq \sum_{t\in T} |V| + |T| + 2|V| - 1 \in \mathcal{O}(|T| \cdot |V|)$ of cuts (each new path neighborhood of the first type reaches a new node, and there are at most $|V| + |V| - 1$ considered node sets in the shrinking procedure). It is strong in the sense that it can produce facet defining inequalities. These include Steiner partition inequalities such as the one shown in Figure 2.4. In general, a Steiner partition inequality

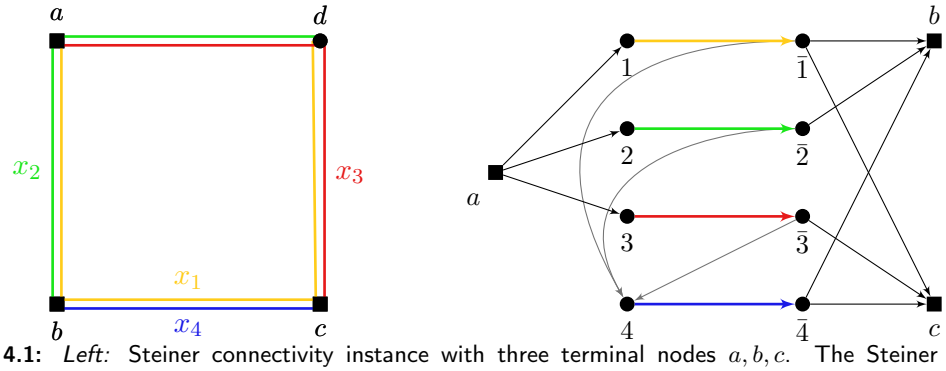


Figure 4.1: *Left:* Steiner connectivity instance with three terminal nodes a, b, c. The Steiner partition inequality $2x_1 + x_2 + x_3 + x_4 \geq 2$ is valid. *Right:* Corresponding Steiner connectivity digraph for $r = a$.

will be separated with our path neighborhood cut subsystem, if each node set of the partition satisfies the following condition: The paths that intersect the node set of a partition correspond to a path neighborhood. This is, e. g., the case for $T = V$ and the Steiner partition inequality corresponding to the partition $V_t = \{t\}$, $t \in T$. The following example shows a different case where the path neighborhood cut subsystem implies a Steiner partition inequality. We then give an example to show that the path neighborhood cut subsystem implies also other types of facet defining inequalities.

Example 4.1. *We show that the Steiner partition inequality shown in Figure 2.4 can also be separated with our path-neighborhood cut subsystem as follows. To this purpose we illustrated the instance and the Steiner connectivity digraph for root node $r = a$ in Figure 4.1. Let $x^* \in P_{LP}(\mathrm{SCP}_{\mathrm{cut}})$ with $x_2^* = x_3^* = x_4^* = 0.5$ and $x_1^* = 0$. Obviously x^* satisfies all Steiner cut inequalities but not $2x_1 + x_2 + x_3 + x_4 \geq 2$. Consider the path-neighborhood cut subsystem (4.5) for the two initial path-neighborhoods $W_b^0 := \{\bar{1}, \bar{2}, 4, \bar{4}, b\}$ and $W_c^0 := \{\bar{1}, \bar{3}, 4, \bar{4}, c\}$*

$$\begin{aligned} y_{\bar{3}4} &\geq 1 - y^*_{1\bar{1}} - y^*_{2\bar{2}} && (W_b^0) && (\mu_b)\\ y_{\bar{2}4} &\geq 1 - y^*_{1\bar{1}} - y^*_{3\bar{3}} && (W_c^0) && (\mu_c)\\ -y_{\bar{1}4} - y_{\bar{2}4} - y_{\bar{3}4} &\geq -y^*_{4\bar{4}} && && (\pi_4) \end{aligned}$$

(all other path-neighborhood cuts are redundant). The dual is

$$\begin{aligned} (1 - y^*_{1\bar{1}} - y^*_{2\bar{2}})\mu_b + (1 - y^*_{1\bar{1}} - y^*_{3\bar{3}})\mu_c - \pi_4 \cdot y^*_{4\bar{4}} &> 0\\ \mu_b - \pi_4 &\leq 0\\ \mu_c - \pi_4 &\leq 0\\ \mu_b, \mu_c &\geq 0\\ \pi_4 &\geq 0. \end{aligned}$$

A valid solution is $\mu_b = \mu_c = 1$, $\pi_4 = 1$. This yields the cutting plane

$$(1 - x_1 - x_2) + (1 - x_1 - x_3) - x_4 \leq 0 \quad \Leftrightarrow \quad 2x_1 + x_2 + x_3 + x_4 \geq 2.$$

It is a facet defining Steiner partition inequality.

Example 4.2. *Consider again the SCP instance in Figure 2.7 that gives an example of a three terminal sets inequality which is not a partition inequality (compare with Subsection 2.2.3). We show that this inequality can be separated with our path-neighborhood cut subsystem as follows. Let* $x^* \in P_{LP}(\mathrm{SCP}_{\mathrm{cut}})$ *with* $x^*_1 = x^*_2 = x^*_3 = 0.5$ *and* $x^*_4 = 0$. *Obviously* x^* *satisfies all Steiner cut inequalities but not* $x_1 + x_2 + x_3 + x_4 \geq 2$. *Consider the path-neighborhood cut subsystem (4.5) for the two initial path-neighborhoods* $W^0_b := \{\bar{1}, 2, \bar{2}, \bar{4}, b\}$ *and* $W^0_c := \{2, \bar{2}, \bar{3}, c\}$

$$\begin{array}{rrrll} & y_{\bar{3}2} & \geq 1 - y^*_{1\bar{1}} - y^*_{4\bar{4}} & (W^0_b) & (\mu_b) \\ y_{\bar{1}2} & +y_{\bar{4}2} & \geq 1 - y^*_{3\bar{3}} & (W^0_c) & (\mu_c) \\ -y_{\bar{1}2} & -y_{\bar{3}2} - y_{\bar{4}2} & \geq -y^*_{2\bar{2}} & & (\pi_2) \end{array}$$

(all other path-neighborhood cuts are redundant). The dual is

$$\begin{aligned} (1 - y^*_{1\bar{1}} - y^*_{4\bar{4}})\mu_b + (1 - y^*_{3\bar{3}})\mu_c - \pi_2 \cdot y^*_{2\bar{2}} > 0 \\ \mu_b - \pi_2 \leq 0 \\ \mu_c - \pi_2 \leq 0 \\ \mu_b, \mu_c \geq 0 \\ \pi_2 \geq 0. \end{aligned}$$

A valid solution is $\mu_b = \mu_c = 1$, $\pi_2 = 1$. *Since* $y^*_{1\bar{1}} = x^*_1 = 0.5$, $y^*_{2\bar{2}} = x^*_2 = 0.5$, $y^*_{3\bar{3}} = x^*_3 = 0.5$, *and* $y^*_{4\bar{4}} = x^*_4 = 0$, *the value of the first inequality in this system is* 0.5. *This yields the cutting plane*

$$(1 - x_1 - x_4) + (1 - x_3) - x_2 \leq 0 \quad \Leftrightarrow \quad x_1 + x_2 + x_3 + x_4 \geq 2.$$

It is a facet defining three terminal sets inequality, see again Subsection 2.2.3.

4.2 Primal Heuristics

We developed two types of primal heuristics for the Steiner connectivity problem. The first one is a greedy type heuristic based on LP values (and on objective coefficients in an initial step). In the initial step we sort the paths in decreasing order of their coefficients in the objective function. We then repeatedly choose the path with the highest coefficient and fix it to zero if the network is still connected. Otherwise, the path is fixed to one. The procedure terminates when all paths are fixed. If we have LP values at hand, we sort the paths in increasing order of their LP value. We then repeatedly choose the path with the smallest LP value and proceed as in the initial step.

The second primal heuristic is based on the idea of Takahashi and Matsuyama [99] for the Steiner tree problem, compare also with Subsection 1.4.2. We briefly recall the idea that is given in Algorithm 1.4. The starting cost of all paths in this heuristic is set to $(1 - x^*_p) \cdot c_p$ for $p \in \mathcal{P}$, where x^* is the optimal solution for the current LP. We start from one terminal and connect the next terminal that can be reached by a cost-minimal

connecting set. We then set the cost of all used paths to zero and search for a cost-minimal connecting set to the next closest terminal. This process is continued until all terminals are connected.

The result of the second heuristic depends on the choice of the starting terminal. It is usually too time consuming to consider all terminals. We therefore use a similar strategy as Koch and Martin [66] for the Steiner tree problem: We try the terminal that gave the best solution the last time the heuristic was started and in addition 5 randomly selected terminals.

4.3 Computational Analysis

We will now show that the Steiner partition inequalities indeed significantly improve the LP relaxation of the canonical undirected cut formulation and help to solve Steiner connectivity problems. We have implemented the methods described in the preceding subsections and tested them on six transportation networks that we denote as China, Dutch, SiouxFalls, Anaheim, Potsdam, and Chicago. Instances Anaheim, SiouxFalls, and Chicago use the graphs of the street networks with the same names from the Transportation Network Test Problems Library of Bar-Gera [98]. Instances China, Dutch, and Potsdam correspond to public transportation networks. The Dutch network was introduced by Bussieck [28] in the context of line planning. The Potsdam data were provided to us in a joint project on line planning by the local public transport company ViP Verkehrsgesellschaft Potsdam GmbH. The China instance is artificial; we constructed it as a showcase example, connecting the twenty biggest cities in China by the 2009 high speed train network. All data but the one of Anaheim includes coordinates of the points of the network. A visualization of the networks can be seen in Figure 4.2. Along with the transportation networks we also considered the two "textbook-examples" given in Figures 2.7 and 2.9 of Chapter 2.2 which we denote by "Tiny1" and "Tiny2".

All instances are associated with an OD matrix. We define as terminal nodes all stations with positive supply or demand, i. e., such that there exists a positive entry in the corresponding row or column of the OD matrix. The paths can then be interpreted as possible lines (e. g., bus lines in the street networks) to connect the terminal nodes/OD nodes. In the Potsdam instance we distinguish between edges of different types, e. g., edges for tram lines and edges for bus lines. Solving the Steiner connectivity problems with costs depending on the lengths of the lines amounts to the construction of a connected line plan with minimum cost (where each line is operated once). Such a solution can be used to estimate a lower bound on the cost of a line plan.

For each network, we consider two benchmark instances of the Steiner connectivity problem. These were constructed as follows. For each network we randomly chose a set of node pairs and computed the shortest path between each pair (instances with suffix 1) and the three shortest paths between each pair (instances with suffix 2). In the Potsdam instances the edges of such a path have to be of the same type (e. g., bus, tram). For the

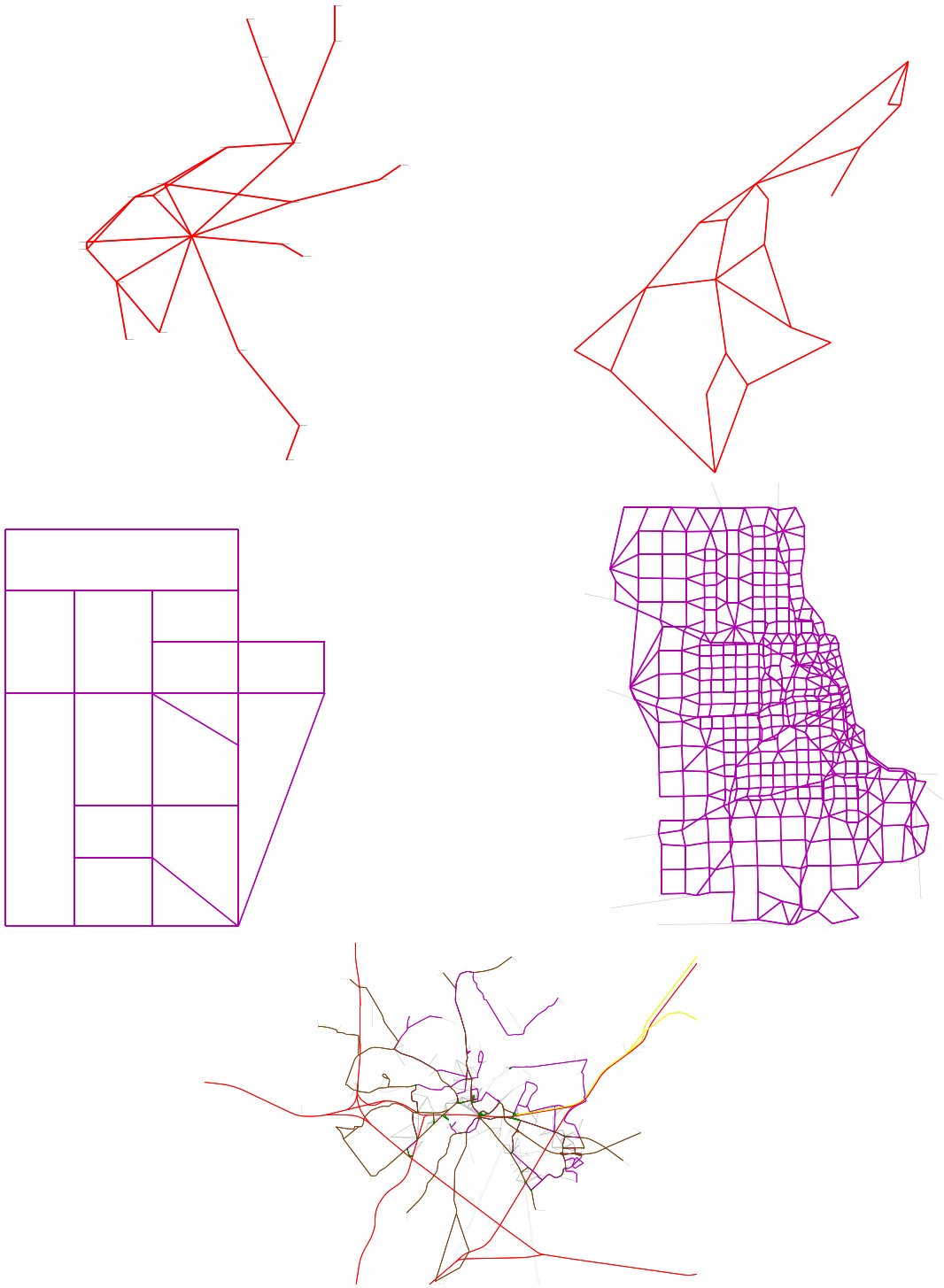


Figure 4.2: Network of instances Dutch (*upper left*), China (*upper right*), Sioux Falls (*middle left*), Chicago (*middle right*), and Potsdam.

Table 4.1: Street and public transportation networks. The columns are as follows: name of the instance, number of terminal nodes, total number of nodes, number of (directed) edges (including OD edges), number of paths, and number of nodes and arcs of the associated Steiner connectivity digraph. (We inserted all terminals twice into the Steiner connectivity digraph in order to use them as sources and sinks at the same time; this speeds up the computations.) The last column gives the maximal length of a path.

name	$\lvert T\rvert$	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert\mathcal{P}\rvert$	$\lvert V'\rvert$	$\lvert A'\rvert$	$\max \lvert p\rvert$
Tiny1	3	4	4	4	14	15	2
Tiny2	4	7	10	7	22	36	2
China1	20	20	98	130	300	9 621	6
China2	20	20	98	211	462	29 225	6
Dutch1	23	23	106	173	392	18 582	6
Dutch2	23	23	106	263	572	53 449	6
SiouxFalls1	24	24	124	186	420	14 650	6
SiouxFalls2	24	24	124	311	670	50 252	6
Anaheim1	38	454	1 344	1 713	3 502	777 466	20
Anaheim2	38	454	1 344	5 135	10 346	7 346 857	20
Potsdam1	107	885	3 572	2 401	5 016	949 440	20
Potsdam2	107	885	3 572	5 349	10 912	5 747 714	20
Chicago1	386	909	3 672	2 546	5 864	1 419 999	20
Chicago2	386	909	3 672	7 638	16 048	12 858 604	20

three smallest instances (China, Dutch, and SiouxFalls) we restricted the lengths of the paths to 6 edges. For the three largest instances (Anaheim, Potsdam, and Chicago) we considered paths with at most 20 edges. These restrictions were chosen in order to avoid that only very few paths connect the whole network; very long lines are also not desired in public transport. The costs of the paths correspond to the lengths of the paths in kilometers, which is given by the lengths of the edges in the network data. All instances were reduced by some preprocessing, see Section 8.1 in Chapter 8.

Table 4.1 gives some statistics on the instances. It shows the number of nodes, edges, and arcs for the networks and the associated Steiner connectivity digraphs as well as the number of paths for all instances. One can see that the number of arcs of the Steiner connectivity digraph, which is the number of variables in the strengthened directed cut formulation ($\mathrm{SCP}^r_{\mathrm{arc}^+}$), is nearly quadratic in the number of paths $\mathcal{P}$.

Table 4.2 presents the performance of the undirected cut formulation including SPI separation in comparison to the undirected cut formulation, the strengthened directed cut formulation, and the undirected cut formulation extended by partial projection. More precisely, the table shows the LP value and the computation time in CPU seconds for solving the LP relaxation of

- the weak cut formulation ($\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}}$),
- the cut formulation ($\mathrm{SCP}_{\mathrm{cut}}$),
- the strengthened directed cut formulation ($\mathrm{SCP}^r_{\mathrm{arc}^+}$),
- the weak cut formulation improved by the SPI separation method ($\mathrm{SCP}^{\mathrm{w,SPI}}_{\mathrm{cut}}$), and
- the weak cut formulation improved by the partial projection method ($\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}^+}$).

A '*' in the time column indicates that the time limit of five hours was reached. A '–' indicates that this formulation exceeds the memory limit of the used computer. All computations were done with version 1.2.0 of SCIP [2, 95] on an Intel Quad-Core 2, 3.0 GHz computer (in 64 bit mode) with 6 MB cache, running Linux and 16 GB of memory. By default, we use the simplex method of Cplex 12.1 [61] for solving LPs (in single core mode).

We initialize all formulations with the following cut

$$\sum_{p \in \mathcal{P}} a_p x_p \geq |T| - 1, \quad a_p = \min\{|\{t \in T : t \in p\}|, |p|\},$$

which can be interpreted as computing the minimum number of lines to connect all terminal nodes. Here, each line is weighted by the minimum of its length and the number of terminal nodes it contains. Note, if all nodes are terminal nodes, each path is weighted by its lengths which is the number of terminal nodes minus one. In this case the above inequality corresponds to the Steiner partition inequality where each node (=terminal node) forms a single partition set. Then a cutting plane algorithm depending on the formulation was run until no improvement could be made or the time limit was exceeded. For the directed cut formulation we inserted all terminals twice into the Steiner connectivity digraph in order to use them as sources and sinks at the same time, i. e., we used back cuts (compare with Koch and Martin [66]) from terminal to root node which speeds up the computations. The partial projection method was stopped if we did not find a cut or if the LP value did not change for ten rounds of cuts by more than 1% (formulation ($\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut+}}$)). For the bigger instances Anaheim, Potsdam, and Chicago, we started the first partial projection separation round after the objective value could not be improved by at least 1.5% in one Steiner partition separation round. This is done to get a better LP value for initializing the path neighborhoods of the shrinking procedure, which is not necessary for the smaller instances. To keep the path neighborhood cut subsystem within a tractable size, we considered $\mathcal{W}' := \{W_t^i \mid t \in T \setminus \{r\},\ 0 \leq i \leq \max\{3, \lceil \frac{|V|}{|T|} \rceil\}$ for the path neighborhoods of the terminals and stopped the procedure when $|\delta^-(W_t^i)| \geq 100\,000$. For the two big Chicago instances, we stopped the shrinking procedure for the path neighborhoods after the first shrinking step, i. e., after all nodes are contained in a partition set with at least one terminal node, see Section 4.1.3.

Let us now analyze the results in Table 4.2. The advantage of the weak cut formulation ($\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}}$) is its compactness. This formulation has the smallest number of variables and inequalities. Moreover, the separation problem for the weak Steiner path cut constraints can be solved in the original undirected graph. The weak cut formulation ($\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}}$), therefore, has the shortest computation times. Considering the cut formulation ($\mathrm{SCP}_{\mathrm{cut}}$), we only get a small increase of the LP value for few instances compared to the weak cut formulation ($\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}}$), whereas the computation time of ($\mathrm{SCP}_{\mathrm{cut}}$) always takes much longer, since the separation problem requires the construction of the Steiner connectivity digraph. However, the Steiner connectivity digraph can be used to improve the LP bound significantly via the strengthened cut formulation ($\mathrm{SCP}^{r}_{\mathrm{arc+}}$). The weaknesses of

Table 4.2: LP values and computation times for solving four formulations of the Steiner connectivity problem. A '*' indicates that the time limit of five hours was reached. A '–' indicates that the memory limit was reached. For $(\mathrm{SCP}^{r}_{\mathrm{arc+}})$ we compared the running times for the barrier and the simplex to solve the LPs. We took the best value of both computations. For instance Potsdam1 the barrier (b) computed the best value. All other models were solved using the simplex method.

	$(\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}})$		$(\mathrm{SCP}_{\mathrm{cut}})$		$(\mathrm{SCP}^{r}_{\mathrm{arc+}})$		$(\mathrm{SCP}^{\mathrm{w,SPI}}_{\mathrm{cut}})$		$(\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut+}})$	
name	value	time	value	time	value	time	value	time	value	time
Tiny1	2.5	< 1	2.5	< 1	3.0	< 1	3.0	< 1	2.5	< 1
Tiny2	4.0	< 1	4.0	< 1	6.0	< 1	4.7	< 1	5.0	< 1
China1	7906.1	< 1	7906.1	< 1	8382.0	31	8382.0	< 1	8329.5	< 1
China2	7892.8	< 1	7892.8	< 1	8089.0	5	8089.0	< 1	8089.0	< 1
Dutch1	19250.0	< 1	19250.0	< 1	19755.7	*	19750.0	< 1	19800.0	< 1
Dutch2	19100.0	< 1	19100.0	< 1	19762.9	*	19750.0	< 1	19800.0	< 1
SiouxFalls1	104.3	< 1	104.3	< 1	114.0	301	114.0	< 1	114.0	1
SiouxFalls2	105.1	< 1	105.1	2	112.0	365	112.0	< 1	112.0	2
Anaheim1	960961.1	2	961690.0	55	964046.3	*	966225.6	4	966834.0	95
Anaheim2	813194.1	45	815267.0	1452	–	–	829451.0	76	820391.2	597
Potsdam1	260886.9	135	261306.8	14132	(b) 261459.3	*	261196.7	194	261846.3	412
Potsdam2	256106.4	1 242	243079.5	*	250504.1	*	256200.2	799	256473.9	1 386
Chicago1	2642.8	378	2644.6	2777	2660.6	*	2642.9	778	2673.1	8 805
Chicago2	2198.3	652	2199.9	*	–	–	2329.2	*	2298.3	*

this model are the long computation times and the memory consumption, since it uses the arcs of the Steiner connectivity digraph as variables. Note that the size of the Steiner connectivity digraph depends on the number and length of the paths in the original graph, which can become very large. For this reason, the strongest formulation, the strengthened directed cut formulation ($SCP^{r}_{arc^{+}}$), becomes practically intractable for large problems. Its LP value could not be computed for Anaheim2 and for Chicago2, because of the excessive memory consumption. Model ($SCP^{w}_{cut^{+}}$) combines the compactness of the weak cut formulation with the quality of the strengthened directed cut formulation. The results show that ($SCP^{w}_{cut^{+}}$) indeed approximates the strengthened directed cut formulation very well. Its memory consumption can be controlled via the definition of the considered cuts. Its LP value can therefore be computed for all instances. Considering the LP value of model ($SCP^{w,SPI}_{cut}$), it becomes apparent that most of the strength of the directed cut formulation can also be achieved by separating Steiner partition inequalities as described in Section 4.1.1. For seven instances the LP bounds of model ($SCP^{w,SPI}_{cut}$) are as good as the LP bounds of ($SCP^{r}_{arc^{+}}$) or ($SCP^{w}_{cut^{+}}$); for two instances ($SCP^{w,SPI}_{cut}$) even yields the best LP bounds that can be computed within the given time limit. The results imply that the Steiner partition inequalities are the key to improve the undirected cut formulation for most Steiner connectivity instances. Model ($SCP^{w,SPI}_{cut}$) provides an easy and time-saving way to utilize them computationally, models ($SCP^{r}_{arc^{+}}$) and ($SCP^{w}_{cut^{+}}$) provide quality certificates in terms of strong lower bounds.

We also used a branch-and-cut method to solve the Steiner connectivity problem for the six test instances (without the "Tiny-examples"). We used the default heuristics of SCIP, the strong branching rule, and the two heuristics described in Subsection 4.2. The SCIP separators were turned off, no presolving, nor propagating, the minimal efficiency for a cut to be included was reduced to 0.001 for the root node. We only made 1 separation round per node. We limited the computation time for each instance to 10 hours. The results for the models (SCP^{w}_{cut}), ($SCP^{w}_{cut^{+}}$), and ($SCP^{w,SPI}_{cut}$) are shown in Tables 4.3, 4.4, and 4.5. These results slightly differ from the results given in our earlier Paper [20]; the results in Tables 4.3, 4.4, and 4.5 include the second heuristic, the results given in the Paper [20] do not include the second heuristic. This heuristic needs much time. We, therefore, did not compute as many nodes in ten hours as in the results in [20]. However, the primal solution especially for the Potsdam instances can be improved by around 1% to 1.6%.

The instances China, Dutch, SiouxFalls, and Anaheim1 can be solved by all three formulations. ($SCP^{w,SPI}_{cut}$) and ($SCP^{w}_{cut^{+}}$) need much less nodes than (SCP^{w}_{cut}). Anaheim2 can also be solved by ($SCP^{w,SPI}_{cut}$) and ($SCP^{w}_{cut^{+}}$) but not by (SCP^{w}_{cut}) within the given time limit. ($SCP^{w,SPI}_{cut}$) performs best on Anaheim2, it solves the problem in the root node within seconds. Model ($SCP^{w,SPI}_{cut}$) finds the best solutions for nearly all instances except Potsdam2, here the partial projection method finds the best solution. The gap for all instances can be reduced by SPI separation, for Potsdam1, Potsdam2, and Chicago1 the gap is below 5%. This shows that indeed large scale Steiner connectivity problems can be solved to proven optimality or near optimality.

Table 4.3: Solving ($\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}}$) with branch-and-cut using SCIP. The time is given in seconds; a '*' indicates that the computation time limit of 10 hours was reached. Instances solved to optimality are in bold characters. We highlighted the best values for primal solution and dual bound by a green color, for instances solved to optimality, we highlighted the shortest computation time.

name	bb nodes	time	dual bound	primal solution	gap
China1	47	<1	8382.0	8382	0.00%
China2	23	<1	8089.0	8089	0.00%
Dutch1	31	<1	198.0	198	0.00%
Dutch2	19	<1	198.0	198	0.00%
SiouxFalls1	3 068	33	114.0	114	0.00%
SiouxFalls2	515	10	112.0	112	0.00%
Anaheim1	491	148	976603.0	976603	0.00%
Anaheim2	37 038	*	826098.5	836167	1.22%
Potsdam1	12 587	*	261912.0	269156	2.77%
Potsdam2	2 543	*	256862.3	266051	3.58%
Chicago1	671	*	2689.0	2806	4.35%
Chicago2	164	*	2214.2	2778	25.47%

Table 4.4: Solving ($\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}^+}$) with branch-and-cut using SCIP.

name	bb nodes	time	dual bound	primal solution	gap
China1	1	<1	8382.0	8382	0.00%
China2	1	<1	8089.0	8089	0.00%
Dutch1	3	<1	198.0	198	0.00%
Dutch2	6	<1	198.0	198	0.00%
SiouxFalls1	1	2	114.0	114	0.00%
SiouxFalls2	1	2	112.0	112	0.00%
Anaheim1	150	293	976603.0	976603	0.00%
Anaheim2	3 703	13 110	831749.0	831749	0.00%
Potsdam1	9 619	*	2626091.1	268784	2.35%
Potsdam2	2 175	*	257157.9	263877	2.61%
Chicago1	268	*	2703.1	2939	8.73%
Chicago2	5	*	2303.7	2980	29.35%

Table 4.5: Solving ($\mathrm{SCP}^{\mathrm{w,SPI}}_{\mathrm{cut}}$) with branch-and-cut using SCIP.

name	bb nodes	time	dual bound	primal solution	gap
China1	11	<1	8382.0	8382	0.00%
China2	1	<1	8089.0	8089	0.00%
Dutch1	33	<1	198.0	198	0.00%
Dutch2	1	<1	198.0	198	0.00%
SiouxFalls1	1	<1	114.0	114	0.00%
SiouxFalls2	1	<1	112.0	112	0.00%
Anaheim1	121	42	976603.0	976603	0.00%
Anaheim2	1	170	831749.0	831749	0.00%
Potsdam1	12 225	*	262186.3	267881	2.17%
Potsdam2	2 556	*	256918.5	265114	3.19%
Chicago1	611	*	2685.8	2800	4.25%
Chicago2	68	*	2332.5	2772	18.84%

Chapter 5

Concluding Remarks for Part I

In the first part of this thesis we introduced the Steiner connectivity problem that generalizes the Steiner tree problem to a path-connectivity setting. We investigated its complexity, approximation results, integer programming formulations, and polyhedral aspects. It turned out that the Steiner connectivity problem has strong relations to the Steiner tree problem, the (submodular) set covering problem, and that it can be interpreted in the context of hypergraphs. The relation to Steiner tree problems yields polynomially solvable cases if the number of terminal nodes is fixed. Moreover, the 2-approximation algorithm of Goemans and Williamson for Steiner tree and Steiner forest problems can be generalized to the Steiner connectivity problem. We showed that this yields a $k+1$-approximation if all paths contain at most k edges or at most k terminal nodes. The relation to the set covering problem yields a $\log k$ approximation if all nodes are terminal nodes. The two terminal case gives rise to a TDI description; this yields a combinatorial companion theorem to Menger's theorem for hypergraphs and characterizes paths and cuts in hypergraphs as well as connecting and disconnecting sets as a blocking pair. Table 5.1 summarizes similarities and differences between the Steiner tree problem and the Steiner connectivity problem.

We further discussed the relative strengths of different formulations of the Steiner connectivity problem, namely, the undirected and the directed cut formulation in several variants. If we take the corresponding LP relaxations as a measure of strength, we obtain the following picture:

$$P_{LP}(\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}}) \supseteq P_{LP}(\mathrm{SCP}_{\mathrm{cut}}) = P_{LP}(\mathrm{SCP}_{\mathrm{arc}})|_{\mathcal{p}} \supseteq \begin{matrix} P_{LP}(\mathrm{SCP}^{\mathrm{SPI}}_{\mathrm{cut}}) \\ P_{LP}(\mathrm{SCP}_{\mathrm{cut}^+}) \end{matrix} \supseteq P_{LP}(\mathrm{SCP}^{r}_{\mathrm{arc}^+})|_{\mathcal{p}},$$

$$P_{LP}(\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}}) \supseteq \begin{matrix} P_{LP}(\mathrm{SCP}^{\mathrm{w,SPI}}_{\mathrm{cut}}) \\ P_{LP}(\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}^+}) \end{matrix} \supseteq P_{LP}(\mathrm{SCP}^{r}_{\mathrm{arc}^+})|_{\mathcal{p}}.$$

Here, $(\mathrm{SCP}^{\mathrm{SPI}}_{\mathrm{cut}})$ and $(\mathrm{SCP}_{\mathrm{cut}^+})$ denote the cut formulation including SPI separation and partial projection, respectively. As $(\mathrm{SCP}_{\mathrm{cut}})$ is so hard to compute, we resorted to the

Table 5.1: Comparing complexity and polyhedral results for the Steiner tree problem and the Steiner connectivity problem.

	Steiner tree problem	Steiner connectivity problem
general case	$\mathcal{NP}$-hard	
$\lvert T\rvert = k$	polynomial	
$\lvert T\rvert = 2$	Menger's theorem and Menger's companion theorem	
$T = V$	polynomial minimum spanning tree	$\mathcal{NP}$-hard Greedy: $H(k)$-approx. k max. number of edges in paths
primal-dual alg.	2-approximative	$(k+1)$-approximative k minimum of (a) max. #edges/path, (b) max. #terminals/path
polyhedral results	directed formulation dominates undirected formulation (has to include flow balance constr.) Steiner partition ineq. implied by directed formulation	

associated weak versions $P_{LP}(\mathrm{SCP}^{\mathrm{w,SPI}}_{\mathrm{cut}})$ and $P_{LP}(\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}^+})$, see the second line of inclusions. SPI separation and partial projection lead to incomparable formulations: The latter might produce only part of the or different SPIs, but some additional inequalities not covered by the former. The tightest formulation is produced by the extended formulation $P_{LP}(\mathrm{SCP}^{r}_{\mathrm{arc}^+})|_{\mathcal{P}}$.

In practice, however, the strongest formulation does not necessarily produce the best results, because there is a tradeoff between the strength of a dual bound and the time needed to compute it. The ranking depends on the time limit and whether we compute an LP bound or an IP value. If the goal is to produce the best bound (including branching) a good choice is formulation $P_{LP}(\mathrm{SCP}^{\mathrm{w,SPI}}_{\mathrm{cut}})$. This is somewhat unexpected: Neither the tightest formulation $P_{LP}(\mathrm{SCP}^{r}_{\mathrm{arc}^+})|_{\mathcal{P}}$ nor the related formulation $P_{LP}(\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}^+})$ dominate the other formulations – although $P_{LP}(\mathrm{SCP}^{\mathrm{w}}_{\mathrm{cut}^+})$ is close to $P_{LP}(\mathrm{SCP}^{\mathrm{w,SPI}}_{\mathrm{cut}})$. In a sense, this result confirms the general experience that extended formulations have strong theoretical properties, but they do not necessarily provide formulations well suited for practical computations – more research is needed to understand this behavior.

In the context of line planning, the Steiner connectivity problem corresponds to computing a cost minimal line plan for a given line pool, ignoring capacities. The considered cut inequalities and Steiner partition inequalities are also valid for line planning. We will consider capacitated versions of such inequalities to improve the LP relaxation of the line planning models in Chapter 7.

Part II

The Line Planning Problem

Chapter 6

Line Planning Models

In this chapter, we investigate the line planning problem, a fundamental problem in the design of a public transportation system. It consists of finding a set of lines in an infrastructure network and their frequencies of operation (line planning) such that a given travel demand can be routed (passenger routing). There are two main objectives, namely, minimization of operation costs (the operator's point of view) and minimization of travel and transfer times (the passengers' point of view). A great challenge is the integration of line planning and passenger routing; it discloses essential degrees of freedom. Another major challenge is the treatment of transfers. Existing models either relax one of these two important aspects or they are of extremely large scale, and seem to be computationally intractable.

We propose a novel direct connection approach that allows an integrated optimization of line planning and passenger routing, including accurate estimates of the number of direct travelers, for large-scale real-world instances. To analyze the improvement over existing models that integrate line planning and passenger routing, we also introduce the approaches of Borndörfer, Grötschel, and Pfetsch [15] and of Schöbel and Scholl [93]. Borndörfer, Grötschel, and Pfetsch propose a multi commodity flow model that generates passenger and line paths dynamically but neglects a large number of transfers. Schöbel and Scholl use a *change-and-go* graph to achieve a detailed treatment of transfers. The drawback of this model is its enormous size. The new approach combines the advantages of the existing models to arrive at a model of manageable size including a transfer treatment. Indeed, a comparison of all models in Chapter 8 shows that the direct connection approach is currently the only computationally tractable integrated line planning and passenger routing method that provides good estimates of transfer times. Some parts of this chapter are published in [17, 23].

The chapter is structured as follows. We give an overview of the literature dealing with the line planning problem in Section 6.1. The basic notations and definitions for the line planning problem are stated in Section 6.2. We discuss the approaches of Borndörfer, Grötschel, and Pfetsch [15] and of Schöbel and Scholl [93] in Sections 6.3 and 6.4, respec-

tively. Our new direct connection approach is introduced in Section 6.5. We end this chapter by comparing all introduced models in Section 6.6.

6.1 Literature Overview

Line planning has been investigated extensively in the operations research literature over the last forty years. Topics range from heuristic approaches over integer programming models with fixed passenger routes to the simultaneous optimization of line planning and passenger routing. The task of all approaches is to find a set of lines (paths in a given infrastructure network) with frequencies (how often the line is operated in a given time horizon) such that a given travel demand (usually a point-to-point demand in the network) can be satisfied. In the following we will overview the main developments to approach the line planning problem with integer programming methods. We will start by summarizing the main results concerning the complexity of this problem and will then discuss models and solution methods. Here, we will distinguish between approaches that assume a fixed passenger routing, i. e., the number of passengers that travel over each edge in the infrastructure is known in advance, and approaches that integrate a passenger routing, i. e., line planning and passenger routing is done simultaneously. A comprehensive survey article on methods and models to deal with the line planning problem is given by Schöbel [92]. The article also discusses further approaches not considered here and covers all of the relevant literature up to the year 2010.

Nearly all integer programming approaches for line planning are based on a so-called *line pool*, i. e., in a "pre-phase" a set of possible lines is generated, e. g., governed by given rules and length restrictions, and afterwards the optimization model chooses a subset of lines out of the line pool. An exception to such a two stage approach is proposed by Borndörfer, Grötschel, and Pfetsch [15, 16], see also below, who integrated the generation of lines in the optimization model. If not stated otherwise all approaches discussed in the following assume a predefined line pool. For the computation of a line pool see, e. g., Ceder and Wilson [34] who describe a method to enumerate all lines satisfying a length restriction with respect to a shortest path.

Complexity. The line planning problem is $\mathcal{NP}$-hard in nearly all cases. Bussieck [29] shows that the *feasibility problem* for line planning with fixed passenger routing is $\mathcal{NP}$-hard. The feasibility problem is to find a set of lines (out of a given set) such that given lower and upper bounds on the sum of the frequencies for every edge in the infrastructure network are satisfied. The lower and upper bounds on the sum of the frequencies can be interpreted as a minimum operation frequency to transport all passengers and a maximum operation frequency to guarantee a safe service. Claessens, van Dijk, and Zwaneveld [37] show that the *cost optimization problem* for line planning with fixed passenger routing is $\mathcal{NP}$-hard, i. e., finding a line plan with minimum costs that satisfies given lower bounds on the sum of the frequencies for all edges in the network.

Schöbel and Scholl [93] show that line planning with integrated passenger routing is also

$\mathcal{NP}$-hard. More precisely, they show that finding a set of lines fulfilling a given budget constraint and allowing a passenger routing with a minimal number of transfers is $\mathcal{NP}$-hard. Their proof can be easily extended to show that the line planning problem with integrated passenger routing is already $\mathcal{NP}$-hard for one OD pair. Torres et al. [102] analyze the complexity of line planning problems for special network topologies, namely, paths and trees, and different degrees of freedom in line construction, e. g., lines skipping some stations on their route, so-called express lines, and uni-directional lines. Their analysis is motivated by the Trolebus system of Quito in Ecuador which consists of a trunk route and a number of feeder bus systems. They show that nearly all combinations of topology, line construction, and line cost structure lead to $\mathcal{NP}$-hard problems. The only polynomially solvable case is on a path-graph with no express lines, no fixed costs for lines, and no directed lines.

Fixed Passenger Routing. The first branch and bound approach for line planning problems was developed by Bouma and Oltrogge [27]. They investigated line planning problems of Nederlandse Spoorwegen, which operates three types of railway transportation systems, two types for regional traffic and one for intercity traffic, which mainly differ in their speed. Each transportation system is considered individually; they propose a method, the so-called *system split*, to compute an a priory distribution of passengers to the different transportation systems. The idea is that passengers change to fast trains as early as possible and to slow trains as late as possible. Taking the additional assumption that passengers travel on shortest paths, one can fix the passenger flow on each edge in the network, independent of the line plan. A subsequent second planning step then uses integer programming techniques to cover this fixed passenger flow by lines. Bouma and Oltrogge's system split approach was taken up and improved by several research groups, who all provide computational analyses for a benchmark data set of the Dutch regional and intercity network. Bussieck, Kreuzer, and Zimmermann [30] maximize the passenger convenience by maximizing the number of direct travelers. They start with a model in which direct travelers of one demand relation are associated with a specific line. This leads to a large scale model which is hard to compute. They propose a relaxation by aggregating the direct travelers on all usable lines which yields an upper bound on the number of direct travelers. This relaxation can be solved quite efficiently. Computing the "real" number of direct travelers for the frequencies of the lines as in the solution of the relaxed model, i. e., setting the lines in the direct traveler model to the frequencies computed by the relaxed model, yields a lower bound on the number of direct travelers. They show that the gaps between these two bounds are small ($< 3.1\%$). A disadvantage of this model is that solutions often result in long routes for the lines. The line operating cost usually depends on the lengths of the lines. This motivated Claessens, van Dijk, and Zwaneveld [37] to propose a model that minimizes the operating costs of a line plan, i. e., the costs of the personnel and the rolling stock. They derive a model with a nonlinear objective function and nonlinear constraints, propose a linearization, and solve it using a branch and bound algorithm including preprocessing techniques and the generation of lower bounds. In their computations they compare an optimal solution of the cost minimization approach with an optimal solution of the direct travelers

approach. The costs can be reduced by around 17% while the number of direct travelers decreases by 6%. Bussieck [29] discusses both approaches, the direct traveler and the cost minimization approach, in detail; he further presents complexity results, polyhedral investigations, algorithmic aspects, and detailed computations. Bussieck, Lindner, and Lübbecke [31] reconsider the nonlinear cost minimization model of Claessens, van Dijk, and Zwaneveld [37]. They study different linearizations of the model and propose a variable fixing method to obtain good primal solutions. More precisely, they use the nonlinear model to determine "promising lines" and solve the integer linear model only for this line set. In this way, they obtain a good primal solution after five minutes which is then used as an upper bound in a branch-and-bound algorithm for a linearized model. Goossens, van Hoesel, and Kroon [55] develop a sophisticated branch-and-cut algorithm for the linearized cost minimization model of Claessens, van Dijk, and Zwaneveld. They propose several preprocessing methods, cutting planes, and branching rules. In a computational analysis they compare their algorithm to the integer programming solver Cplex 6.6.1 (which was at that time the state of the art) and conclude that their algorithm performs significantly better.

The drawback of a fixed passenger routing is that it ignores that the passenger flow strongly depends on the line plan which is to be computed. Hence, approaches that consider an integrated line planning and passenger routing become more and more popular.

Integrated Passenger Routing. Schöbel and Scholl [93] and Scholl [94] propose a model to compute a line plan that minimizes the travel times including a penalty for each transfer. To this end, they construct a so-called *change-and-go* graph. This graph contains multiple copies of infrastructure edges for all passing lines and transfer edges for all possible transfers from one line to another. They propose a Dantzig-Wolfe decomposition to solve the LP relaxation of the change-and-go model. Recently, Schmidt [91] proposes a model also based on the change-and-go graph that minimizes the travel and transfer times under the assumption that the capacities of the lines allow a shortest path routing for all passengers. The idea is to overcome unrealistic effects in a system optimum, e. g., that a passenger has to use a long detour because the capacity of a shortest path is used up by other passengers; in reality passengers probably do not behave this way. A system optimum is usually computed by the other line planning models. A drawback of the shortest path routing of Schmidt, however, is that some instances may not even yield valid solutions since the total capacities do not suffice to transport all passengers on shortest paths.

Borndörfer, Grötschel, and Pfetsch [15] present a multi commodity flow model that allows to generate the paths of the passengers and the lines dynamically during the optimization process. The objective is a weighted sum of travel times and operating costs. As mentioned above, the advantage of this model is that it does not require a predefined line pool. Requirements on valid lines, e. g., turning restrictions, are integrated in the dynamic line generation. They showed the tractability of their model by solving its LP relaxation for data of the public transport system in Potsdam, a city in Germany, and present a heuristic to find a feasible solution. Solving the (integer) model by a

branch-and-bound method remains a challenge. However, computational results for a branch-and-bound implementation of this model for a given line pool (with up to 30 000 lines) are presented by Borndörfer, Neumann, and Pfetsch [25].

Nachtigall and Jerosch [74] propose to partition the passenger paths into partial routes of direct connections. This allows to count the number of transfers on each passenger path. They consider a utility function for each path by subtracting the travel time (including a transfer penalty) for each path from a minimum possible travel time multiplied by a given detour parameter. If the utility gets negative the passengers will not travel by public transport and are therefore "lost". The objective of the model is to maximize the overall utility. They propose a dynamic generation of the passenger paths similar to Borndörfer, Grötschel, and Pfetsch. They present computational results for the bus network (line pool of 119 lines) and the tram network (line pool of 18 lines) of Berlin. Using a system split, the travel demand is distributed to the different transportation systems without fixing it to the edges, and each transportation system is solved independently.

Considering these three approaches, one can conclude that a detailed treatment of transfers in line planning with integrated passenger routing seems to involve a model of enormous size. The change-and-go model of Schöbel and Scholl requires a capacity constraint for each edge in each line. In the model of Nachtigall and Jerosch, each partial passenger route has to be assigned to a certain line that can operate this partial route. This leads to a number of capacity constraints in the same dimension as for the change-and-go approach which increases polynomially with the number of considered lines, i. e., such models are hard to handle for real world problems. Compared to that, the model of Borndörfer, Grötschel, and Pfetsch is of "relatively small" size since the capacity constraints are aggregated for each transportation mode, e. g., bus or tram. However, this aggregation does not allow the treatment of transfers within one transportation mode. We will propose in this chapter a new approach that handles transfers in an approximative way. We will see that the resulting model can be solved as efficiently as the model of Borndörfer Grötschel, and Pfetsch for a given line pool, and that it estimates the number of direct travelers quite accurately. To this purpose, we will introduce the column generation model of Borndörfer, Grötschel, and Pfetsch [15] in Section 6.3 in more detail. The change-and-go approach of Schöbel and Scholl [93] is described in Section 6.4; we will use this approach for evaluations. The new approach is proposed in Section 6.5. All three models use the same setting and basic notation which we will introduce in the following section.

6.2 Notation

We state in this section basic notation and definitions for a mathematical treatment of the integrated line planning and passenger routing problem. Our setting is as follows.

Transportation Network. Considering the public transport in a city, we are usually given different *modes* of transportation, e. g., bus, tram, and subway. Let M

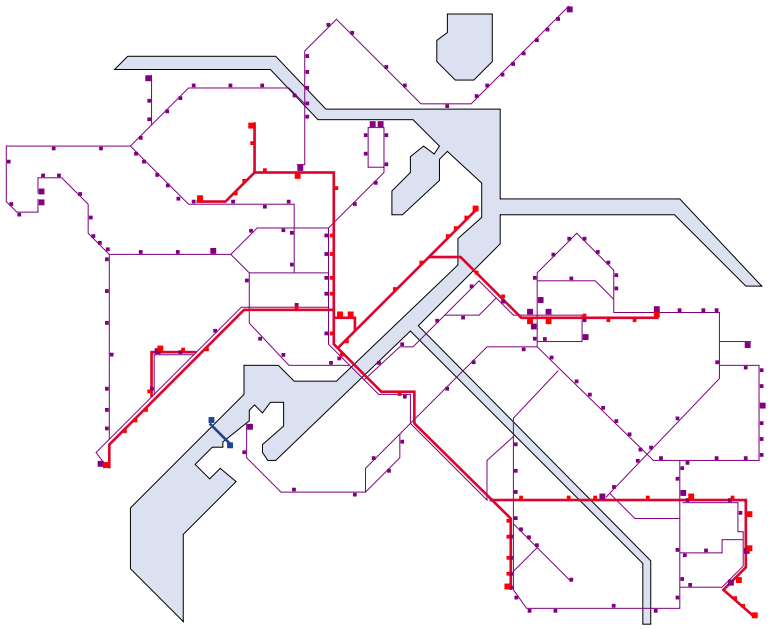


Figure 6.1: Multi-modal transportation network in Potsdam. Red: tram, violet: bus, blue: ferry, large nodes: termini, small nodes: stations, light blue: rivers and lakes.

be the number of different transportation modes and let $N_i = (V_i, E_i)$ be an undirected graph representing the infrastructure of mode $i = 1, \dots, M$, i. e., stations or stops and streets (e. g., for bus transportation) or tracks (e. g., for tram transportation) are given as nodes and edges of an *infrastructure network*. We then denote by $N = (V, E) = (V_1 \cup \dots \cup V_M, E_1 \cup \dots \cup E_M \cup E_{M+1})$ an undirected graph representing a multi-modal *transportation network*. Here, $E_T := E_{M+1} \subseteq \bigcup_{1 \le i,j \le M} V_i \times V_j$ are *transfer edges*, i. e., walking connections between stations of different or equal modes. All other edges $E_L := \cup_{i=1}^{M} E_i$ are called *infrastructure edges* or *line edges*. Each edge $e \in E$ has a *length* $l_e \in \mathbb{Q}_{\ge 0}$ and a *travel time* $\tau_e \in \mathbb{Q}_{\ge 0}$. We add a *transfer penalty* $\sigma \in \mathbb{Q}_{\ge 0}$ to the travel time of each transfer edge $e \in E_T$. Figure 6.1 shows three infrastructure networks (bus, tram, ferry) of the public transportation system of the city of Potsdam in Germany.

OD Matrix and Demand Graph. Line planning is usually based on demand data in the form of so-called *origin-destination matrices (OD-matrices)*. An OD matrix gives the number of passengers that want to travel from one point in the network to another point within a fixed time horizon. We call the points where passengers start and end their trips *OD nodes* and denote them by V_O. Each OD node is connected to the transportation network N by so called *OD edges* which we denote by E_O, i. e., $E_O \subseteq V_O \times (V_1 \cup \dots \cup V_M)$. Sometimes, an OD node is connected by a single OD edge to N, i. e., the OD node is linked to exactly one station in the public transport network. In other cases, an OD node is a virtual node representing a certain area (a traffic cell) and all stations (network nodes) within the cell are connected to this OD node by OD edges. We are also given lengths $l_e \in \mathbb{Q}_{\ge 0}$ and travel times $\tau_e \in \mathbb{Q}_{\ge 0}$ on the OD edges $e \in E_O$. Let the entries

of the (not necessarily symmetric) OD matrix be denoted by $(d_{st}) \in \mathbb{Q}_{\geq 0}^{V_O \times V_O}$, i. e., d_{st} is the number of passengers that want to travel from $s \in V_O$ to $t \in V_O$. We denote by $D = \{(s,t) \in V_O \times V_O \,|\, d_{st} > 0\}$ the set of all *OD-pairs*, i. e., the set of every two OD nodes with positive travel demand.

Let $H = (V_O, F)$ be the (undirected) *demand graph*, where $F = \big\{\{u,v\} \,|\, u,v \in V_O,\, d_{uv} + d_{vu} > 0\big\}$ is the set of edges between every two OD nodes $u, v \in V_O$ with positive demand in at least one direction.

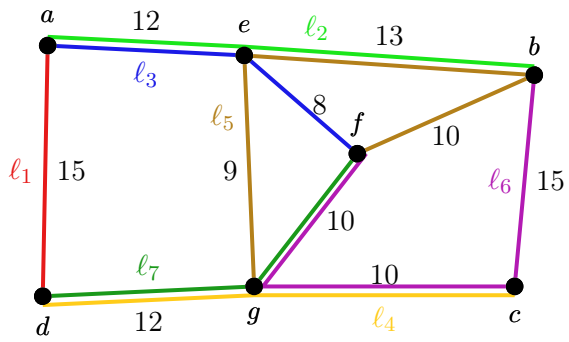
Lines and Frequencies. Given a set of *termini* $\mathcal{T}_i \subseteq V_i$ for each mode $i \in 1, \ldots, M$, a *line* ℓ of mode i is a simple path in the mode infrastructure network $N_i = (V_i, E_i)$ that starts and ends at a terminus. Let $\mathcal{L}$ be the set of all lines; $\mathcal{L}$ can be given explicitly by a so called *line pool* or implicitly by a set of rules that allow to generate lines dynamically. This depends on the model and on the restrictions for the validity of lines. Throughout this thesis we assume a given line pool. Denote by $\mathcal{F} \subseteq \mathbb{N}$ a set of possible frequencies at which these lines can be operated. It is possible to restrict the set of frequencies for different modes. In this case we denote by $\mathcal{F}_i \subseteq \mathcal{F}$ the set of possible frequencies for mode i. Then, in the following, a combination of line ℓ and frequency f is always chosen such that $f \in \mathcal{F}_i$ if line ℓ has mode i. We further denote by $F_\ell \in \mathbb{N}_{\geq 0}$ the maximal possible frequency of line ℓ which is, of course, equal for all lines of the same mode. Finally, we define by $F_e \in \mathbb{N}$ the *maximum operation frequency of edge* $e \in E$, i. e., F_e is an upper bound on the sum of the frequencies of lines operating on edge e.

A *line plan* is a tuple $(\mathcal{L}', f')$ with $\mathcal{L}' \subseteq \mathcal{L}$ being a subset of lines and f' being a function that assigns exactly one frequency to each line $\ell \in \mathcal{L}'$, i. e., $f' : \mathcal{L}' \to \mathcal{F}$.

Capacities and Costs. For a line $\ell \in \mathcal{L}$ of mode $i \in \{1, \ldots, M\}$ and a frequency $f \in \mathcal{F}$, we are given a capacity $\kappa_{\ell,f} = \kappa^i \cdot f$ with κ^i being the capacity of a line of mode i and a cost $c_{\ell,f} = C^i + f \cdot \sum_{e \in \ell} c_e^i$. The latter term includes a fixed cost $C^i \in \mathbb{Q}_{\geq 0}$ depending on the transportation mode and an operating cost per edge $c_e^i \in \mathbb{Q}_{\geq 0}$ depending also on the transportation mode; a possible definition can be $c_e^i = c^i \cdot l_e$ with $c^i \in \mathbb{Q}_{\geq 0}$, $i \in \{1, \ldots, M\}$, i. e., the operating cost depends on the length of the edge.

Passenger Routing Graph. Passengers travel along routes in a directed *passenger routing graph* $G = (V \cup V_O, A)$ that arises from the network $(V \cup V_O, E \cup E_O)$ by replacing each edge $e \in E \cup E_O$ with two antiparallel arcs $a(e)$ and $\bar{a}(e)$; conversely let $e(a) \in E \cup E_O$ be the undirected edge corresponding to $a \in A$. We will denote by A_T the set of all transfer arcs ($a \in A_T \Leftrightarrow e(a) \in E_T = E_{M+1}$), by A_O all OD arcs ($a \in A_O \Leftrightarrow e(a) \in E_O$) and by A_L all infrastructure arcs or line arcs ($a \in A_L \Leftrightarrow e(a) \in E_L = \cup_{i=1}^{M} E_i$). Travel times and lengths of the undirected edges are carried over to their directed counterparts. Let us say that an undirected line covers arcs $a(e)$ and $\bar{a}(e)$ if it contains edge $e \in E$, i. e., we interpret an undirected line ℓ in such a way that passengers can travel on this line in both directions. Denote by $\mathcal{P}_{st}$ the set of all possible directed passenger paths from $s \in V_O$ to $t \in V_O$ in G and by $\mathcal{P} = \bigcup_{(s,t) \in D} \mathcal{P}_{st}$ the set of all such paths. The travel time of path $p \in \mathcal{P}$ is $\tau_p = \sum_{a \in p} \tau_a$.

Line Planning Problem. A *feasible line plan* is a line plan, i. e., a tuple $(\mathcal{L}', f')$, $\mathcal{L}' \subseteq \mathcal{L}$,



from→ to	no. of passengers
$a \to f$	50
$a \to b$	50
$d \to f$	20
$d \to c$	80
$g \to b$	40

Figure 6.2: Public transport network with seven stops, seven lines (*left*), and given demand (*right*).

$f' : \mathcal{L}' \to \mathcal{F}$, that provides enough capacity for the given demand, i. e., there exists a multi commodity flow to transport the given demand that satisfies the capacities implied by the line plan. The *line planning problem* is to find a feasible line plan that optimizes a certain objective. There are mainly two competing objectives for line planning: minimizing passenger discomfort such as travel times or number of transfers, and minimizing the costs of a line plan.

Example 6.1. *The left of Figure 6.2 shows an infrastructure network of some public transport system. The nodes represent stations/stops and the edges represent, e. g., streets connecting the stations/stops. The numbers on the edges give the travel times in minutes. The picture also shows a pool of seven possible lines that can be established in the infrastructure network. The table on the right of Figure 6.2 lists demands that have to be satisfied, e. g., 50 passengers want to travel from station/stop a to f. Here, we assume that an OD node corresponds to exactly one station/stop. Each line can be operated once or twice, i. e., $\mathcal{F} = \{1, 2\}$; a line can transport 50 passengers at each operation, i. e., $\kappa = 50$. Finally, passengers can transfer in every stop/station from one line to another line if both lines contain this stop/station.*

Operating each line twice produces a feasible line plan. If each transfer counts as 5 minutes, then operating lines $\ell_2, \ell_3, \ell_5, \ell_7$ once and ℓ_4 twice is an alternative feasible solution with minimal travel time connections, including transfer penalties, for all passengers. In fact, all passenger can travel directly, i. e., without transfers, on shortest connections w. r. t. travel times and transfer penalties. A feasible line plan with a minimum number of lines is, e. g., to operate lines ℓ_2, ℓ_6, ℓ_7 twice. This solution is also cost minimal if we only consider a constant cost for establishing a line and no operating costs.

Remark 6.2. *If the demand graph $H = (V_O, F)$ is connected, then a feasible line plan must connect all OD nodes. In other words, if we neglect capacities and frequencies of the lines and consider a cost optimization, the line planning problem with connected demand graph reduces to the Steiner connectivity problem, see Definition 1.2.*

In the following, we will consider three approaches to model the integrated line planning and passenger routing problem in order to analyze situations as those discussed in Example 6.1 above. In particular, we will investigate models to find a feasible line plan that

minimize a weighted sum of total travel times and line operating costs. Furthermore, we will analyze how transfer penalties (as considered in the example) can be best included in the total travel time.

6.3 Basic Dynamic Model

The first model we will consider in detail is a "discrete frequency variant" of the model proposed by Borndörfer, Grötschel, and Pfetsch [15]. They developed a multi-commodity flow model that aims at a dynamic generation of line and passenger path variables. The objective is a combination of total passenger travel times and operating costs. We will consider a variant that has the above mentioned features but handles the frequency of a line explicitly, i. e., we consider a finite set of possible integral frequencies for each line instead of a continuous frequency. We will denote this model as the *basic dynamic model*.

6.3.1 Integer Program

The integer programming formulation for the basic dynamic model (BD) is

$$
\begin{aligned}
\text{(BD)} \quad \min\ & \lambda \sum_{\ell \in \mathcal{L}} \sum_{f \in \mathcal{F}} c_{\ell,f}\, x_{\ell,f} + (1-\lambda) \sum_{p \in \mathcal{P}} \tau_p\, y_p & & \\
& \sum_{p \in \mathcal{P}_{st}} y_p = d_{st} & \forall\, (s,t) \in D & \quad (6.1)\\
& \sum_{p : a \in p} y_p \le \sum_{\ell : e(a) \in \ell} \sum_{f \in \mathcal{F}} \kappa_{\ell,f}\, x_{\ell,f} & \forall\, a \in A_L & \quad (6.2)\\
& \sum_{f \in \mathcal{F}} x_{\ell,f} \le 1 & \forall\, \ell \in \mathcal{L} & \quad (6.3)\\
\sum_{\ell : e \in \ell} \sum_{f \in \mathcal{F}} f \cdot x_{\ell,f} \le F_e & & \forall\, e \in E & \quad (6.4)\\
x_{\ell,f} \in \{0,1\} & & \forall\, \ell \in \mathcal{L},\, \forall\, f \in \mathcal{F} & \quad (6.5)\\
y_p \ge 0 & & \forall\, p \in \mathcal{P}. & \quad (6.6)
\end{aligned}
$$

We consider two types of variables, a binary variable $x_{\ell,f} \in \{0,1\}$ that indicates if line ℓ is operated with frequency $f \in \mathcal{F}$, and a continuous variable $y_p \ge 0$ that accounts for the number of passengers traveling on path p. The objective of program (BD) aims at a minimization of line operating costs and passenger travel times, weighted by a parameter $0 \le \lambda \le 1$. Equations (6.1) stipulate a passenger flow of d_{st} for each OD-pair $(s,t) \in D$. Inequalities (6.2) enforce sufficient transportation capacity on each arc. Inequalities (6.3) ensure that a line is operated with at most one frequency, while inequalities (6.4) bound the sum of the frequencies of lines that can be operated on an individual edge.

We use a path formulation for both passenger paths and line paths since it allows to include possible restrictions of passenger paths and/or line paths, e. g., to avoid long detours in the network or to require, e. g., an unsplittable flow of the passenger demand. Pfetsch and Borndörfer consider different passenger routings in terms of a path based model in [78]. They also discuss an arc formulation for passenger and line paths in [16].

Program (BD) differs from the one in Borndörfer, Grötschel, and Pfetsch [15] by the use of binary variables $x_{\ell,f}$ for the operation of line $\ell \in \mathcal{L}$ at frequency $f \in \mathcal{F}$ instead of a binary variable x_ℓ that indicates if line ℓ is operated and a continuous variable $f_\ell \geq 0$ for the frequency of line ℓ. We use a discrete set of frequencies since this is an important requirement in practice. It can, however, be shown that we get the same LP relaxation for model (BD) as Borndörfer, Grötschel, and Pfetsch for the case $F_\ell \geq F_e$ for all $\ell \in \mathcal{L}$, $e \in E$. More precisely, the following is true.

Proposition 6.3. *Let $F_\ell \geq F_e$ for all $\ell \in \mathcal{L}$, $e \in E$, i. e., the maximal frequency to operate a line is at least as high as the maximum operation frequency of an edge. Let further be the costs and the capacities as defined in Section 6.2, i.e., $\kappa_{\ell,f} = \kappa^i \cdot f$ with $\kappa^i \geq 0$ and $c_{\ell,f} = C^i + f \cdot \sum_{e\in\ell} c^i_e$ with $C^i, c^i_e \geq 0$ for all $\ell \in \mathcal{L}$, $f \in \mathcal{F}$. Then each optimal LP solution x^* of (BD) satisfies $x^*_{\ell,f} = 0$ for all $\ell \in \mathcal{L}$, $f \in \mathcal{F} \setminus \{F_\ell\}$.*

Proof. Assume there is an optimal LP solution x^* for model (BD) with $x^*_{\ell,f_1} > 0$ for $f_1 \in \mathcal{F} \setminus \{F_\ell\}$, $\ell \in \mathcal{L}$. Let $\tilde{f} := \sum_{f\in\mathcal{F}} f \cdot x^*_{\ell,f}$. Note that $0 \leq \tilde{f} \leq F_e \leq F_\ell$ for all $\ell \in \mathcal{L}$, $e \in E$. Then $x^{**} := x^*$ with $x^{**}_{\ell,f} := 0$ for all $f \in \mathcal{F} \setminus \{F_\ell\}$ and $x^{**}_{\ell,F_\ell} := \frac{\tilde{f}}{F_\ell}$ yields an LP solution x^{**} for (BD) with the same capacities as for x^* and lower cost since

$$\begin{aligned}
\sum_{f\in\mathcal{F}} \kappa_{\ell,f} x^*_{\ell,f} &= \kappa^i \sum_{f\in\mathcal{F}} f\, x^*_{\ell,f} = \kappa^i \tilde{f} = \kappa^i \kappa^i F_\ell x^{**}_{\ell,F_\ell} = \sum_{f\in\mathcal{F}} \kappa_{\ell,f} x^{**}_{\ell,f} \\
\sum_{f\in\mathcal{F}} c_{\ell,f} x^*_{\ell,f} &= \sum_{f\in\mathcal{F}} C^i\, x^*_{\ell,f} + \sum_{f\in\mathcal{F}} f x^*_{\ell,f} \cdot \sum_{e\in\ell} c^i_e = \sum_{f\in\mathcal{F}} C^i\, x^*_{\ell,f} + F_\ell x^{**}_{\ell,F_\ell} \sum_{e\in\ell} c^i_e \\
&\geq \sum_{f\in\mathcal{F}} c_{\ell,f} x^{**}_{\ell,f} \qquad \Big(\sum_{f\in\mathcal{F}} x^*_{\ell,f} \geq x^{**}_{\ell,f}\Big).
\end{aligned}$$

□

Proposition 6.3 implies that we can omit all line variables that do not correspond to the maximal frequency in the LP relaxation of model (BD), i. e., inequalities (6.3) are redundant for the LP relaxation of model (BD):

$$\begin{aligned}
(\mathrm{BD}_{\mathrm{LP}}) \qquad \min\ & \lambda \sum_{\ell\in\mathcal{L}} \gamma_\ell\, f_\ell + (1-\lambda) \sum_{p\in\mathcal{P}} \tau_p\, y_p & & \\
& \sum_{p\in\mathcal{P}_{st}} y_p = d_{st} & & \forall\, (s,t) \in D \\
& \sum_{p:a\in p} y_p \leq \sum_{\ell:e(a)\in\ell} \kappa_\ell\, f_\ell & & \forall\, a \in A_L \\
& \sum_{\ell:e\in\ell} f_\ell \leq F_e & & \forall\, e \in E \\
& f_\ell \geq 0 & & \forall\, \ell \in \mathcal{L} \\
& y_p \geq 0 & & \forall\, p \in \mathcal{P},
\end{aligned}$$

where $x_{\ell,F} \cdot F_\ell$ is substituted by f_ℓ and $\gamma_\ell := \frac{C^i}{F_\ell} + \sum_{e \in \ell} c^i_e$. This is exactly the same LP relaxation as Borndörfer, Grötschel, and Pfetsch derived in their paper [15]. Note that they use the same definition of line cost and the same assumption, namely, $f_\ell \leq F$, $\ell \in \mathcal{L}$ with $F \geq F_e$ for all $e \in E$.

6.3.2 Pricing Problem

The number of passenger variables in model (BD) is usually large, i. e., exponential in the number of arcs of the passenger routing graph. To handle such a problem, the LP relaxation is solved by column generation and the integer program by a branch-and-price algorithm. A *column generation* or a dynamic generation of variables means that the linear program is solved iteratively by looking in each step at a subset of variables and adding additional variables if they have negative reduced cost. The problem of finding variables with negative reduced cost is called the *pricing problem*, see also Desrosiers and Lübbecke [42] for a didactical example of column generation.

The reduced cost can be read off the dual program. Associate dual variables π (unbounded), $\mu \geq 0$, $\psi \geq 0$, and $\eta \geq 0$ with constraints (6.1), (6.2), (6.3), and (6.4) of program (BD). The dual program is then

$$\begin{aligned}
\max \sum_{(s,t) \in D} d_{st} \pi_{st} - \sum_{\ell \in \mathcal{L}} \psi_\ell - \sum_{e \in E} F_e \eta_e & \\
\pi_{st} - \sum_{a \in p} \mu_a \leq (1-\lambda)\tau_p & \quad \forall\, (s,t) \in D,\, \forall\, p \in \mathcal{P}_{st}, \\
\kappa_{\ell,f} \sum_{a : e(a) \in \ell} \mu_a - \psi_\ell - f \sum_{e \in \ell} \eta_e \leq \lambda c_{\ell,f} & \quad \forall\, \ell \in \mathcal{L},\, \forall\, f \in \mathcal{F} \\
\mu_a \geq 0 & \quad \forall\, a \in A \\
\eta_e \geq 0 & \quad \forall\, e \in E.
\end{aligned}$$

The reduced cost $\overline{\tau}_p$ for variable y_p with $p \in \mathcal{P}_{st}$, $(s,t) \in D$, is

$$\overline{\tau}_p = (1-\lambda)\tau_p - \pi_{st} + \sum_{a \in p} \mu_a = -\pi_{st} + \sum_{a \in p} (\mu_a + (1-\lambda)\tau_a).$$

The pricing problem for the y-variables is to find a path p with $\overline{\tau}_p < 0$ or to conclude that no such path exists. This can be done in polynomial time as follows. For all $(s,t) \in D$, we search for a shortest (s,t)-path p with respect to the nonnegative weights $(\mu_a + (1-\lambda)\tau_a)$ on the arcs; we can, for instance, use Dijkstra's algorithm. If the length of this path p is less than π_{st}, then y_p is a candidate variable to be added to the LP, otherwise, we proved that no such path exists (for the pair (s,t)).

Proposition 6.4. *The pricing problem for passenger path variables in the LP relaxation of program (BD) is a shortest path problem. It can be solved in polynomial time.*

Remark 6.5. *Assume $\mathcal{L}$ is not a predefined line pool but has to be generated dynamically. Since the LP relaxation of model (BD) is equivalent to the LP relaxation of the original*

model of Borndörfer, Grötschel, and Pfetsch it follows with the same arguments as in [15] that the pricing problem for the line path variables is a longest path problem and thus $\mathcal{NP}$-hard. If the lines have length in $\mathcal{O}(\log |V|)$, it can be solved in polynomial time.

A drawback of model (BD) is the treatment of transfers. Passenger paths and line paths are only coupled via the capacity constraints (6.2) in program (BD). Since an arc aggregates the utilization of all lines of the same mode, passengers are not assigned to a line and, thus, many transfers, especially between lines of the same mode, cannot be controlled. Transfers along transfer arcs, in particular transfers between lines of different modes, can be considered by adding a transfer penalty to the travel time of those arcs. Using an appropriate expansion of the graph, i. e., using multiple copies of infrastructure arcs, one for each line, and including an arc for each possible transfer, would allow to incorporate all transfers. This is the idea of the change-and-go model which we will introduce in the following section.

6.4 Change-and-Go Model

Schöbel and Scholl [93] proposed an approach that allows a detailed treatment of transfers. The inclusion of transfers is necessary because transfers are an important decision factor when choosing the travel routes in public transport. The idea of this model is to set up a so-called *change-and-go network* that contains a copy of each node and each edge for every line that contains this node and edge, respectively. Further transfer edges are added when needed.

6.4.1 Integer Program

A formal description of the change-and-go model is as follows. The basis is a directed *change-and-go graph* $G_{CG} = (\mathcal{V}, \mathcal{A})$ which corresponds to the passenger routing graph, see Section 6.2. Its nodes $\mathcal{V} = \mathcal{V}_O \cup \mathcal{V}_L$ represent origin/destination nodes and station-line-pairs, i. e.,

$$\mathcal{V}_L = \{(v,\ell) : v \in V,\, \ell \in \mathcal{L},\, v \in V(\ell)\}, \qquad \mathcal{V}_O = \{s \in V_O : \exists\, t \in V_O,\, d_{st} + d_{ts} > 0\}.$$

Its arcs $\mathcal{A} = \mathcal{A}_O \cup \mathcal{A}_L \cup \mathcal{A}_T$ connect nodes of $\mathcal{V}_O$ to nodes of $\mathcal{V}_L$ (*OD arcs*, $\mathcal{A}_O$) and different nodes in $\mathcal{V}_L$. We distinguish connections between different nodes of the same line (*traveling arcs*, $\mathcal{A}_L$), and different nodes at the same station (*transfer arcs*, $\mathcal{A}_T$):

$$\begin{aligned}
\mathcal{A}_O &= \{(s,(v,\ell)),((v,\ell),s) : s \in \mathcal{V}_O,\, (v,\ell) \in \mathcal{V}_L,\, (s,v) \in A_O\} \\
\mathcal{A}_L &= \{((u,\ell),(v,\ell)) : (u,\ell),(v,\ell) \in \mathcal{V}_L,\, (u,v) \in A\} \\
\mathcal{A}_T &= \{((v,\ell),(v,\tilde{\ell})) : (v,\ell),(v,\tilde{\ell}) \in \mathcal{V}_L,\, \ell \neq \tilde{\ell}\} \\
&\quad \cup \{((u,\ell),(v,\tilde{\ell})) : (u,\ell),(v,\tilde{\ell}) \in \mathcal{V}_L,\, \ell \neq \tilde{\ell},\, (u,v) \in A_T\}.
\end{aligned}$$

Travel times and lengths of arcs in G_{CG} are set to travel times and lengths of the "original" arcs in G, all transfer arcs are (additionally) penalized by $\sigma \in \mathbb{Q}_{\geq 0}$. Denote by $e(a) \in E_L$

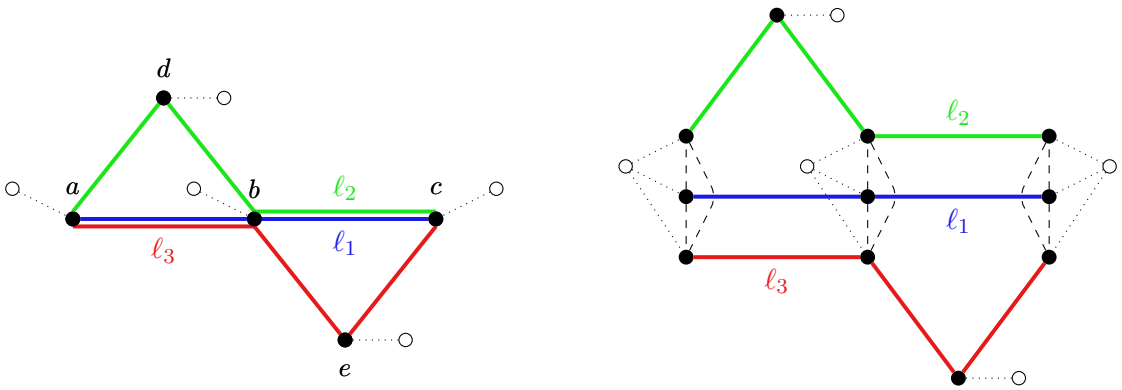


Figure 6.3: Construction of an undirected change-and-go graph. *Left:* A small piece of a transportation network with three lines, OD nodes (unfilled nodes), and OD edges (dotted). *Right:* Undirected change-and-go graph: the line nodes ($\mathcal{V}_L$) are filled, OD nodes ($\mathcal{V}_O$) not; the line arcs ($\mathcal{A}_L$) correspond to solid edges, transfer arcs ($\mathcal{A}_T$) to dashed edges, and OD arcs ($\mathcal{A}_O$) to dotted edges.

the edge in the infrastructure network that is associated with $a \in \mathcal{A}_L$. Note that each $a \in \mathcal{A}_L$ is associated with exactly one line $\ell \in \mathcal{L}$; denote by $\ell(a)$ the line that is associated with $a \in \mathcal{A}_L$.

To keep track of all relevant details of the construction, Figure 6.3 shows a visualization of the undirected counterpart of the change-and-go graph. We get the full change-and-go graph by replacing each edge with two oppositely directed arcs.

We use the same notation for the passenger routing in G_{CG} as for the passenger routing in G. We denote by $\mathcal{P}_{st}$ the set of all possible directed paths from $s \in \mathcal{V}_O$ to $t \in \mathcal{V}_O$ in G_{CG} and by $\mathcal{P} = \bigcup_{(s,t)\in D} \mathcal{P}_{st}$ the set of all such paths. The travel time τ_p of the passenger path $p \in \mathcal{P}$ then accounts for the travel time and all transfer penalties. The integer programming formulation (CG) of the line planning problem based on the change-and-go graph is

$$
\begin{aligned}
\text{(CG)} \quad \min\ & \lambda \sum_{\ell\in\mathcal{L}} \sum_{f\in\mathcal{F}} c_{\ell,f}\, x_{\ell,f} + (1-\lambda) \sum_{p\in\mathcal{P}} \tau_p\, y_p && && \\
& \sum_{p\in\mathcal{P}_{st}} y_p = d_{st} && \forall\, (s,t) \in D && (6.7)\\
& \sum_{p:a\in p} y_p \le \sum_{f\in\mathcal{F}} \kappa_{\ell(a),f}\, x_{\ell(a),f} && \forall\, a \in \mathcal{A}_L && (6.8)\\
& \sum_{f\in\mathcal{F}} x_{\ell,f} \le 1 && \forall\, \ell \in \mathcal{L} && (6.9)\\
& \sum_{\ell:e\in\ell} \sum_{f\in\mathcal{F}} f \cdot x_{\ell,f} \le F_e && \forall\, e \in E && (6.10)\\
& x_{\ell,f} \in \{0,1\} && \forall\, \ell \in \mathcal{L},\, \forall\, f \in \mathcal{F} && (6.11)\\
& y_p \ge 0 && \forall\, p \in \mathcal{P}. && (6.12)
\end{aligned}
$$

We have a binary variable $x_{\ell,f}$ that indicates if line ℓ is operated with frequency f, and a continuous variable y_p that accounts for the number of passengers traveling on path p.

The model (CG) minimizes a weighted sum of line operating costs and passenger travel times including a penalty for each transfer. Equations (6.7) stipulate a passenger flow of d_{st} for each OD-pair $(s,t) \in D$. Inequalities (6.8) enforce sufficient transportation capacity on each line arc. Inequalities (6.9) ensure that a line is operated with at most one frequency, and inequalities (6.10) bound the sum of the line operation frequencies for each edge.

6.4.2 Pricing Problem

We associate dual variables π (unbounded), $\mu \geq 0$, $\psi \geq 0$, and $\eta \geq 0$ with constraints (6.7), (6.8), (6.9), and (6.10) of (CG). The dual program of the LP relaxation of (CG) is then

$$\begin{aligned}
\max \sum_{(s,t)\in D} d_{st}\pi_{st} - \sum_{\ell\in\mathcal{L}} \psi_\ell - \sum_{e\in E} F_e\eta_e & \\
\pi_{st} - \sum_{a\in p} \mu_a \leq (1-\lambda)\tau_p & \quad \forall\,(s,t)\in D,\ \forall\, p\in\mathcal{P}_{st}, \\
\kappa_{\ell,f} \sum_{a:\ell(a)=\ell} \mu_a - \psi_\ell - f\sum_{e\in\ell} \eta_e \leq \lambda c_{\ell,f} & \quad \forall\,\ell\in\mathcal{L},\ \forall\, f\in\mathcal{F} \\
\mu_a \geq 0 & \quad \forall\, a\in\mathcal{A}_L \\
\eta_e \geq 0 & \quad \forall\, e\in E \\
\psi_\ell \geq 0 & \quad \forall\,\ell\in\mathcal{L}.
\end{aligned}$$

It is easy to see that we get the same passenger path pricing problem, a shortest path problem, as for model (BD), see Subsection 6.3.2, except that the underlying graph is the change-and-go graph.

6.5 Direct Connection Models

The change-and-go model (CG) can be used to compute the travel times including a penalty for each transfer. It uses the same line variables as the basic dynamic model, but a more detailed definition of passenger path variables and a much larger number of rows, namely, one row for every arc in every line. This makes the model much larger and leads to memory problems and high computation times when solving the problem.

On the other hand, the basic dynamic model (BD) can be solved much faster, but it neglects transfers within a transportation mode. This is a serious drawback, because in urban public transport the number of transfers is an important convenience factor and also a decision factor for using public transport. We therefore want to propose in this section a novel *direct connection* approach that tries to combine the advantages of both models. It focuses on direct connections by penalizing paths that do not provide direct connections.

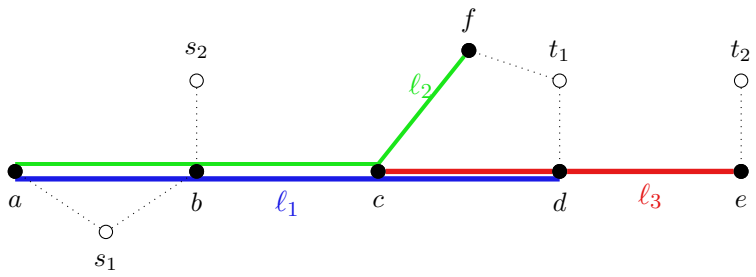


Figure 6.4: A public transport network with six stops (filled nodes), four OD nodes (unfilled), and three lines; OD edges are dotted. Recall that the directed passenger routing graph is constructed by replacing each undirected edge by two directed arcs. The paths (s_1, a, b, c, d, t_1) and (s_1, b, c, d, t_1) are s_1t_1-dcpaths for line ℓ_1, i. e., $\mathcal{P}^{0,\ell_1}_{s_1t_1} = \{(s_1, a, b, c, d, t_1), (s_1, b, c, d, t_1)\}$; $\mathcal{P}^{0,\ell_2}_{s_1t_1} = \{(s_1, a, b, c, f, t_1), (s_1, b, c, f, t_1)\}$. No s_1t_2-path is supported by a direct connection line, i. e., there exists no s_1t_2-dcpath, i. e., $\mathcal{P}^{0}_{s_1t_2} = \emptyset$. Note that $\mathcal{P}^{0,\ell_1}_{s_2t_1} = \{(s_2, b, c, d, t_1)\}$ and $\mathcal{P}^{0,\ell_2}_{s_2t_1} = \{(s_2, b, c, f, t_1)\}$, i. e., $|\mathcal{P}^{0,\ell_1}_{s_2t_1}| = |\mathcal{P}^{0,\ell_2}_{s_2t_1}| = 1$ but $\delta^-(t_1) = 2$.

We will first introduce an intermediate *direct line connection model* that accounts exactly for the number of travelers on direct connections according to the model assumptions. We then derive in two steps the *direct connection model* as a compact approximation of the direct line connection model. We finally extend the direct connection model by the incorporation of *unavoidable transfers*.

6.5.1 Exact Model – Direct Line Connections

The idea of the direct connection model is to favor direct connections by adding a transfer penalty to the travel time of paths including transfers, i. e., if the passengers on some path are forced to transfer in the computed line plan, we associate with the travel time an additional transfer penalty. To formulate this idea, we extend the notation of Section 6.2 as follows.

A *direct connection st-passenger path for line* ℓ or an *st-dcpath for line* ℓ is an st-passenger path p of the form $p = (s, a_0, v_1, \dots, v_r, a_r, t)$, where $e(a_i) \in \ell$, $i = 1, \dots, r-1$, i. e., passengers can travel along p from origin s directly to destination t via line ℓ without transfers. Let $\mathcal{P}^{0,\ell}_{st} \subseteq \mathcal{P}_{st}$ be the set of st-dcpaths for line ℓ and $\mathcal{P}^{0}_{st} = \bigcup_{\ell\in\mathcal{L}} \mathcal{P}^{0,\ell}_{st}$, $\mathcal{P}^{0,\ell} = \bigcup_{(s,t)\in D} \mathcal{P}^{0,\ell}_{st}$ be their union over lines and OD pairs, respectively. Figure 6.4 illustrates the notation. Note that $|\mathcal{P}^{0,\ell}_{st}| > 1$ is possible if $|\delta^+(s)| > 1$ or $|\delta^-(t)| > 1$ $(s, t \in V_O)$. Let further $\mathcal{P}^{0,\ell}_{st}(a) = \{p \in \mathcal{P}^{0,\ell}_{st} : a \in p\}$ be the set of st-dcpaths for line ℓ that pass over arc a and $\mathcal{P}^{0}_{st}(a) = \bigcup_{\ell\in\mathcal{L}} \mathcal{P}^{0,\ell}_{st}(a)$, $\mathcal{P}^{0,\ell}(a) = \bigcup_{(s,t)\in D} \mathcal{P}^{0,\ell}_{st}(a)$ their union over lines and OD pairs, respectively. We denote by $\mathcal{L}_{st} = \{\ell \in \mathcal{L} : \mathcal{P}^{0,\ell}_{st} \neq \emptyset\}$ the set of all *direct connection lines* for s, t, and let $\mathcal{L}_{st}(a) = \{\ell \in \mathcal{L}_{st} : a \in \ell\}$ be the set of all lines that support an st-dcpath via arc a. A path p is a *direct connection st-passenger path* (st-dcpath), if it is an st-dcpath for some line ℓ. We denote by $\mathcal{P}^0 = \bigcup_{(s,t)\in D} \mathcal{P}^0_{st}$ the union of all st-dcpaths, i. e., the set of all *dcpaths*. For a dcpath $p \in \mathcal{P}^0$, we set the travel time to the sum of the arc travel times $\tau_{p,0} = \sum_{a\in p} \tau_a$. For an st-passenger path $p \in \mathcal{P}$, we set the travel time to the sum of the arc travel times plus a summand $\sigma(p)$ to

arrive at a travel time of $\tau_{p,1} = \sigma(p) + \sum_{a \in p} \tau_a$, where $\sigma(p) = \sigma$ if p does not contain a transfer arc, i. e., $p \cap A_T = \emptyset$, and 0 otherwise, since we already incorporated a penalty on transfer arcs.

We introduce path flow variables $z^\ell_{p,0}$, $p \in \mathcal{P}^0$, and $y_{p,1}$, $p \in \mathcal{P}$, for the number of passengers that travel on dcpath p on line ℓ and on path p with at least one transfer, respectively. Note that the definition of the variables $y_{p,1}$ also involves dcpaths $p \in \mathcal{P}^0 \subseteq \mathcal{P}$. The idea is that a passenger on a dcpath has to transfer if there is not enough transportation capacity of direct connection lines. Having further variables $x_{\ell,f} \in \{0,1\}$ for the operation of line ℓ at frequency f, we state the *direct line connection model* (DLC) as follows:

$$\text{(DLC)} \quad \min \lambda \sum_{\ell \in \mathcal{L}} \sum_{f \in \mathcal{F}} c_{\ell,f}\, x_{\ell,f} + (1-\lambda)\Big(\sum_{\ell \in \mathcal{L}} \sum_{p \in \mathcal{P}^{0,\ell}} \tau_{p,0}\, z^\ell_{p,0} + \sum_{p \in \mathcal{P}} \tau_{p,1}\, y_{p,1} \Big)$$

$$\sum_{\ell \in \mathcal{L}_{st}} \sum_{p \in \mathcal{P}^{0,\ell}_{st}} z^\ell_{p,0} + \sum_{p \in \mathcal{P}_{st}} y_{p,1} = d_{st} \qquad \forall\, (s,t) \in D \tag{6.13}$$

$$\sum_{\ell \in \mathcal{L}} \sum_{p \in \mathcal{P}^{0,\ell}(a)} z^\ell_{p,0} + \sum_{p \in \mathcal{P}: a \in p} y_{p,1} \le \sum_{\ell \in \mathcal{L}: e(a) \in \ell} \sum_{f \in \mathcal{F}} \kappa_{\ell,f}\, x_{\ell,f} \qquad \forall\, a \in A \tag{6.14}$$

$$\sum_{p \in \mathcal{P}^{0,\ell}(a)} z^\ell_{p,0} \le \sum_{f \in \mathcal{F}} \kappa_{\ell,f}\, x_{\ell,f} \qquad \forall\, \ell \in \mathcal{L},\, \forall\, a \in A \tag{6.15}$$

$$\sum_{f \in \mathcal{F}} x_{\ell,f} \le 1 \qquad \forall\, \ell \in \mathcal{L} \tag{6.16}$$

$$\sum_{\ell \in \mathcal{L}: e \in \ell} \sum_{f \in \mathcal{F}} f \cdot x_{\ell,f} \le F_e \qquad \forall\, e \in E \tag{6.17}$$

$$x_{\ell,f} \in \{0,1\} \qquad \forall\, \ell \in \mathcal{L},\, \forall\, f \in \mathcal{F} \tag{6.18}$$

$$z^\ell_{p,0} \ge 0 \qquad \forall\, \ell \in \mathcal{L},\, \forall\, p \in \mathcal{P}^{0,\ell} \tag{6.19}$$

$$y_{p,1} \ge 0 \qquad \forall\, p \in \mathcal{P}. \tag{6.20}$$

Model (DLC) minimizes a weighted sum of line operating costs and passenger travel times. Note that the st-passenger path variables $y_{p,1}$ incur a penalty for each transfer arc and exactly one transfer penalty otherwise, i. e., the number of transfers may be underestimated. Equations (6.13) enforce the passenger flow. Inequalities (6.14) guarantee sufficient total transportation capacity on each arc. Constraints (6.15), the *direct line connection constraints*, ensure sufficient transportation capacity for direct connection passenger paths on each arc of each line. Inequalities (6.16) ensure that a line is operated at one frequency at most and inequalities (6.17) bound the sum of the line operation frequencies for each edge.

Model (DLC) includes a variable $z^\ell_{p,0}$ for the assignment of each direct connection passenger path p to a direct connection line ℓ. This is similar to the direct traveler model of Bussieck, Kreuzer, and Zimmermann [30]. Using our notation the direct traveler model

(DT) reads as follows

$$
\begin{aligned}
\text{(DT)} \quad \max \sum_{(s,t)\in D} \sum_{\ell\in\mathcal{L}} w_{st}^{\ell} z_{st}^{\ell} & & & \\
\sum_{\ell\in\mathcal{L}:e\in\ell} \sum_{f\in\mathcal{F}} \kappa_{\ell,f} x_{\ell,f} &\leq \overline{c}_e & \forall e \in E & \quad (6.21)\\
\sum_{\ell\in\mathcal{L}:e\in\ell} \sum_{f\in\mathcal{F}} \kappa_{\ell,f} x_{\ell,f} &\geq \underline{c}_e & \forall e \in E & \quad (6.22)\\
\sum_{\ell\in\mathcal{L}} z_{st}^{\ell} &\leq d_{st} & \forall (s,t) \in D & \quad (6.23)\\
\sum_{(s,t)\in D:\ell\in\mathcal{L}_{st},e\in q_\ell(s,t)} z_{st}^{\ell} &\leq \sum_{f\in\mathcal{F}} \kappa_{\ell,f} x_{\ell,f} & \forall \ell \in \mathcal{L},\, \forall\, e \in \ell & \quad (6.24)\\
x_{\ell,f} &\in \{0,1\} & \forall\, \ell \in \mathcal{L},\, \forall\, f \in \mathcal{F} & \quad (6.25)\\
z_{st}^{\ell} &\geq 0 & \forall\, (s,t) \in D,\, \forall\, \ell \in \mathcal{L}_{st}, & \quad (6.26)
\end{aligned}
$$

where w_{st}^{ℓ} is, e. g., inversely proportional to the travel time from s to t on line ℓ. Model (DT) has for each OD pair $(s,t) \in D$ and each direct connection line $\ell \in \mathcal{L}_{st}$ a variable z_{st}^{ℓ} that assigns direct travelers from s to t to line ℓ. The capacity constraints (6.21) and (6.22) of model (DT) ensure that given lower bounds $\underline{c} \in \mathbb{R}^E_{\geq 0}$ and upper bounds $\overline{c} \in \mathbb{R}^E_{\geq 0}$ for the demand on each edge in the network are satisfied. Inequalities (6.23) bound the number of direct travelers for each OD pair $(s,t) \in D$ by the maximal number of travelers for (s,t) while inequalities (6.24) bound the number of direct travelers of line $\ell \in \mathcal{L}$ on edge $e \in E$ by the capacity of line ℓ.

Note that the direct traveler model does not distinguish between the paths (s_1, a, b, c, d, t_1) and (s_1, b, c, d, t_1) in Figure 6.4. Indeed, exactly one st-path has to be assigned to each direct connection line $\ell \in \mathcal{L}_{st}$ in advance (which we denoted by $q_\ell(s,t) \subseteq \ell$ in model (DT)). The main difference, however, between the direct traveler model (DT) and the direct line connection model (DLC) is the passenger routing. More precisely, the bounds $\overline{c}, \underline{c} \in \mathbb{R}^E_{\geq 0}$ in (DT) are supposed to guarantee that a precomputed passenger routing is possible. The optimization model then computes frequencies for the lines satisfying these bounds such that the number of direct travelers is maximal. Here, it is allowed that passengers change their routes to use direct connection lines. But these changes are only allowed according to the given capacity bounds on the edges, i. e., the precomputed passenger routing restricts these changes. All other passengers, i. e., passengers on non-direct connections, are not considered. It is, hence, possible that direct travelers force other passengers to use long detours by utilizing the capacity on short connections. Even more, it is not clear whether there exists a passenger routing for all passengers that yields the number of direct travelers computed with model (DT). Model (DT) also does not include the costs of a line plan, compare with Section 6.1. Model (DLC) overcomes both drawbacks of model (DT). It computes a passenger routing depending on the computed line plan and minimizes costs for the line plan as well as travel times for passengers on direct connections and non-direct connections.

A problem with both models is their complexity. A line of length k is usually a direct connection line for $\mathcal{O}(k^2)$ OD-pairs, such that the number of variables is much larger than the number of lines. Moreover, choices between several possible direct connection lines for every dcpath produce lots of degeneracy. Bussieck [29] stated that solving the LP relaxation of the direct traveler model for medium and large problem sizes with Cplex 3.0 "required an exhaustive use of resources [...] which is absolutely unacceptable for a practically relevant approach". Hence, Bussieck [29] and Bussieck, Kreuzer, and Zimmermann [30] propose an approximation model by aggregating direct traveler variables of the same OD pair. They end up with one direct traveler variable for each OD pair $(s,t) \in D$. We will use a similar idea to compress our model by relaxing the explicit assignment of dcpaths to direct connection lines. However, in contrast to Bussieck, Kreuzer, and Zimmermann we keep the traveling paths for all passengers on direct connections, i. e., we end up with a direct connection passenger path variable $y_{p,0}$ for each path $p \in \mathcal{P}^0_{st}$ and each OD pair $(s,t) \in D$.

6.5.2 Approximation Model I – Relaxing Direct Line Capacities

To construct a compact approximation of (DLC), we eliminate the assignment of passenger paths to particular lines by aggregating the dcpath variables as $y_{p,0} = \sum_{\ell \in \mathcal{L}} z^\ell_{p,0}$, i. e., we introduce line-independent dcpath variables $y_{p,0}$ for the number of direct travelers on path p. Such a substitution can be easily done in the objective of model (DLC) and in the constraints (6.13) and (6.14). More effort is needed to replace the direct line connection constraints

$$\sum_{p \in \mathcal{P}^{0,\ell}(a)} z^\ell_{p,0} \leq \sum_{f \in \mathcal{F}} \kappa_{\ell,f}\, x_{\ell,f} \quad \forall\, \ell \in \mathcal{L}, \forall\, a \in A. \tag{6.15}$$

The left hand side of a direct line connection constraint is the number of all passengers on direct connections using a certain direct connection line on a certain arc. This number is bounded by the capacity of the considered line. Since the left hand side involves only dcpath variables for one line, there is no easy way to rephrase (6.15) in terms of aggregated dcpath variables. A simple approximation is the following

$$\sum_{p \in \mathcal{P}^0_{st}(a)} y_{p,0} \leq \sum_{\ell \in \mathcal{L}_{st}: e(a) \in \ell} \sum_{f \in \mathcal{F}} \kappa_{\ell,f} x_{\ell,f} \quad \forall\, a \in A, \forall\, (s,t) \in D. \tag{6.27}$$

Constraints (6.27) ensure that there is enough capacity on direct connection lines for each OD pair and each arc individually. More precisely, considering OD pair $(s,t) \in D$ and arc $a \in A$, constraints (6.27) guarantee that the number of direct travelers from s to t on a is always smaller than or equal to the total capacity of all direct connection st-lines containing a. However, the needed capacity of a direct connection line is probably underestimated since it is a direct connection line for several OD pairs. We explain this problem by means of an example, consider Figure 6.5. Assume we have demands of 50 from s_1 to t_1 and from s_2 to t_1 and lines ℓ_1, ℓ_3, and ℓ_4 are operated once with capacity 50 each. Line ℓ_2 is not operated. Then constraints (6.27) are satisfied for OD pair (s_1, t_1)

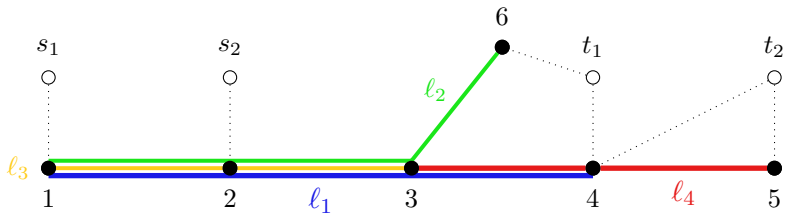


Figure 6.5: A public transport network with five stops (filled nodes), four OD nodes (unfilled), and three lines; OD edges are dotted. Recall that the directed passenger routing graph is constructed by replacing each undirected edge by two directed arcs.

as well as for OD pair (s_2, t_1) and for arcs $(2, 3)$ as well as $(3, 4)$, e. g., for (s_1, t_1) and arc $(2, 3)$ we get

$$y_{(s_1,1,2,3,4,t_1),0} + y_{(s_1,1,2,3,6,t_1),0} \leq \sum_{f \in \mathcal{F}} (\kappa_{\ell_1,f} x_{\ell_1,f} + \kappa_{\ell_2,f} x_{\ell_2,f}),$$

for (s_2, t_1) and arc $(2, 3)$ we get

$$y_{(s_2,2,3,4,t_1),0} + y_{(s_2,2,3,6,t_1),0} \leq \sum_{f \in \mathcal{F}} (\kappa_{\ell_1,f} x_{\ell_1,f} + \kappa_{\ell_2,f} x_{\ell_2,f}).$$

But the capacity of line ℓ_1 does not suffice to transport the passengers from s_1 to t_1 *and* the passengers from s_2 to t_1 directly; 50 passengers have to change from ℓ_3 to ℓ_4. In this example both OD pairs (s_1, t_1) and (s_2, t_1) have the same set of direct connection lines, i. e., $\mathcal{L}_{s_1 t_1} = \mathcal{L}_{s_2 t_1}$. The relaxation of the direct line connection constraints can therefore be improved by considering all direct connection passenger paths on one arc using the same set of direct connection lines. We then get for arc $(2, 3)$ in our example the following constraint

$$y_{(s_1,1,2,3,4,t_1),0} + y_{(s_1,1,2,3,6,t_1),0} + y_{(s_2,2,3,4,t_1),0} + y_{(s_2,2,3,6,t_1),0} \leq \sum_{f \in \mathcal{F}} (\kappa_{\ell_1,f} x_{\ell_1,f} + \kappa_{\ell_2,f} x_{\ell_2,f}).$$

We say that OD-pairs (s, t) and (u, v) are *dc-equivalent with respect to arc* a, if $\mathcal{L}_{uv}(a) = \mathcal{L}_{st}(a)$, i. e., if the st- and the uv-dcpaths via arc a are supported by the same set of direct connection lines (also via arc a). In Figure 6.5 the OD pairs (s_1, t_1) and (s_2, t_1) are dc-equivalent with respect to arc $(2, 3)$ because $\mathcal{L}_{s_1 t_1}(2, 3) = \mathcal{L}_{s_2 t_1}(2, 3) = \{\ell_1, \ell_2\}$. They are also dc-equivalent to arc $(3, 4)$ because $\mathcal{L}_{s_1 t_1}(3, 4) = \mathcal{L}_{s_2 t_1}(3, 4) = \{\ell_1\}$. Denote by $[s, t]_a$ the corresponding equivalence class, i. e., $(u, v) \in [s, t]_a$ if $\mathcal{L}_{uv}(a) = \mathcal{L}_{st}(a)$. Let $D(a) = \{[s, t]_a\}$ be the set of equivalence classes for dc-equivalent OD-pairs w. r. t. a. Then a better approximation than (6.27) for the direct line capacity constraints is

$$\sum_{(u,v) \in [s,t]_a} \sum_{p \in \mathcal{P}^0_{uv}(a)} y_{p,0} \leq \sum_{\ell \in \mathcal{L}_{st}(a)} \sum_{f \in \mathcal{F}} \kappa_{\ell,f} \, x_{\ell,f} \quad \forall \, a \in A, \, \forall \, [s, t]_a \in D(a). \tag{6.28}$$

We want to further improve this relaxation by adding to the left hand side of constraint (6.28) for $a \in A$ $[s, t]_a \in D(a)$ all OD pairs whose set of direct connection lines

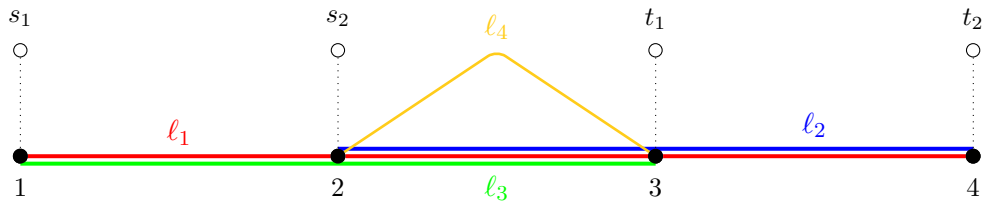


Figure 6.6: A transportation network with four stations $\{a, b, c, d\}$ and four lines $\ell_1 = (1, 2, 3, 4)$, $\ell_2 = (2, 3, 4)$, $\ell_3 = (1, 2, 3)$, and $\ell_4 = (2, 3)$. We have $\mathcal{F} = \{1\}$, $\kappa_{\ell,1} = 1$ for all ℓ, and a passenger demand of one from s_1 to t_1 and from s_2 to t_2.

on arc a is a subset of the direct connection lines on a for (s, t). We explain this idea again on Figure 6.5. Assume we have a further positive demand from s_1 to t_2. A direct connection s_1t_2-path is only supported by line ℓ_1, i. e., the set $\mathcal{L}_{s_1t_2}(2, 3) = \{\ell_1\}$ of direct connection lines for (s_1, t_2) w. r. t. arc $(2, 3)$ is included in the set $\mathcal{L}_{s_1t_1}(2, 3) = \{\ell_1, \ell_2\}$ of direct connection lines for (s_1, t_1) w. r. t. arc $(2, 3)$. The number of direct travelers from s_1 to t_2 can then be included in the left hand side of constraints (6.28) for arc $(2, 3)$ and equivalence class $[s_1, t_1]_{(2,3)}$. We then get

$$y_{(s_1,1,2,3,4,t_1),0} + y_{(s_1,1,2,3,6,t_1),0} + y_{(s_2,2,3,4,t_1),0} + y_{(s_2,2,3,6,t_1),0}$$
$$+y_{(s_1,1,2,3,4,t_2),0} + y_{(s_1,1,2,3,4,5,t_2),0} \leq \sum_{f\in\mathcal{F}} (\kappa_{\ell_1,f} x_{\ell_1,f} + \kappa_{\ell_2,f} x_{\ell_2,f}).$$

We say that OD-pair (u, v) is *dc-dominated with respect to arc a* by OD-pair (s, t) if $\mathcal{L}_{uv}(a) \subseteq \mathcal{L}_{st}(a)$. Denote by $[s, t]_a^{\leq}$ the corresponding domination set, i. e., $(u, v) \in [s, t]_a^{\leq}$ if $\mathcal{L}_{uv}(a) \subseteq \mathcal{L}_{st}(a)$. Then the dcpath variables $y_{p,0}$ for the number of direct travelers on path p must satisfy the following *direct connection constraints* for each arc a and each class $[s, t]_a$ of equivalent OD-pairs:

$$\sum_{(u,v)\in[s,t]_a^{\leq}} \sum_{p\in\mathcal{P}^0_{uv}(a)} y_{p,0} \leq \sum_{\ell\in\mathcal{L}_{st}(a)} \sum_{f\in\mathcal{F}} \kappa_{\ell,f}\, x_{\ell,f} \qquad \forall\, a \in A,\, \forall\, [s, t]_a \in D(a). \tag{6.29}$$

These constraints enforce sufficient transportation capacity to route all uv-dcpaths, $(u, v) \in [s, t]_a^{\leq}$, via arc a. That this is still an approximation can be seen in Figure 6.6. In this example the OD pairs (s_1, t_1) and (s_2, t_2) are not dc-equivalent w. r. t. arc $(2, 3)$ nor is one dc-dominated by the other; namely, we have $\mathcal{L}_{s_1t_1}(2, 3) = \{\ell_1, \ell_3\}$ and $\mathcal{L}_{s_2t_2}(b, c) = \{\ell_1, \ell_2\}$. Line ℓ_1 is a direct connection line for both OD pairs. Setting $x_{\ell_1,1} = x_{\ell_4,1} = 1$ and $x_{\ell_2,1} = x_{\ell_3,1} = 0$ satisfies the direct connection constraints (6.29) and implies enough capacity to transport all passengers. But either the passenger from a to c or the one from b to d has to transfer, i. e., the number of direct travelers is overestimated.

We finally obtain the following *relaxed line connection model* (RLC) by using variables $y_{p,0}$ instead of $z^{\ell}_{p,0}$ and substituting the direct line connection constraints (6.15) by the

direct connection constraints (6.29).

$$\text{(RLC)} \quad \min \lambda \sum_{\ell \in \mathcal{L}} \sum_{f \in \mathcal{F}} c_{\ell,f}\, x_{\ell,f} + (1-\lambda)\Big(\sum_{p \in \mathcal{P}^0} \tau_{p,0}\, y_{p,0} + \sum_{p \in \mathcal{P}} \tau_{p,1}\, y_{p,1} \Big)$$

$$\sum_{p \in \mathcal{P}^0_{st}} y_{p,0} + \sum_{p \in \mathcal{P}_{st}} y_{p,1} = d_{st} \qquad \forall\, (s,t) \in D \tag{6.30}$$

$$\sum_{p \in \mathcal{P}^0 : a \in p} y_{p,0} + \sum_{p \in \mathcal{P} : a \in p} y_{p,1} \leq \sum_{\ell \in \mathcal{L} : e(a) \in \ell} \sum_{f \in \mathcal{F}} \kappa_{\ell,f}\, x_{\ell,f} \qquad \forall\, a \in A \tag{6.31}$$

$$\sum_{(u,v) \in [s,t]_a^{\leq}} \sum_{p \in \mathcal{P}^0_{uv}(a)} y_{p,0} \leq \sum_{\ell \in \mathcal{L}_{st}(a)} \sum_{f \in \mathcal{F}} \kappa_{\ell,f}\, x_{\ell,f} \qquad \forall\, a \in A,\, \forall\, [s,t]_a \in D(a) \tag{6.29}$$

$$\sum_{f \in \mathcal{F}} x_{\ell,f} \leq 1 \qquad \forall\, \ell \in \mathcal{L} \tag{6.32}$$

$$\sum_{\ell \in \mathcal{L} : e(a) \in \ell} \sum_{f \in \mathcal{F}} f \cdot x_{\ell,f} \leq F_e \qquad \forall\, e \in E \tag{6.33}$$

$$x_{\ell,f} \in \{0,1\} \qquad \forall\, \ell \in \mathcal{L},\, \forall\, f \in \mathcal{F} \tag{6.34}$$

$$y_{p,0} \geq 0 \qquad \forall\, p \in \mathcal{P}^0 \tag{6.35}$$

$$y_{p,1} \geq 0 \qquad \forall\, p \in \mathcal{P}. \tag{6.36}$$

Model (RLC) minimizes a weighted sum of line operating costs and passenger travel times. As in model (DLC) the st-passenger path variables $y_{p,1}$ incur a penalty for each transfer arc and exactly one transfer penalty otherwise. Equations (6.30) enforce the passenger flow. Inequalities (6.31) guarantee sufficient total transportation capacity on each arc. Constraints (6.29), the direct connection constraints, approximate the sufficiency of transportation capacity for direct connection passenger paths on each arc. Inequalities (6.32) ensure that a line is operated at one frequency at most and inequalities (6.33) bound the sum of the line operation frequencies for each edge.

The size of model (RLC) is smaller than the size of model (DLC). Model (RLC) has a smaller number of variables since it relaxes the assignment of passenger paths to particular lines. The number of direct connection constraints is also in general smaller than the number of direct line connection constraints since the number of OD pairs is in general smaller than the number of lines. A problem with model (RLC) is that the sets $\mathcal{P}^0$, $\mathcal{P}^0_{st}$, and $\mathcal{P}^0_{st}(a)$ are all defined via dcpath sets $\{\mathcal{P}^{0,\ell}_{st},\, \ell \in \mathcal{L}\}$ that are in turn defined in terms of individual lines, i. e., we have $p \in \mathcal{P}^0_{st}$ if there exists an $\ell \in \mathcal{L}$ such that $p \in \mathcal{P}^{0,\ell}_{st}$. This means that the pricing problem for the direct connection passenger path variables is hard to handle without looping over all lines, i. e., (RLC) is algorithmically as hard as (DLC), even though it is more compact. In the following section we will propose a relaxation of the sets $\mathcal{P}^0$, $\mathcal{P}^0_{st}$, and $\mathcal{P}^0_{st}(a)$ to overcome this problem in the *direct connection model*. In this relaxation the specific routes of direct st-connection lines are not important. Rather it is only relevant which edges in the public transport network are covered by these direct st-connection lines. In this way, the effort for pricing the direct connection passenger path variables is reduced since we do not have to loop over lines anymore.

6.5.3 Approximation Model II – Relaxing Direct Connections

Model (RLC) substitutes the variables $z^{\ell}_{p,0}$ by aggregated variables $y_{p,0}$ and relaxes the direct line connection constraints. We will now propose an approximation of model (RLC) that further relaxes the set of direct connection passenger paths in such a way that an efficient pricing becomes possible. To this purpose, consider for each OD-pair $(s,t) \in D$ the set $\mathcal{P}^0_{st} = \bigcup_{\ell \in \mathcal{L}_{st}} \mathcal{P}^{0,\ell}_{st}$ of all st-dcpaths and unite them to construct what we call a *direct connection st-passenger routing graph* $G_{st} = (V_{st}, A_{st}) = \bigcup_{p \in \mathcal{P}^0_{st}} (V(p), A(p))$, where $V(p)$ and $A(p)$ denote the nodes and arcs of dcpath p, respectively, see Figure 6.7 for an example. Note that G_{st} can be constructed in polynomial time. We proceed by considering *all* st-paths in G_{st} as *relaxed st-dcpaths* (st-rdcpaths); let $\mathcal{P}^{0+}_{st}$ be the set of all such st-rdcpaths, $\mathcal{P}^{0+}_{st}(a) = \{p \in \mathcal{P}^{0+}_{st} : a \in p\}$ the set of all st-rdcpaths via arc a, and $\mathcal{P}^{0+} = \bigcup_{(s,t)\in D} \mathcal{P}^{0+}_{st}$ the set of all rdcpaths. Obviously, $\mathcal{P}^{0+}_{st} \supseteq \mathcal{P}^0_{st}$, i. e., $\mathcal{P}^{0+}_{st}$ overestimates the number of direct connections between origin s and destination t, see again Figure 6.7. Replacing all sets of direct connection passenger paths by the corresponding sets of relaxed direct connection passenger paths in model (RLC) yields the following *direct connection model* (DC)

$$
\begin{aligned}
\text{(DC)} \quad \min\ & \lambda \sum_{\ell \in \mathcal{L}} \sum_{f \in \mathcal{F}} c_{\ell,f}\, x_{\ell,f} + (1-\lambda)\Big(\sum_{p \in \mathcal{P}^{0+}} \tau_{p,0}\, y_{p,0} + \sum_{p \in \mathcal{P}} \tau_{p,1}\, y_{p,1} \Big) \\
& \sum_{p \in \mathcal{P}^{0+}_{st}} y_{p,0} + \sum_{p \in \mathcal{P}_{st}} y_{p,1} = d_{st} && \forall\, (s,t) \in D && (6.37)\\
& \sum_{p \in \mathcal{P}^{0+} : a \in p} y_{p,0} + \sum_{p \in \mathcal{P} : a \in p} y_{p,1} \le \sum_{\ell \in \mathcal{L} : e(a) \in \ell} \sum_{f \in \mathcal{F}} \kappa_{\ell,f}\, x_{\ell,f} && \forall\, a \in A && (6.38)\\
& \sum_{(u,v) \in [s,t]^{\le}_a} \sum_{p \in \mathcal{P}^{0+}_{uv}(a)} y_{p,0} \le \sum_{\ell \in \mathcal{L}_{st}(a)} \sum_{f \in \mathcal{F}} \kappa_{\ell,f}\, x_{\ell,f} && \forall\, a \in A,\ \forall\, [s,t]_a \in D(a) && (6.39)\\
& \sum_{f \in \mathcal{F}} x_{\ell,f} \le 1 && \forall\, \ell \in \mathcal{L} && (6.40)\\
& \sum_{\ell \in \mathcal{L} : e(a) \in \ell} \sum_{f \in \mathcal{F}} f \cdot x_{\ell,f} \le F_e && \forall\, e \in E && (6.41)\\
& x_{\ell,f} \in \{0,1\} && \forall\, \ell \in \mathcal{L},\ \forall\, f \in \mathcal{F} && (6.42)\\
& y_{p,0} \ge 0 && \forall\, p \in \mathcal{P}^{0+} && (6.43)\\
& y_{p,1} \ge 0 && \forall\, p \in \mathcal{P}. && (6.44)
\end{aligned}
$$

Model (DC) is a relaxation of model (RLC). It only differs from model (RLC) by considering the (possibly bigger) set of rdcpaths $\mathcal{P}^{0+}$ instead of the set of dcpaths $\mathcal{P}^0$. Model (DC) is, hence, also a relaxation of model (DLC). The advantage of model (DC) compared to model (RLC) is an improved tractability since the pricing problem for the relaxed direct connection passenger path variables for an OD pair (s,t) is just a shortest path problem in the direct connection st-passenger routing graph while the pricing problem for model (RLC) involves an enumeration of all direct connection lines, see Subsection 6.5.5.

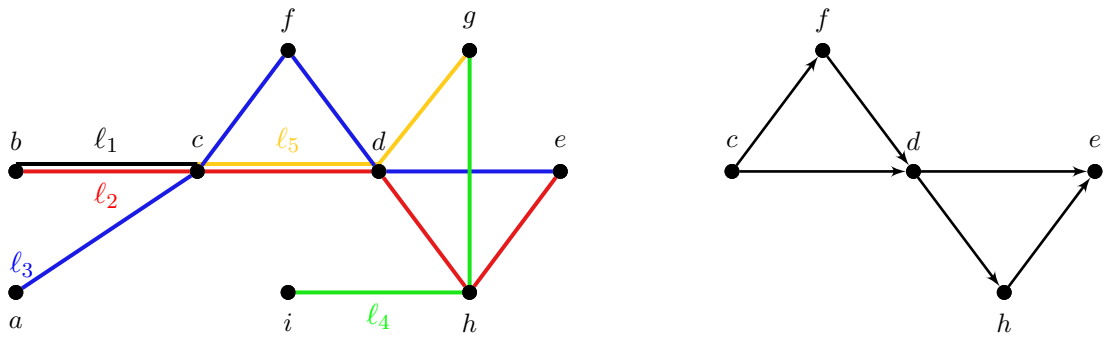


Figure 6.7: *Left:* A transportation network with nine stations and five lines $\ell_1 = (b, c)$, $\ell_2 = (b, c, d, h, e)$, $\ell_3 = (a, c, f, d, e)$, $\ell_4 = (i, h, g)$, and $\ell_5 = (c, d, g)$. There are two ce-dcpaths (c, f, d, e) and (c, d, h, e). *Right:* A direct connection ce-passenger routing graph. The path $p = (c, d, e)$ is an ec-rdcpath but not an ec-dcpath, i. e., $p = (c, d, e) \in \mathcal{P}^{0+}_{ce}$ but $p = (c, d, e) \notin \mathcal{P}^{0}_{ce}$.

6.5.4 Model Extension – Unavoidable Transfers

The set of dcpaths $\mathcal{P}^0$ represents all paths in the network that are covered by direct connection lines. Model (RLC) associates with each other path in the set $\mathcal{P} \setminus \mathcal{P}^0$ *exactly* one transfer (if they contain no explicit transfer arcs). In the following we want to improve this approach by accounting for the number of *unavoidable transfers* k_p w. r. t. the set of lines $\mathcal{L}$. The number k_p indicates for each path $p \in \mathcal{P}$ the minimum number of transfers a passenger has to do with respect to the predefined set of lines. More precisely, considering a certain passenger path, it may not be possible to cover this path by a single line or even by two lines, i. e., in any definition of a line plan, passengers on the path under consideration have to transfer at least once or twice, respectively. We call such transfers *unavoidable*. We consider three types of variables:

- A variable y_{p,k_p} for all $p \in \mathcal{P} \setminus \mathcal{P}^0$, i. e., we associate non-direct connection passenger paths with its number of unavoidable transfers.
- A variable $y_{p,0}$, for all $p \in \mathcal{P}^0$, i. e., we have a variable associating direct connection passenger paths with 0 transfers.
- A variable $y_{p,1}$, for all $p \in \mathcal{P}^0$, i. e., we have a variable associating direct connection passenger paths with 1 transfer if the capacity of direct connection lines does not suffice.

This approach needs a slight change in the notation. Namely, we associate with τ_a on all transfer arcs $a \in A_T$ just the travel or walking time and do not incorporate the transfer penalty. Each path $p \in \mathcal{P} \setminus \mathcal{P}^0$ associated with k_p unavoidable transfers gets travel and transfer time $\tau_{p,k_p} = \tau_p + k_p\sigma = \sum_{a \in p} \tau_a + k_p\sigma$. Note that one can identify $y_{p,0}$ and $y_{p,1}$, $p \in \mathcal{P}$, with variables y_{p,k_p} and y_{p,k_p+1}, respectively. The integer programming formulation of the *direct connection model with unavoidable transfers* (UT) is

$$\text{(UT)} \quad \min \lambda \sum_{\ell\in\mathcal{L}} \sum_{f\in\mathcal{F}} c_{\ell,f}\, x_{\ell,f} + (1-\lambda)\Big(\sum_{p\in\mathcal{P}^0} (\tau_{p,1}\, y_{p,1} + \tau_{p,0}\, y_{p,0}) + \sum_{p\in\mathcal{P}\setminus\mathcal{P}^0} \tau_{p,k_p}\, y_{p,k_p}\Big)$$

$$\sum_{p\in\mathcal{P}^0_{st}} (y_{p,0} + y_{p,1}) + \sum_{p\in\mathcal{P}_{st}\setminus\mathcal{P}^0_{st}} y_{p,k_p} = d_{st} \qquad \forall\, (s,t) \in D \tag{6.45}$$

$$\sum_{p\in\mathcal{P}^0:a\in p} (y_{p,0} + y_{p,1}) + \sum_{p\in\mathcal{P}\setminus\mathcal{P}^0:a\in p} y_{p,k_p} \le \sum_{\ell\in\mathcal{L}:e(a)\in\ell} \sum_{f\in\mathcal{F}} \kappa_{\ell,f}\, x_{\ell,f} \qquad \forall\, a \in A \tag{6.46}$$

$$\sum_{(u,v)\in[s,t]_a^{\le}} \sum_{p\in\mathcal{P}^0_{uv}(a)} y_{p,0} \le \sum_{\ell\in\mathcal{L}_{st}(a)} \sum_{f\in\mathcal{F}} \kappa_{\ell,f}\, x_{\ell,f} \qquad \forall\, a \in A,\, \forall\, [s,t]_a \in D(a) \tag{6.47}$$

$$\sum_{f\in\mathcal{F}} x_{\ell,f} \le 1 \qquad \forall\, \ell \in \mathcal{L} \tag{6.48}$$

$$\sum_{\ell\in\mathcal{L}:e(a)\in\ell} \sum_{f\in\mathcal{F}} f \cdot x_{\ell,f} \le F \qquad \forall\, e \in E \tag{6.49}$$

$$x_{\ell,f} \in \{0,1\} \qquad \forall\, \ell \in \mathcal{L},\, \forall\, f \in \mathcal{F} \tag{6.50}$$

$$y_{p,0}, y_{p,1} \ge 0 \qquad \forall\, p \in \mathcal{P}^0 \tag{6.51}$$

$$y_{p,k_p} \ge 0 \qquad \forall\, p \in \mathcal{P}\setminus\mathcal{P}^0. \tag{6.52}$$

Model (UT) minimizes a weighted sum of line operating costs and passenger travel times. The passenger path variables y_{p,k_p} incur a penalty for each unavoidable transfer (including all transfer arcs). The passenger path variables $y_{p,1}$ incur a penalty for one transfer caused by insufficient capacity of direct connection lines. Equations (6.45) enforce the passenger flow. Inequalities (6.46) guarantee sufficient total transportation capacity on each arc. Analogous to model (RLC) the direct connection constraints (6.47) approximate the sufficiency of transportation capacity for direct connection passenger paths on each arc. Inequalities (6.48) ensure that a line is operated at one frequency at most and inequalities (6.49) bound the sum of the line operation frequencies for each edge.

Example 6.6. *Consider again the network given in the left of Figure 6.7. The given set of lines is* $\mathcal{L} = \{\ell_1, \ell_2, \ell_3, \ell_4, \ell_5\}$. *The path* $p_1 = (b, c, d)$ *is a bd-dcpath since it is covered by line* ℓ_2. *It, hence, has* $k_{p_1} = 0$ *unavoidable transfers. We consider the variables* $y_{p_1,0}$ *and* $y_{p_1,1}$ *as in model (RLC). The path* $p_2 = (c, d, e)$ *has* $k_{p_2} = 1$ *unavoidable transfer, i. e., in model (UT) we consider a variable* $y_{p_2,k_{p_2}} = y_{p_2,1}$. *So far, model (UT) would not improve model (RLC). However, path* $p_3 = (i, h, d, f)$ *has* $k_{p_3} = 2$ *unavoidable transfers. In model (UT) we consider the variable* $y_{p_3,k_{p_3}} = y_{p_3,2}$, *i. e., this path is associated with 2 transfers, in model (RLC) it is associated with only one transfer.*

Algorithm 6.8 computes a travel-time minimal path from a given node $s \in V$ to all other nodes including a uniform transfer penalty $\sigma \in \mathbb{Q}_+$ for each transfer w. r. t. a given set of lines $\mathcal{L}$. The input of the algorithm is a directed graph $G = (V, A)$, nonnegative arc weights τ_a, $a \in A$, a transfer penalty σ, a start node s, and a set of lines $\mathcal{L}$, i. e., simple paths in G. Let $\ell \in \mathcal{L}$. We denote by $q_\ell(u, v)$ the subpath $q \subseteq \ell$ and $u, v \in V$ being the start and end node of q and by $dist_\ell(u, v) = \sum_{a\in q_{\ell(u,v)}} \tau_a$ the travel time of this path. The distance, i. e., the travel time plus transfer penalties from s to a node $v \in V$ is stored

Algorithm 6.8: A modified Dijkstra algorithm to compute shortest paths from s to all other nodes including transfer penalties with respect to $\mathcal{L}$.

Input : A connected graph $G = (V, A)$ with arc weights $\tau_a \in \mathbb{Q}_{\geq 0}$, $a \in A$, a transfer penalty $\sigma \in \mathbb{Q}_{\geq 0}$, a start node s, a set of lines $\mathcal{L}$.

Output: Shortest paths including transfers from s to all other nodes.

$d(s) := 0$, $d(v) := \min\{\text{dist}_\ell(s, v) : s, v \in \ell, \ell \in \mathcal{L}\}$ $\forall\, v \in V \setminus \{s\}$,

$tr(v) := \begin{cases} 0 & d(v) < \infty \\ \infty & \text{otherwise} \end{cases}$ $\forall\, v \in V$, mark s, all other nodes are unmarked.

while exists unmarked node v **do**
  Choose v with $d(v) = \min\{d(w), w \text{ unmarked}\}$
  for all $\ell \in \mathcal{L}$ with $v \in \ell$ **do**
    for all unmarked w with $w \in \ell$ **do**
      if $d(v) + \sigma + \text{dist}_\ell(v, w) < d(w)$ **then**
        $d(w) := d(v) + \sigma + \text{dist}_\ell(v, w)$
        $tr(w) := tr(v) + 1$
        $p(w) := q^\ell(v, w)$
      end
    end
  end
  mark v
end

in $d(v)$. The distance for the start node is 0. The distance for all other nodes $v \in V \setminus \{s\}$ is initialized by the length of a shortest path from s to v covered by a direct sv-connection line. Here, we assume that $\min \emptyset = \infty$, i. e., all nodes that cannot be reached by a direct connection line from s get an initial distance of infinity. The algorithm works similar to Dijkstra's algorithm. In each step an unmarked node v with minimum distance is considered. The weights for all (unmarked) nodes w that can be reached from v via a line are updated if the travel time from v to w plus one transfer penalty plus the distance from s to v is smaller than the distance from s to w considered so far. Then the node v is marked, $d(v)$ is the minimum travel time plus transfer penalties from s to v, $tr(v)$ gives the number of transfers, and $p(v)$ the last transfer free part of the path from s to v. Note that the algorithm is stated in a simplified form that ignores transfer arcs and OD arcs. The inclusion of those arcs requires just some technical effort, which we skip for ease of exposition.

Algorithm 6.8 reminds of Algorithm 3.6 of Section 3.3. Setting $c \equiv 1$, Algorithm 3.6 can be interpreted as computing the minimum number of paths/lines that connect the nodes $s \in V$ and $v \in V$, i. e., the minimum number of unavoidable transfers from s to v plus 1. Conversely, replacing $\text{dist}_\ell(v, w)$ for each $v, w \in \ell$ by the cost c_ℓ of line ℓ and setting $\sigma = 0$, Algorithm 6.8 computes the same distance labels $d(v)$, $v \in V$, as Algorithm 3.6 with $\mathcal{L} = \mathcal{P}$.

Lemma 6.7. *Algorithm 6.8 computes a shortest path from node s to all other nodes including a transfer penalty for all transfers with respect to the line pool $\mathcal{L}$. The running time is $\mathcal{O}(|V|\log|V| + |\mathcal{L}||V|^2)$.*

6.5.5 Pricing Problems

In the following, we consider the passenger path pricing problems for the three direct connection models (DLC), (DC), and (UT). We skip the intermediate model (RLC); the pricing for this model can be handled in a similar way as for model (UT).

Direct Line Connection Model

Associate dual variables π (unbounded), $\mu \geq 0$, $\nu \geq 0$, $\psi \geq 0$, and $\eta \geq 0$ with constraints (6.13), (6.14), (6.15), (6.16), and (6.17) of program (DLC). The dual of the LP relaxation of (DLC) is

$$\begin{aligned}
\max \sum_{(s,t)\in D} d_{st}\pi_{st} - \sum_{\ell\in\mathcal{L}} \psi_\ell - \sum_{e\in E} F_e\eta_e & \\
\pi_{st} - \sum_{a\in p} \mu_a - \sum_{a\in p} \nu_{\ell,a} \leq (1-\lambda)\tau_{p,0} & \quad \forall\, \ell\in\mathcal{L},\, \forall\,(s,t)\in D,\, \forall\, p\in\mathcal{P}_{st}^{0,\ell} \\
\pi_{st} - \sum_{a\in p} \mu_a \leq (1-\lambda)\tau_{p,1} & \quad \forall\,(s,t)\in D,\, \forall\, p\in\mathcal{P}_{st} \\
\kappa_{\ell,f} \sum_{a:e(a)\in\ell} (\mu_a + \nu_{\ell,a}) - \psi_\ell - f\sum_{e\in\ell} \eta_e \leq \lambda c_{\ell,f} & \quad \forall\,\ell\in\mathcal{L},\, \forall\, f\in\mathcal{F} \\
\mu_a \geq 0 & \quad \forall\, a\in A \\
\nu_{\ell,a} \geq 0 & \quad \forall \ell\in\mathcal{L},\, \forall\, a\in A : e(a)\in\ell \\
\psi_\ell \geq 0 & \quad \forall\,\ell\in\mathcal{L} \\
\eta_e \geq 0 & \quad \forall\, e\in E.
\end{aligned}$$

The pricing problem for the passenger variables is twofold: Find an st-dcpath with negative reduced cost or find a path from s to t with at least one transfer and negative reduced cost. The reduced costs can be computed as follows

$$\bar{\tau}_{p,0} = -\pi_{st} + \sum_{a\in p} (\mu_a + \nu_{\ell,a} + (1-\lambda)\tau_a)\,, \qquad p\in\mathcal{P}_{st}^{0,\ell} \tag{6.53}$$

$$\bar{\tau}_{p,1} = -\pi_{st} + \sum_{a\in p} (\mu_a + (1-\lambda)\tau_a) + (1-\lambda)\sigma(p), \qquad p\in\mathcal{P}_{st}. \tag{6.54}$$

The first case requires to consider each direct connection line individually since the arc weights depend on the line. More precisely, for each line $\ell\in\mathcal{L}_{st}$ one has to find a weight minimal path $p\in\mathcal{P}_{st}^{0,\ell}$ with arc weights set to $\mu_a + \nu_{\ell,a} + (1-\lambda)\tau_a$. If the weight of the path is less than π_{st}, the path is added to the model. The arc weights depend on $\ell\in\mathcal{L}_{st}$, i. e., we have to consider each direct connection line for OD pair (s,t) with all

its edges. For a given line ℓ the set $|\mathcal{P}_{st}^{0,\ell}|$ is usually very small, in particular, $|\mathcal{P}_{st}^{0,\ell}| = 1$, $s, t \in V_O$, if $|\delta^+(s)| = |\delta^-(t)| = 1$, i. e., the path $p \in \mathcal{P}_{st}^{0,\ell}$ is usually implied by ℓ. We can assume a running time of $\mathcal{O}(|\mathcal{L}_{st}||V_{st}|)$ to find an st-dcpath with negative reduced cost or to conclude that no such path exists. If all direct connection st-lines have parallel routes between s and t this bound is strict. Recall that V_{st} is the set of nodes in the direct connection st-passenger routing graph, which was defined in Subsection 6.5.3.

In the second case we have to distinguish whether p contains a transfer arc or not. This can be done by a shortest path algorithm with two labels for each node. We have to compute (i) a shortest path p with $p \cap A_T = \emptyset$ and (ii) a shortest path p with $p \cap A_T \neq \emptyset$. The arc weights are set to $\omega_a = \mu_a + (1-\lambda)\tau_a \geq 0$ for $a \in A$ in both cases. Only in case (i) we have to add $(1-\lambda)\sigma$ to the weight of the path. If we have found, in case (i) or case (ii), a path with weight smaller then π_{st}, the associated variable has to be added to the problem.

Proposition 6.8. *The pricing problem for the passenger path variables in model (DLC) can be solved in polynomial time. In particular, finding an st-dcpath with negative reduced cost or concluding that no such path exists, can be done in $\mathcal{O}(|\mathcal{L}_{st}||V_{st}|)$. Finding an st-path with at least one transfer or concluding that no such path exists, needs $\mathcal{O}(|V|\log|V|+|A|)$ time.*

Direct Connection Model

Consider the solution of the LP relaxation of model (DC) by column generation. Associate dual variables π (unbounded), $\mu \geq 0$, $\nu \geq 0$, $\psi \geq 0$, and $\eta \geq 0$ with constraints (6.37), (6.38), (6.39), (6.40), and (6.41) of program (DC). The dual of the LP relaxation of (DC) is

$$\begin{aligned}
\max \sum_{(s,t)\in D} d_{st}\pi_{st} - \sum_{\ell\in\mathcal{L}} \psi_\ell - \sum_{e\in E} F_e\eta_e & \\
\pi_{st} - \sum_{a\in p} \mu_a - \sum_{a\in p} \nu_{a,[s,t]_a} \leq (1-\lambda)\tau_{p,0} & \quad \forall\,(s,t)\in D,\, \forall\, p \in \mathcal{P}_{st}^{0+} \\
\pi_{st} - \sum_{a\in p} \mu_a \leq (1-\lambda)\tau_{p,1} & \quad \forall\,(s,t)\in D,\, \forall\, p \in \mathcal{P}_{st} \\
\kappa_{\ell,f} \sum_{a:e(a)\in\ell} \Big(\mu_a + \sum_{[s,t]_a\in D(a)} \nu_{a,[s,t]_a}\Big) - \psi_\ell - f\sum_{e\in\ell} \eta_e \leq \lambda c_{\ell,f} & \quad \forall\,\ell\in\mathcal{L},\, \forall\, f\in\mathcal{F} \\
\mu_a \geq 0 & \quad \forall\, a \in A \\
\nu_{a,[s,t]_a} \geq 0 & \quad \forall\, a\in A,\, \forall\,[s,t]_a \in D(a) \\
\psi_\ell \geq 0 & \quad \forall\,\ell\in\mathcal{L} \\
\eta_e \geq 0 & \quad \forall\, e\in E.
\end{aligned} \tag{6.55}$$

The pricing problem for the passenger path variables decomposes again into two cases. Find an st-rdcpath with negative reduced cost or find a path from s to t with at least

one transfer and negative reduced cost. The reduced costs can be computed as follows

$$\bar{\tau}_{p,0} = -\pi_{st} + \sum_{a \in p} \left(\mu_a + \nu_{a,[s,t]_a} + (1-\lambda)\tau_a\right), \qquad p \in \mathcal{P}_{st}^{0+} \tag{6.56}$$

$$\bar{\tau}_{p,1} = -\pi_{st} + \sum_{a \in p} \left(\mu_a + (1-\lambda)\tau_a\right) + (1-\lambda)\sigma(p), \qquad p \in \mathcal{P}_{st}^{0}. \tag{6.57}$$

The second case is analogous to the passenger path pricing problem for model (DLC). In the first case we have to find an st-rdcpath with weight smaller then π_{st}. This means that we have to solve exactly one shortest path problem in G_{st} for each OD pair $(s,t) \in D$. This can be done by Dijkstra's algorithm. The arc weights are set to $\omega_a = \mu_a + \nu_{a,[s,t]_a} + (1-\lambda)\tau_a \geq 0$ for $a \in A_{st}$, i. e., they are independent of a line.

Proposition 6.9. *The pricing problem for the passenger path variables in model (DC) can be solved in polynomial time. In particular, finding an st-rdcpath with negative reduced cost or concluding that no such path exists can be done in $\mathcal{O}(|A_{st}| + |V_{st}| \log |V_{st}|)$. The pricing problem for an st-path with at least one transfer needs $\mathcal{O}(|V| \log |V| + |A|)$ time.*

We often have $|\mathcal{L}_{st}| > |A_{st}|$ in our test instances in Chapter 8. Hence, the pricing problem for rdcpaths can be solved much faster than the pricing problem for dcpaths.

Direct Connection Model with Unavoidable Transfers

Associate dual variables π (unbounded), $\mu \geq 0$, $\nu \geq 0$, $\psi \geq 0$, and $\eta \geq 0$ with constraints (6.45), (6.46), (6.47), (6.48), and (6.49) of program (UT). The dual of the LP relaxation of (UT) is

$$\begin{aligned}
\max \sum_{(s,t) \in D} d_{st}\pi_{st} - \sum_{\ell \in \mathcal{L}} \psi_\ell - \sum_{e \in E} F_e \eta_e & \\
\pi_{st} - \sum_{a \in p} \mu_a - \sum_{a \in p} \nu_{a,[s,t]_a} \leq (1-\lambda)\tau_{p,0} & \quad \forall\, (s,t) \in D,\, \forall\, p \in \mathcal{P}_{st}^0 \\
\pi_{st} - \sum_{a \in p} \mu_a \leq (1-\lambda)\tau_{p,1} & \quad \forall\, (s,t) \in D,\, \forall\, p \in \mathcal{P}_{st}^0 \\
\pi_{st} - \sum_{a \in p} \mu_a \leq (1-\lambda)\tau_{p,k_p} & \quad \forall\, (s,t) \in D,\, \forall\, p \in \mathcal{P}_{st} \setminus \mathcal{P}_{st}^0 \\
\kappa_{\ell,f} \sum_{a : e(a) \in \ell} \Big(\mu_a + \sum_{[s,t]_a \in D(a)} \nu_{a,[s,t]_a}\Big) - \psi_\ell - f \sum_{e \in \ell} \eta_e \leq \lambda c_{\ell,f} & \quad \forall\, \ell \in \mathcal{L},\, \forall\, f \in \mathcal{F} \\
\mu_a \geq 0 & \quad \forall\, a \in A \\
\nu_{a,[s,t]_a} \geq 0 & \quad \forall\, a \in A,\, \forall\, [s,t]_a \in D(a) \\
\psi_\ell \geq 0 & \quad \forall\, \ell \in \mathcal{L} \\
\eta_e \geq 0 & \quad \forall\, e \in E.
\end{aligned}$$

Algorithm 6.9: A modified version of Algorithm 6.8 to compute a shortest path from s to t including a positive number of unavoidable transfers with respect to $\mathcal{L}$ or at least one transfer penalty.

Input : A connected graph $G = (V, A)$ with arc weights $\tau_a \geq 0$, $a \in A$, a transfer penalty σ, start node s, end node t, set of lines $\mathcal{L}$.

Output: Shortest paths including at least one transfer penalty from s to t.

$d(s) := 0$, $d(v) := \min\{\text{dist}_\ell(s, v) : s, v \in \ell, \ell \in \mathcal{L}\}\ \forall v \in V \backslash \{s\}$,

$tr(v) := \begin{cases} 0 & d(v) < \infty \\ \infty & \text{otherwise} \end{cases} \quad \forall v \in V,$ mark s, all other nodes are unmarked.

if $d(t) < \infty$ **then**
  $d(t) = d(t) + \sigma$
  $tr(t) = 1$
end
while exists unmarked node v and t is not marked **do**
  Choose v with $d(v) = \min\{d(w), w \text{ unmarked}\}$
  for all $\ell \in \mathcal{L}$ with $v \in \ell$ **do**
    for all unmarked w with $w \in \ell$ **do**
      if $d(v) + \sigma + \text{dist}_\ell(v, w) < d(w)$ **then**
        $d(w) = d(v) + \sigma + \text{dist}_\ell(v, w)$
        $tr(w) = tr(v) + 1$
        $p(w) = q^\ell(v, w)$
      end
    end
  end
  mark v
end

The reduced costs can be computed as follows

$$\bar{\tau}_{p,0} = -\pi_{st} + \sum_{a \in p} \left(\mu_a + \nu_{a,[s,t]_a} + (1-\lambda)\tau_a\right), \qquad p \in \mathcal{P}^0, \tag{6.58}$$

$$\bar{\tau}_{p,1} = -\pi_{st} + \sum_{a \in p} (\mu_a + (1-\lambda)\tau_a) + (1-\lambda)\sigma, \qquad p \in \mathcal{P}^0, \tag{6.59}$$

$$\bar{\tau}_{p,k_p} = -\pi_{st} + \sum_{a \in p} (\mu_a + (1-\lambda)\tau_a) + (1-\lambda)k_p\sigma, \qquad p \in \mathcal{P} \setminus \mathcal{P}^0 \ (k_p \geq 1). \tag{6.60}$$

We have three cases since we distinguish between $y_{p,0}$, $y_{p,1}$, $p \in \mathcal{P}^0$, and y_{p,k_p}, $p \in \mathcal{P} \setminus \mathcal{P}^0$. The second case, equation (6.59) can be interpreted as finding a path with one (unavoidable) transfer which is caused by insufficient capacity of direct connection lines.

In the first case, equation (6.58), we have to find an st-dcpath with weight smaller than π_{st}. The arc weights are set to $\omega_a = \mu_a + \nu_{a,[s,t]_a} + (1-\lambda)\tau_a \geq 0$ for $a \in A$. We can

solve this problem by the initialization step in line 1 of Algorithm 6.8 to find a shortest st-dcpath w. r. t. ω. If the weight of this path is less than π_{st}, the path is added to the model.

The second and third case can be handled simultaneously. We have to find a path with at least one (unavoidable) transfer. We can set the arc weights to $\mu_a + (1-\lambda)\tau_a \geq 0$ for $a \in A$, and use Algorithm 6.9 to compute a shortest path in combination with the number of unavoidable transfers. This algorithm is a slightly modified version of Algorithm 6.8. It adds a transfer penalty in the case that we can reach t from s directly in lines 2 to 5. This corresponds to a transfer caused by insufficient capacity of direct connection lines (the second case, equation (6.59)). The third case, equation (6.60), i. e., the case that t can be reached via a path p that is not covered by direct connection lines and has k_p unavoidable transfers, is handled in the while loop.

Proposition 6.10. *The pricing problem for the passenger path variables in model (UT) can be solved in polynomial time. In particular, the pricing problem for an st-dcpath can be solved in $\mathcal{O}(|\mathcal{L}_{st}||V_{st}|)$, while the pricing problem for an st-path with at least one unavoidable transfer can be solved in $\mathcal{O}(|V|\log|V| + |\mathcal{L}||V|^2)$.*

6.6 Model Discussion

We first relate models (DLC), (RLC), (DC), and (UT). Then we compare models (BD), (DC), and (CG).

To relate the models (DLC) and (DC), we show now that (DC) is a relaxation of the projection of model (DLC) onto the space of the dcpath variables. This can be seen as follows. For each st-dcpath $p \in \mathcal{P}^0_{st}$, link the flow variables $y_{p,0}$ and $z^\ell_{p,0}$ via equations

$$y_{p,0} = \sum_{\ell \in \mathcal{L}: p \in \mathcal{P}^{0,\ell}_{st}} z^\ell_{p,0}. \tag{6.61}$$

Consider the polytopes

$$\begin{aligned}
P &:= \{(x, y_1, z) \in \mathbb{R}^{(\mathcal{L}\times\mathcal{F})\times\mathcal{P}\times\mathcal{P}^{0,\mathcal{L}}}_{\geq 0} \mid \text{(DLC)}(6.13) - (6.17)\},\\
PQ &:= \{(x, y_0, y_1, z) \in \mathbb{R}^{(\mathcal{L}\times\mathcal{F})\times\mathcal{P}^0\times\mathcal{P}\times\mathcal{P}^{0,\mathcal{L}}}_{\geq 0} \mid (6.61), \text{(DLC)}(6.13) - (6.17)\},\\
Q &:= \{(x, y_0, y_1) \in \mathbb{R}^{(\mathcal{L}\times\mathcal{F})\times\mathcal{P}^0\times\mathcal{P}}_{\geq 0} \mid \exists z \in \mathbb{R}^{\mathcal{P}^{0,\mathcal{L}}}_{\geq 0} \text{ s.t. } (x, y_0, y_1, z) \in PQ\},
\end{aligned}$$

with $\mathcal{P}^{0,\mathcal{L}} = \dot\cup_{\ell\in\mathcal{L}}\mathcal{P}^{0,\ell}_{st}$, i. e., a direct connection path is indexed with each possible direct connection line. P is the solution set of the LP relaxation of (DLC). PQ extends this set into a higher-dimensional space by adding the aggregated flow variables $(y_{p,0})$; hence, P is the projection of PQ onto the space of (x, y_1, z) variables. Q is the projection of PQ onto the space of (x, y_0, y_1) variables, i. e., Q describes exactly the feasible combinations of line plans and aggregated direct connection passenger flows.

Let $Q = \{Ax + By \leq b\}$; then adding constraints $Ax + By \leq b$ to model (RLC) produces a strengthening of this model that is equivalent to the direct line connection model (DLC), i. e., that handles all direct connections correctly. Note that the cuts in the system $Ax + By \leq b$ can be separated using Benders decomposition, i. e., this construction is algorithmic. Both models, (RLC) and (DC), replace the Benders cut system $Ax + By \leq b$ by the smaller, explicit, and purely combinatorial set of direct connection constraints (6.29) and (6.39), respectively. Model (DC) further uses rdcpaths instead of dcpaths, i. e., it considers a larger set of paths $\mathcal{P}_{st}^{0+} \supseteq \mathcal{P}_{st}^{0}$. This makes model (DC) algorithmically tractable. Indeed, considering the LP relaxation, we have shown that the pricing problem for passenger path variables is a shortest path problem in G_{st} for relaxed direct connection passenger paths, and a (polynomially solvable) constrained shortest path problem in G for paths with at least one transfer.

Models (DLC), (RLC), and (DC) can be seen as a "first order approximation" to model (CG): Models (DLC), (RLC), and (DC) do not consider transfer penalties for the second, third, etc., transfer in a passenger path that cannot be attributed to a transfer arc. Model (UT) provides an approximation of higher order by additionally accounting for unavoidable transfers. All models except (DLC) further relax the assignment of direct connection paths to particular lines. They can also be seen as a "transfer improvement" of model (BD) since (BD) arises from (DC) by dropping the direct connection constraints and (DLC), (RLC), and (UT) are more detailed than (DC). Model (UT) extends the idea behind model (RLC) toward multiple transfers. It treats direct connections in the same way as model (RLC), but it further incorporates unavoidable transfers. Adding the constraints $Ax + By \leq b$ to model (UT) produces a strengthening of the direct line connection model (DLC) that considers possible second, third, etc., unavoidable transfers.

Let us denote by $v_R(M)$ the optimal objective value of relaxation R of an integer programming model M. Considering the IP values and the LP relaxation values of all defined models, we, finally, get the following picture:

Proposition 6.11. *Model (DC) dominates model (BD) as an integer program and with respect to the LP relaxation. Model (CG) dominates models (DLC) and (UT) as an integer program and with respect to the LP relaxation. Models (DLC) and (UT) dominate model (RLC) (which dominates (DC)) as an integer program and with respect to the LP relaxation. We, therefore, get*

$$v_{IP}(\text{CG}) \geq \left\{ \begin{array}{c} v_{IP}(\text{DLC}) \\ v_{IP}(\text{UT}) \end{array} \right\} \geq v_{IP}(\text{RLC}) \geq v_{IP}(\text{DC}) \geq v_{IP}(\text{BD}),$$

$$v_{LP}(\text{CG}) \geq \left\{ \begin{array}{c} v_{LP}(\text{DLC}) \\ v_{LP}(\text{UT}) \end{array} \right\} \geq v_{LP}(\text{RLC}) \geq v_{LP}(\text{DC}) \geq v_{LP}(\text{BD}).$$

□

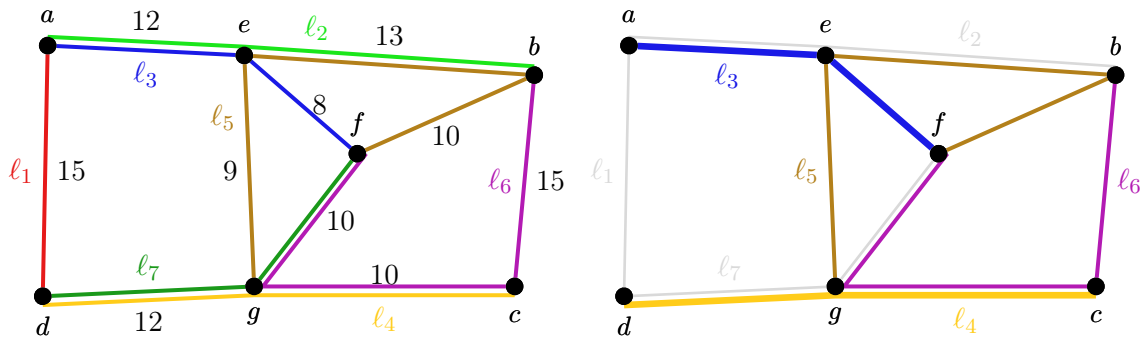


Figure 6.8: *Left:* Public transport network with seven stops and seven lines. *Right:* Feasible line plan for demand given in Table 6.1.

Table 6.1: Comparing line planning models according to the inclusion of a transfer penalty for passenger paths. The first column gives the travel demand for the instance in Figure 6.8. The second and third columns indicate possible passenger paths to fulfill the demand. In the last four columns it is marked whether the considered model associates a transfer penalty with the considered passenger paths for the line plan given in the right of Figure 6.8.

				transfer penalty added by model			
demand	path	travel time	no. trans.	(BD)	(DC)	(DLC)	(UT)
$a \to f$ 50	$p_1 = (a, e, f)$	20	0	0	0	0	0
$a \to b$ 50	$p_2 = (a, e, b)$	25	1	0	1	1	1
$d \to f$ 20	$p_3 = (d, g, f)$	22	1	0	1	1	1
$d \to c$ 80	$p_4 = (d, g, c)$	22	0	0	0	0	0
$g \to b$ 40	$p_5 = (g, f, b)$	20	1	0	0	1	1
$c \to a$ 30	$p_6 = (c, g, e, a)$	31	2	0	1	1	2

In the following example we compare the models according to the transfer penalties that are included.

Example 6.12. *The left of Figure 6.8 revisits the Example 6.1. Recall that the capacity of a line is $\kappa_\ell = 50$ for all $\ell \in \{\ell_1, \dots, \ell_7\}$. The possible frequencies are $\mathcal{F} = \{1, 2\}$. The right of Figure 6.8 shows a feasible solution for the demand given in the first column of Table 6.1. In this solution lines ℓ_3, ℓ_4 are operated with frequency 2 and ℓ_5, ℓ_6 with frequency 1. Note that a line provides the capacity in both directions on all edges it is operated. Table 6.1 also gives the possible passenger paths (column 2) with pure travel times (column 3), i. e., without transfer penalties, and number of transfers (column 4). The public transport network features only one transportation mode, i. e., model (BD) does not include any transfer penalty. It is not stated in the table but model (CG) includes a transfer penalty for each transfer. Model (DC) cannot rule out path p_5 as a direct connection path because each edge of the path is covered by a direct connection line (edge $\{g, f\}$ by ℓ_6 and edge $\{f, b\}$ by ℓ_5), so the direct connection constraints are satisfied; but there is no direct connection line covering all edges of this path. For each other non-direct connection path one transfer penalty is included in model (DC). Model (DLC) considers each direct connection path correctly since in model (DLC) each direct connection pas-*

senger path is associated with a direct connection line. All other paths are associated with one transfer penalty. Model (UT) detects that p_5 is not a direct connection path because the line pool contains no line $\tilde{\ell}$ containing edges $\{g, f\}$ and $\{f, b\}$. If the line pool would contain $\tilde{\ell}$ and the solution would be as in the right of Figure 6.8, model (UT) would also consider path p_5 as a direct connection path. Moreover, model (UT) accounts for two transfer penalties for path $p_6 = (c, g, e, a)$ since the whole line pool yields no possibility to cover this path by at most two lines. The last four columns of Table 6.1 summarize the inclusion of transfer penalties for the four models.

We have seen that model (CG) includes a complete treatment of transfers. However, computational experiments show that the change-and-go model is hard to handle. Namely, the LP relaxation for only half of our test instances could be solved within 5 hours; the LP relaxation of the other instances are even not solved after 10 hours, see Chapter 8. The basic dynamic model (BD), on the other hand, can be solved quite efficiently. The integrality gaps are close to 0, and some instances can even be solved to optimality within 10 hours, see Chapter 8. However, the example above shows that transfers are often not accounted in model (BD). The transfer handling capabilities of the direct connection models are between those of model (BD) and model (CG). Indeed, a computational comparison of the models (BD), (DC), and (UT) in Chapter 8 will show that model (DC) performs similar as model (BD) in a branch-and-bound context concerning the integrality gap. Moreover, this model gives a good estimate on the number of direct travelers. In fact, the number of direct travelers can be significantly improved by model (DC) compared to model (BD). Model (UT) is much harder to solve then model (DC). It further improves the solutions of model (DC) only a little bit. We, therefore, think that model (DC) is currently the best choice to solve the integrated line planning and passenger routing problem.

Chapter 7

Polyhedral Aspects

The *Steiner tree problem* is a prototype of all problems where nodes are connected by *edges* or *arcs*, see Dell'Amico, Maffioli, and Martello [4]. The *(capacitated) network design problem* generalizes this connectivity setting to the installation of numeric arc capacities to satisfy a given demand. Basic inequalities for the Steiner tree problem, e. g., the cut inequalities, and fundamental classes of facet defining inequalities, e. g., the Steiner partition inequalities, can be extended to the network design problem to obtain strong inequalities such as *(multi-facility) cutset inequalities*, see, e. g., Magnanti, Mirchandani [71] and Raack [84], *band inequalities*, see, e. g., Stoer and Dahl [97], and *Steiner partition band inequalities*, see again Stoer and Dahl [97].

The *Steiner connectivity problem* is a prototype of all problems where nodes are connected by *paths*. The *line planning problem* generalizes this connectivity setting to the choice of lines with positive frequencies, i. e., the installation of numeric path capacities, to satisfy a given demand. The Steiner connectivity problem can, hence, be seen as an idealized line planning problem (Remark 6.2), i. e., an *uncapacitated line planning problem*. In the same way as for the Steiner tree problem and the network design problem, we can extend valid inequalities for the Steiner connectivity problem to the line planning problem. On the other hand, it is also possible to generalize inequalities for the network design problem to the line planning problem since the latter can be seen as a generalization of the first, similar as the Steiner connectivity problem is a generalization of the Steiner tree problem. Figure 7.1 illustrates the described setting.

In this chapter, we will show that cutset inequalities, band inequalities, and Steiner partition band inequalities for the network design problem can be applied or generalized to the line planning problem. Although the close relation between the line planning problem and the network design problem has been observed before, see, e. g., Magnanti and Wong [72], only few results for the network design problem have been carried over or generalized to the line planning problem. As far as we know, only Dix [44] pursues a similar approach as for the network design problem, namely, she describes the line planning problem with continuous frequencies by metric inequalities and derives cutset

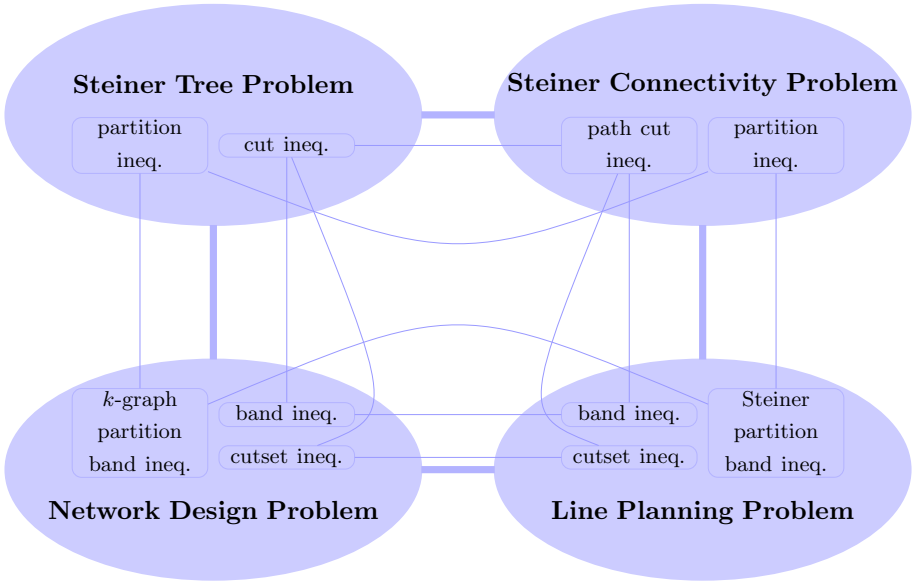


Figure 7.1: Relations between the Steiner tree problem, Steiner connectivity problem, (capacitated) network design problem, and line planning problem with important classes of inequalities. Horizontal edges represent the generalization from edges/arcs (left) to paths (right); vertical edges correspond to the generalization from pure connectivity (top) to connectivity w. r. t. capacity (bottom).

inequalities. We go one step further, put band inequalities in the context of line planning, and prove that the class of Steiner partition band inequalities yields facet defining inequalities for the line planning problem. Steiner partition band inequalities generalize not only k-graph-partition band inequalities for the network design problem, see, e. g., Stoer and Dahl [97] and Wessäly [104], but also Steiner partition inequalities of the Steiner connectivity problem, see again Figure 7.1 and compare with Chapter 2.

The structure of the chapter is as follows. We define the line planning polytope in terms of metric inequalities and investigate basic properties, e. g., its dimension, in Section 7.1. We first restrict our investigation to one metric inequality and consider a relaxation of the line planning polytope defined by one metric inequality. In Section 7.2 we apply band inequalities to our setting obtaining a large class of facet defining inequalities for this relaxation. We show that the mixed integer rounding technique also yields facet defining inequalities for this polytope in Section 7.3. In Section 7.4 we apply the developed technique to a special class of metric inequalities, the cutset inequalities. Finally, we investigate Steiner partition band inequalities in Section 7.5.

7.1 The Line Planning Polytope

In this section we define the *line planning polytope*. This polytope describes all *feasible line plans*. Recall that the *line planning problem* is to find a *feasible line plan* that optimizes a certain objective; and a feasible line plan has to provide enough capacity for

a given passenger demand. We will see that the line planning polytope can be defined by the class of so-called *metric inequalities*.

To clarify the setting, we will recall and simplify the notation of Section 6.2. Let $G = (V, E)$ be an undirected graph, $\mathcal{L}$ a set of lines (paths in G) and $\mathcal{F} = \{f_1, \dots, f_m\} \subseteq \mathbb{N}$ a set of frequencies. For ease of notation, in this chapter, we do not distinguish between OD edges and edges of different transportation types. We also assume that each line can be operated with any frequency in $\mathcal{F}$. Let further $d \in \mathbb{Q}_{\geq 0}^{V \times V}$ be a demand matrix; we denote by $D = \{(s,t) \in V \times V \mid d_{st} > 0\}$ the set of all node pairs with positive demand. Let $\kappa_{\ell,f} \in \mathbb{Q}_{\geq 0}$ be the capacity of line ℓ operated with frequency $f \in \mathcal{F}$; we have $\kappa_{\ell,f} < \kappa_{\ell,g}$ for $f < g$, $f, g \in \mathcal{F}$, $\ell \in \mathcal{L}$, and define $\kappa_{\ell,0} := 0$, $\ell \in \mathcal{L}$. We denote by (V, A) the directed counterpart of G that arises from G by replacing each edge $e \in E$ with two antiparallel arcs $a(e)$ and $\bar{a}(e)$; conversely let $e(a) \in E$ be the undirected edge corresponding to $a \in A$.

A vector $x^* \in \{0,1\}^{\mathcal{L} \times \mathcal{F}}$ is feasible for the line planning problem if the following two conditions are satisfied:

(f1) $\sum_{f \in \mathcal{F}} x^*_{\ell,f} \leq 1$ for all $\ell \in \mathcal{L}$ and
(f2) the arc capacities c_a, $a \in A$, defined as $c_a = \sum_{\ell \in \mathcal{L}: e(a) \in \ell} \sum_{f \in \mathcal{F}} \kappa_{\ell,f} x^*_{\ell,f}$, *support* d, i. e., the capacities allow a multi-commodity flow routing for the given demand d.

Choosing $\mathcal{L}' = \{\ell \in \mathcal{L} : \sum_{f \in \mathcal{F}} x^*_{\ell,f} > 0\}$ and $f'(\ell) = \sum_{f \in \mathcal{F}} f \cdot x^*_{\ell,f}$, condition (f1) says that the tuple $(\mathcal{L}', f')$ is a line plan, condition (f2) requires that this line plan is feasible.

There exists a theorem, known as the Japanese Theorem, stating necessary and sufficient conditions for a capacity vector to support a given demand:

Theorem 7.1 (Iri [62], Kakusho and Onaga [76]). *A capacity vector $\bar{c}$ supports a demand d if and only if*

$$\sum_{a \in A} \omega_a \bar{c}_a \geq \sum_{(s,t) \in D} \operatorname{dist}_\omega(s,t) d_{st} \qquad \forall\, \omega \in \mathbb{Q}^A_{\geq 0}, \tag{7.1}$$

where $\operatorname{dist}_\omega(s,t)$ is the length of a shortest st-path with respect to ω.

Inequalities (7.1) are called *metric inequalities* since ω can be restricted such that the triangle inequalities are satisfied, i. e., $\omega_{uv} \leq \operatorname{dist}_\omega(u,v)$ for $(u,v) \in A$, i. e., ω induces a (pseudo-)metric in G, see Stoer and Dahl [97]. If all data is rational, it is even sufficient to consider $\omega \in \mathbb{N}_0^A$, see also Raack [84].

Corollary 7.2. *The vector $x^* \in \{0,1\}^{\mathcal{L} \times \mathcal{F}}$ is feasible for the line planning problem if and only if*

$$\sum_{a \in A} \omega_a \sum_{\ell \in \mathcal{L}: e(a) \in \ell} \sum_{f \in \mathcal{F}} \kappa_{\ell,f} x^*_{\ell,f} \geq \sum_{(s,t) \in D} \operatorname{dist}_\omega(s,t) d_{st} \qquad \forall\, \omega \in \mathbb{Q}^A_{\geq 0} \tag{7.2}$$

$$\sum_{f \in \mathcal{F}} x^*_{\ell,f} \leq 1 \qquad \forall\, \ell \in \mathcal{L}. \tag{7.3}$$

Proof. Inequalities (7.2) correspond to the metric inequalities (7.1) for the arc capacities as defined in (f2), inequalities (7.3) correspond to (f1), i. e., (f1) and (f2) are satisfied and, hence, the claim follows. □

We denote by

$$P_{\text{LPP}} = \text{conv}\{x \in \{0,1\}^{\mathcal{L}\times\mathcal{F}} : x \text{ satisfies inequalities (7.2) and (7.3)}\}$$

the *line planning polytope*. In the following we want to investigate the line planning polytope. We will consider basic properties, e. g., its dimension, and methods to derive valid inequalities for P_{LPP} that may separate some fractional points satisfying inequalities (7.2) and (7.3). To this purpose, we apply and generalize results of Dix [44] and Stoer and Dahl [97] for related problems to our setting.

Dix [44] defined a polytope $P_{\text{LPP}'}$ for the line planning problem with continuous frequencies where the x-variables only depend on the lines (not on their frequencies) and the arc capacities are defined by $c_a = \sum_{\ell\in\mathcal{L}:e(a)\in\ell} F\cdot\kappa_\ell\cdot x_\ell$, with $F\kappa_\ell$ being the maximal capacity for line ℓ. Dix stated a result analogous to Corollary 7.2 for this setting, considered basic polyhedral properties, and motivated cutset inequalities for $P_{\text{LPP}'}$. We use similar ideas as Dix [44] to prove basic properties of P_{LPP} in Section 7.1.1.

Stoer and Dahl [97] considered a polytope for the network design problem which is equivalent to

$$\begin{aligned} P_{\text{NDP}} = \text{conv}\{x \in \{0,1\}^E : &\sum_{a\in A} \omega_a \sum_{f\in\mathcal{F}} \kappa_{e(a),f} x_{e(a),f} \geq \sum_{(s,t)\in D} \text{dist}_\omega(s,t) d_{st} && \forall\, \omega \in \mathbb{Q}^A_{\geq 0} \\ &\sum_{f\in\mathcal{F}} x_{e,f} \leq 1 && \forall\, e \in E\}, \end{aligned}$$

i. e., it can be seen as a special case of P_{LPP} where each line corresponds to one edge. More precisely, Stoer and Dahl considered a polytope defined as

$$\begin{aligned} \bar{P}_{\text{NDP}} = \{y \in \{0,1\}^E : &\sum_{a\in A} \omega_a \sum_{\bar{f}\in\bar{\mathcal{F}}} \bar{\kappa}_{e(a),\bar{f}} y_{e(a),\bar{f}} \geq \sum_{(s,t)\in D} \text{dist}_\omega(s,t) d_{st} && \forall\, \omega \in \mathbb{Q}^A_{\geq 0} \\ &1 \geq y_{e,\bar{f}_1} \geq \cdots \geq y_{e,\bar{f}_m} \geq 0 && \forall e \in E\}. \end{aligned}$$

Remark 7.3. Let $\bar{\mathcal{F}} = \{\bar{f}_1, \ldots, \bar{f}_m\}$ and define $f_i = \sum_{j=1}^i \bar{f}_j$, $i = 1, \ldots, m$, and $\kappa_{e,f_i} = \sum_{j=1}^i \bar{\kappa}_{e,\bar{f}_j}$, then $x_{e,f_m} = y_{e,f_m}$ and $x_{e,f_i} = y_{e,f_i} - y_{e,f_{i+1}}$, $i = 1, \ldots, m-1$, is a transformation between $x \in P_{\text{NDP}}$ and $y \in \bar{P}_{\text{NDP}}$ which is described by a regular matrix, i. e., the polytopes P_{NDP} and $\bar{P}_{\text{NDP}}$ are equivalent.

This means that a facet of one polytope can be transformed into a facet of the other polytope and vice versa. Dahl and Stoer present an investigation of $\bar{P}_{\text{NDP}}$ in [97]. They considered band inequalities, which can be applied in a straight forward way to our setting, and partition inequalities, which we will generalize to our setting. It will turn

out that these partition inequalities also generalize the Steiner partition inequalities of the Steiner connectivity problem considered in Subsection 2.2.1.

Bussieck [29] investigates a different polytope, namely, the polytope associated with the direct traveler model (DT). It covers the line planning problem with a fixed passenger routing, i. e., the needed capacity for each arc/edge in the network is known in advance. Hence, a description for the capacity vector by metric inequalities (7.1) is not necessary in his setting.

7.1.1 Basic Properties of the Line Planning Polytope

In the following, we analyze the dimension of the line planning polytope and conditions for basic inequalities to be facet defining.

Proposition 7.4. *The polytope P_{LPP} is full-dimensional if and only if for all $\ell \in \mathcal{L}$ the polyhedron*

$$P_\ell = \{x \in P_{\text{LPP}} \mid x_{\ell,f} = 0 \quad \forall\, f \in \mathcal{F}\}$$

is nonempty, i. e., for each line exists a solution in which the line is not operated.

Proof. "⇒" Assume the polytope P_ℓ is empty for some line $\ell \in \mathcal{L}$. Then either P_{LPP} is empty or each valid solution $x^* \in P_{\text{LPP}}$ satisfies $\sum_{f\in\mathcal{F}} x^*_{\ell,f} = 1$. In both cases P_{LPP} is not full dimensional.

"⇐" We assume that the polytope P_ℓ is nonempty for each $\ell \in \mathcal{L}$ and that the polytope P_{LPP} is not full dimensional, i. e., there exists a linear equation $a^T x = \alpha$ with nonzero coefficients a that is satisfied by each solution of P_{LPP}. Let $\ell \in \mathcal{L}$ be any line. By assumption there exists a solution x^* with $x^*_{\ell,f} = 0$ for all $f \in \mathcal{F}$. Denote by $e^{\ell,f} \in \{0,1\}^{\mathcal{L}\times\mathcal{F}}$ the unit vector that has a 1 at the position corresponding to line ℓ and frequency f and 0 at all other positions. Then the vectors $x^* + e^{\ell,f}$ for each $f \in \mathcal{F}$ are feasible for P_{LPP}. Then we have $a^T x^* = a^T(x^* + e^{\ell,f}) = \alpha$ and by subtraction we get $a_{\ell,f} = 0$ for all $f \in \mathcal{F}$. Since the line ℓ was chosen arbitrarily, we get $a = 0$, a contradiction. Therefore, the equality system of polytope P_{LPP} is empty and P_{LPP} is full dimensional. □

We assume in the following that the polytope P_{LPP} is full-dimensional, i. e., the condition of Proposition 7.4 holds.

Proposition 7.5.

1. *The inequalities $\sum_{f\in\mathcal{F}} x_{\ell,f} \le 1$ are facet-defining for P_{LPP} for all $\ell \in \mathcal{L}$.*
2. *Let $\ell \in \mathcal{L}$, $f \in \mathcal{F}$. The inequality $x_{\ell,f} \ge 0$ defines a facet for P_{LPP} if and only if the polytope*

$$P_{\ell,\tilde{\ell}} = \{x \in P_{\text{LPP}} \mid x_{\tilde{\ell},\tilde{f}} = 0 \quad \forall\, \tilde{f} \in \mathcal{F}\} \cap \{x \in P_{\text{LPP}} \mid x_{\ell,f} = 0\}$$

is nonempty for each $\tilde{\ell} \in \mathcal{L}$, i. e., setting $x_{\ell,f} = 0$ does not imply that a certain line is operated with positive frequency in any solution.

Proof. 1. Let $\ell \in \mathcal{L}$. We can define the following solutions.

(a) $\chi^X \in P_{\text{LPP}}$ for the set $X = \{(\tilde{\ell}, f_m) : \tilde{\ell} \in \mathcal{L}\}$, i. e., setting all lines to the maximal frequency.

(b) $\chi^{X_{\tilde{\ell},f}} \in P_{\text{LPP}}$ with $X_{\tilde{\ell},f} = X \setminus \{(\tilde{\ell}, f_m)\} \cup \{(\tilde{\ell}, f)\}$ for all $\tilde{\ell} \in \mathcal{L}$, $f \in \mathcal{F} \setminus \{f_m\}$, i. e., reducing the frequency of any line.

(c) $\chi^{X_{\tilde{\ell},0}} \in P_{\text{LPP}}$ with $X_{\tilde{\ell},0} = X \setminus \{(\tilde{\ell}, f_m)\}$, for all $\tilde{\ell} \in \mathcal{L} \setminus \{\ell\}$, i. e., setting any line but ℓ to 0.

The cases (b) and (c) yield solutions since the line planning polytope is full-dimensional. We have $1 + |\mathcal{L}|(|\mathcal{F}| - 1) + (|\mathcal{L}| - 1) = |\mathcal{L}||\mathcal{F}|$ affinely independent vectors and each such vector satisfies the inequality $\sum_{f \in \mathcal{F}} x_{\ell,f} \leq 1$ with equality. Therefore, this inequality is facet defining.

2. "$\Rightarrow$" Assume the polytope $P_{\ell,\tilde{\ell}}$ is empty for an $\tilde{\ell} \in \mathcal{L} \setminus \{\ell\}$, i. e., setting $x_{\ell,f} = 0$ implies $\sum_{\tilde{f} \in \mathcal{F}} x_{\tilde{\ell},\tilde{f}} = 1$. In this case the polytope $\{x \in P_{\text{LPP}} : x_{\ell,f} = 0\}$ is not full dimensional $\Rightarrow$ it contains at most $|\mathcal{L} \times \mathcal{F}| - 1$ linearly independent vectors with $x_{\ell,f} = 0 \Rightarrow x_{\ell,f} \geq 0$ does not define a facet.

"$\Leftarrow$" Let $\ell \in \mathcal{L}$ and $f \in \mathcal{F}$. If $P_{\ell,\tilde{\ell}} \neq \emptyset$ for all $\tilde{\ell} \in \mathcal{L} \setminus \{\ell\}$, the vectors of the proof of part 1 (of this proposition) where the component associated with ℓ and f is set to 0 are feasible and affinely independent. □

7.1.2 Polytope for One Metric Inequality

In the following we want to discuss two methods to derive new inequalities from a metric inequality such that the new inequalities are valid for all integral solutions of P_{LPP} but may separate fractional solutions satisfying inequalities (7.2) and (7.3). One method is to identify a *band*, i. e., frequencies for the lines such that the considered metric inequality is not satisfied. Such a band gives rise to so-called *band inequalities* which correspond to cover inequalities for knapsack problems with generalized upper bound constraints. The other method is to use mixed integer rounding, i. e., the metric inequality is multiplied with a scalar and then the coefficients and the right hand side are rounded exploiting the integrality requirements of the x-variables.

Consider some metric inequality, i. e., let $\omega \in \mathbb{Q}^A_{\geq 0}$. We set $a_{\ell,f} = \sum_{a \in A : e(a) \in \ell} \omega_a \kappa_{\ell,f}$ and $b = \sum_{(s,t) \in D} \text{dist}_\omega(s,t) d_{st}$. Let $\mathcal{L}_M$ be the set of lines with $a_{\ell,f} > 0$ for a frequency $f \in \mathcal{F}$. Then the metric inequality corresponding to ω reads as follows, compare with (7.2):

$$\sum_{\ell \in \mathcal{L}_M} \sum_{f \in \mathcal{F}} a_{\ell,f} x_{\ell,f} \geq b. \tag{7.4}$$

Recall that $a_{\ell,0} = 0$ for all $\ell \in \mathcal{L}$ and $a_{\ell,f_1} < a_{\ell,f_2}$ for $f_1 < f_2$. In the following we identify

this inequality with the tuple (a, b) and assume that $b > 0$. Let

$$P_M = \text{conv}\{x \in \{0,1\}^{\mathcal{L}_M \times \mathcal{F}} \mid x \text{ satisfies (7.4)}, \sum_{f \in \mathcal{F}} x_{\ell,f} \leq 1 \, \forall \, \ell \in \mathcal{L}_M\}.$$

be the polytope for the metric inequality (7.4) with additional generalized upper bound constraints. Note that the definition of this polytope is independent of whether ℓ is a path or just an edge in the considered graph, i. e., the polytope P_M is equivalent to the ICOV polytope defined by Stoer and Dahl, i. e., the polytope for one metric inequality. We can, hence, transfer their results to our setting by using the transformation given in Remark 7.3. Each inequality valid for P_M is also valid for P_{LPP} by setting the coefficients of all variables corresponding to lines in $\mathcal{L} \setminus \mathcal{L}_M$ to 0. In the next two sections, we show that the band inequalities and the mixed integer rounding technique yield facet defining inequalities for P_M.

7.2 Band Inequalities

The band inequalities were originally proposed and analyzed for the telecommunication network design problem by Stoer and Dahl [97]. They can be defined for any metric inequality. We apply them to our setting. The idea of a band inequality is to associate with each line a frequency such that the considered inequality is not satisfied. We start with the definition of a *valid band*.

Definition 7.6. *We call any assignment $B : \mathcal{L}_M \to \mathcal{F} \cup \{0\}$ of lines in $\mathcal{L}_M$ to a positive frequency or zero a* band for $\mathcal{L}_M$. *The band B is* valid *for (a, b) if*

$$\sum_{\ell \in \mathcal{L}_M} a_{\ell,B(\ell)} < b.$$

If we have found a valid band then we know that at least one line $\ell \in \mathcal{L}_M$ has to be operated at a higher frequency than $B(\ell)$. This is the idea of the band inequality.

Lemma 7.7. *Let B be a valid band for (a, b), then the* band inequality

$$\sum_{\ell \in \mathcal{L}_M} \sum_{\substack{f \in \mathcal{F} \\ f > B(\ell)}} x_{\ell,f} \geq 1$$

is valid for P_{LPP}.

The band inequalities are of particular interest if the valid band is maximal.

Definition 7.8. *A band B valid for (a, b) is* maximal *if there does not exist a valid band $\tilde{B}$ for (a, b) with $B(\ell) \leq \tilde{B}(\ell)$ for all $\ell \in \mathcal{L}_M$ and $B(\tilde{\ell}) < \tilde{B}(\tilde{\ell})$ for at least one $\tilde{\ell} \in \mathcal{L}_M$, in particular*

$$\sum_{\ell \in \mathcal{L}_M} a_{\ell,B(\ell)} < \sum_{\ell \in \mathcal{L}_M} a_{\ell,\tilde{B}(\ell)} < b.$$

Band inequalities are equivalent to cover inequalities for the knapsack problem with generalized upper bound constraints. Stoer and Dahl [97] showed that band inequalities are facet defining for the network design polytope for one metric inequality if and only if the band is maximal. More precisely, Dahl and Stoer proved the following result (interpreted in our notation).

Proposition 7.9. *Assume that $b \leq a_{\ell,f_m}$ for all $\ell \in \mathcal{L}_M$, i. e., operating any line at its maximal frequency satisfies the metric inequality (7.4). Let B be a valid band for (a, b) and $|\mathcal{L}_M| \geq 2$. Then the band inequality*

$$\sum_{\ell \in \mathcal{L}_M} \sum_{\substack{f \in \mathcal{F} \\ f > B(\ell)}} x_{\ell,f} \geq 1 \tag{7.5}$$

is facet defining for P_M if and only if the band is maximal.

We give an example of a band inequality that is not facet defining if the assumption $b \leq a_{\ell,f_m}$ for all $\ell \in \mathcal{L}_M$ is not satisfied, and we show that the assumption is not necessary.

Example 7.10. *Consider three lines $\mathcal{L}_M = \{1, 2, 3\}$ and three frequencies $\mathcal{F} = \{1, 2, 3\}$; set $a_{i,1} = 1$, $a_{i,2} = 3$, and $a_{i,3} = 5$ for $i = \{1, 2, 3\}$. The right hand side is $b = 7$. The metric inequality and the generalized upper bound constraints for this example are:*

$$\begin{aligned} x_{1,1} + 3x_{1,2} + 5x_{1,3} + x_{2,1} + 3x_{2,2} + 5x_{2,3} + x_{3,1} + 3x_{3,2} + 5x_{3,3} &\geq 7 \\ x_{1,1} + x_{1,2} + x_{1,3} &\leq 1 \\ x_{2,1} + x_{2,2} + x_{2,3} &\leq 1 \\ x_{3,1} + x_{3,2} + x_{3,3} &\leq 1. \end{aligned}$$

Let P_M be the polytope defined by all integral solutions of the above inequalities and the nonnegative constraints $x_i \geq 0$, $i = 1, 2, 3$. A valid and maximal band for the metric inequality is $B(1) = B(2) = 1$, $B(3) = 2$. The associated band inequality is $x_{1,2} + x_{1,3} + x_{2,2} + x_{2,3} + x_{3,3} \geq 1$. It is the sum of the two inequalities $x_{1,2} + x_{1,3} + x_{2,2} + x_{2,3} + x_{3,1} + x_{3,2} + 2x_{3,3} \geq 2$ (number (15) in Table 7.1) and $x_{3,1} + x_{3,2} + x_{3,3} \leq 1$ (a generalized upper bound constraint) which are facet defining inequalities for P_M. However, setting $B(1) = B(2) = 2$ and $B(3) = 0$ is also a maximal valid band. The corresponding band inequality is $x_{1,3} + x_{2,3} + x_{3,1} + x_{3,2} + x_{3,3} \geq 1$ (number (18) in Table 7.1) which is facet defining for P_M, i. e., the assumption $b \leq a_{\ell,f_m}$ for all $\ell \in \mathcal{L}_M$ is not necessary for a band inequality to be facet defining. The nontrivial facets for this instance were computed by PORTA *[101] and are listed in Table 7.1.*

Note that the separation problem for band inequalities is $\mathcal{NP}$-complete since it is equivalent to a knapsack problem with ordering constraints, see Dahl and Stoer [40].

Table 7.1: Facets for the polytope described by all vectors $x \in \{0,1\}^{3\times 3}$ that satisfy the inequalities given in Example 7.10. The facets were computed by PORTA [101]. We list the coefficients and the right hand side for all facets but the trivial facets and the generalized upper bound constraints.

	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$	$\geq b$
(1)	1	2	3	1	2	3	1	2	3	5
(2)	1	2	2	1	2	2	1	2	2	4
(3)	1	1	2	1	1	2	1	1	2	3
(4)	1	2	2	1	1	2		1	2	3
(5)	1	1	2	1	2	2		1	2	3
(6)	1	2	2		1	2	1	1	2	3
(7)	1	1	2		1	2	1	2	2	3
(8)		1	2	1	2	2	1	1	2	3
(9)		1	2	1	1	2	1	2	2	3
(10)	1	1	1	1	1	1		1	1	2
(11)	1	1	1		1	1	1	1	1	2
(12)		1	1	1	1	1	1	1	1	2
(13)	1	1	2		1	1		1	1	2
(14)		1	1	1	1	2		1	1	2
(15)		1	1		1	1	1	1	2	2
(16)	1	1	1			1			1	1
(17)			1	1	1	1			1	1
(18)			1			1	1	1	1	1

7.3 Improving Inequalities by Mixed Integer Rounding

The metric inequality (7.4) can also be strengthened by a mixed integer rounding technique. We can generate *mixed integer rounding inequalities* from $ax \geq b$ as follows.

Proposition 7.11. *(Wolsey [106]) Let* $r := b - \lfloor b \rfloor$, $r_{\ell,f} := a_{\ell,f} - \lfloor a_{\ell,f} \rfloor$. *The mixed integer rounding inequality*

$$\sum_{\ell \in \mathcal{L}_M} \sum_{f \in \mathcal{F}} \left(r \lfloor a_{\ell,f} \rfloor + \min\{r_{\ell,f}, r\} \right) x_{\ell,f} \geq r \lceil b \rceil \tag{7.6}$$

is valid for P_M.

Multiplying inequality (7.4) by a scalar $\lambda \in \mathbb{R}_+$ does not change the polytope P_M, i. e., we can try different values for λ to find strengthening inequalities for P_M by the mixed integer rounding method. Obviously, this only yields nontrivial inequalities if the right hand side of the scaled inequality $ax \geq b$ is not integral. Dash, Günlück, and Lodi [41] showed which multipliers are relevant to get all different mixed integer rounding inequalities for a given set of constraints. Applied to our setting their results yield the following theorem.

Theorem 7.12 (Dash, Günlück, and Lodi [41]). *Each non-redundant mixed integer rounding inequality for* $ax \geq b$ *is defined by rational multipliers taking values in* $(0,1)$ *with a denominator equal to* $a_{\ell,f}$, $\ell \in \mathcal{L}$, $f \in \mathcal{F}$, *or equal to* b.

Example 7.13. *Consider again the instance given in Example 7.10 with the metric inequality*

$$x_{1,1} + 3x_{1,2} + 5x_{1,3} + x_{2,1} + 3x_{2,2} + 5x_{2,3} + x_{3,1} + 3x_{3,2} + 5x_{3,3} \geq 7.$$

Multiplying this inequality by $\frac{1}{3}$ *and applying mixed integer rounding yields inequality*

$$\begin{aligned} & \tfrac{1}{3}x_{1,1} + \tfrac{1}{3}x_{1,2} + \tfrac{2}{3}x_{1,3} + \tfrac{1}{3}x_{2,1} + \tfrac{1}{3}x_{2,2} + \tfrac{2}{3}x_{2,3} + \tfrac{1}{3}x_{3,1} + \tfrac{1}{3}x_{3,2} + \tfrac{2}{3}x_{3,3} \geq \tfrac{1}{3} \cdot 3 \\ \Leftrightarrow \quad & x_{1,1} + x_{1,2} + 2x_{1,3} + x_{2,1} + x_{2,2} + 2x_{2,3} + x_{3,1} + x_{3,2} + 2x_{3,3} \geq 3. \end{aligned}$$

which is facet defining for P_M *(as defined in Example 7.10), compare with number (3) in Table 7.1.*

7.4 Cutset Inequalities

We have seen in the previous two sections how metric inequalities can be strengthened by band inequalities and by the mixed integer rounding technique. In this section we consider capacitated cutset inequalities which are the most important and most investigated metric inequalities, see, e. g., Raack [84]. We can define band inequalities for and apply mixed integer rounding to capacitated cut inequalities to generate strengthened inequalities for the line planning problem.

A capacitated cutset inequality stipulates a minimal capacity requirement on a cut in a graph, i. e., a partition of the nodes into two sets such that the demand that has to be transported from one set to the other is positive. Capacitated cutset inequalities can be derived from metric inequalities as follows. Let $\emptyset \neq W \subset V$ be a nonempty node set and $S = \delta^+(W)$ be the arcs of the cut from W to $V \setminus W$. Define $\omega_a = 1$ for $a \in S$ and $\omega_a = 0$, $a \notin S$. The metric inequality (7.2) for this setting reads

$$\sum_{a \in S} \sum_{\ell \in \mathcal{L}: e(a) \in \ell} \sum_{f \in \mathcal{F}} \kappa_{\ell,f} x_{\ell,f} \geq \sum_{(s,t) \in D} \operatorname{dist}_\omega(s,t) d_{st}, \tag{7.7}$$

i. e., here $a_{\ell,f} = \sum_{e \in S: e \in \ell} \kappa_{\ell,f}$ and $b = \sum_{(s,t) \in D} \operatorname{dist}_\omega(s,t) d_{st}$ to get the structure of (7.4).

Let $\bar{W} := V \setminus W$, $\bar{S} = \delta^+(\bar{W}) = \delta^-(W)$, $\bar{\omega}_a = 1$ for all $a \in \bar{S}$, and $\bar{\omega}_a = 0$ for all $a \notin \bar{S}$. Then $\operatorname{dist}_w(s,t) = \operatorname{dist}_{\bar{\omega}}(t,s)$, for all $s \in W, t \in \bar{W}$. The metric inequality for this setting reads

$$\sum_{a \in \bar{S}} \sum_{\ell \in \mathcal{L}: e(a) \in \ell} \sum_{f \in \mathcal{F}} \kappa_{\ell,f} x_{\ell,f} \geq \sum_{(s,t) \in D} \operatorname{dist}_{\bar{\omega}}(s,t) d_{st}. \tag{7.8}$$

Since the lines are undirected, inequalities (7.7) and (7.8) have the same left hand side. In particular, inequality (7.7) dominates inequality (7.8) if

$$\sum_{(s,t) \in D: s \in W, t \notin W} d_{st} > \sum_{(s,t) \in D: s \notin W, t \in W} d_{st}.$$

If the relation is the other way round inequality (7.8) dominates inequality (7.7).

Let $\mathcal{L}_S = \{\ell \in \mathcal{L} \,|\, \exists\, a \in S : e(a) \in \ell\}$ be the set of all lines that have at least one edge covering an arc connecting a node in W with a node in $\bar{W} = V \setminus W$, $\alpha_\ell = \sum_{a \in S, e(a) \in \ell} 1$ be the number of all such edges line contained in ℓ, and

$$d_S = \max\{\sum_{(s,t)\in D: s\in W, t\notin W} d_{st}, \sum_{(s,t)\in D: s\notin W, t\in W} d_{st}\}$$

be the maximum of the total number of passengers who want to travel either from a node in W to a node in $\bar{W}$ or from a node in $\bar{W}$ to a node in W. In the following we assume that $d_S > 0$. We then define the *capacitated cutset inequality* as

$$\sum_{\ell \in \mathcal{L}_S} \sum_{f \in \mathcal{F}} \alpha_\ell \kappa_{\ell,f} x_{\ell,f} \geq d_S. \tag{7.9}$$

Assume $\sum_{(s,t)\in D: s\in W, t\notin W} d_{st} > \sum_{(s,t)\in D: s\notin W, t\in W} d_{st}$. Then the capacitated cutset inequality (7.9) is, in general, dominated by the metric inequality (7.7) since in (7.9) each demand between a node in W and a node in $V \setminus W$ is weighted with 1 while in (7.7) it is possible that $\text{dist}_\omega(s,t) \geq 1$ for $s \in W$ and $t \notin W$. Both inequalities are equivalent (for undirected graphs) if the node sets W and $V \setminus W$ are connected, see Costa, Cordeau, and Gendron [39]. More precisely, the result of Costa, Cordeau, and Gendron [39] for our setting reads as follows.

Proposition 7.14. *The capacitated cutset inequality (7.9) and the metric inequality (7.7) are equivalent if and only if the following two conditions hold.*

1. *For each OD pair $(s,t) \in D$ with $s \in W$ and $t \in V \setminus W$ there exists an st-path that crosses the cut S exactly once.*
2. *For each OD pair $(s,t) \in D$ with $s,t \in W$ or $s,t \in V \setminus W$ there exists an st-path that never crosses the cut S.*

Setting $B(\ell) = 0$, for all $\ell \in \mathcal{L}_S$, is a *trivial band* for the capacitated cutset inequality (7.9). Certainly this is not necessarily a maximal band. It yields

$$\sum_{\ell \in \mathcal{L}_S} \sum_{f \in \mathcal{F}} x_{\ell,f} \geq 1. \tag{7.10}$$

This inequality can be interpreted as "at least one line has to be operated on this cut". We call this inequality a *line cutset inequality*. The trivial band can be easily improved as follows. Let

$$\tilde{f} := \underset{f \in \mathcal{F} \cup 0}{\text{argmax}} \left\{ \sum_{\ell \in \mathcal{L}_S} \alpha_\ell \kappa_{\ell,f} < d_S \right\},$$

i. e., setting every line in $\mathcal{L}_S$ to frequency $\tilde{f}$ is not sufficient to transport all passengers on the cut. Applying the improved trivial band to the capacitated cutset inequality (7.9)

yields

$$\sum_{\ell \in \mathcal{L}_S} \sum_{\substack{f \in \mathcal{F} \\ f > \tilde{f}}} x_{\ell,f} \geq 1, \tag{7.11}$$

the *improved line cutset inequality*. It is easy to see that the improved line cutset inequality is not facet defining if the band is not maximal. In the next section, we will discuss a more general setting that yields also sufficient conditions for a band inequality derived from a capacitated cutset inequality to be facet defining for P_{LPP}.

7.5 Steiner Partition Band Inequalities

Stoer and Dahl [97] also presented and analyzed so-called *k-graph-partition band inequalities*. The idea is to combine the Steiner partition inequalities and the band inequalities. In the following, we generalize the k-graph-partition band inequalities to our case and call them *Steiner partition band inequalities*.

Let $P = (V_1, \ldots, V_k)$, $k \geq 2$, be a *capacitated Steiner partition* of the node set V, i. e., for each V_i, $i = 1, \ldots, k$, we have

$$d_{V_i} := \max\{ \sum_{(s,t) \in D : s \in V_i, t \notin V_i} d_{st}, \sum_{(s,t) \in D : s \notin V_i, t \in V_i} d_{st} \} > 0,$$

i. e., V_i contains at least one OD node with positive supply or demand. Let $G_P = (V_P, E_P)$ be the graph that arises from contracting each node set V_i to a single node $V_i \in V_P$. Here, we denote by V_i a node set in a partition of G as well as a node in the shrunk graph G_P. Note that G_P can have parallel edges. Furthermore, let $\mathcal{L}_P := \{\ell \in \mathcal{L} \mid \exists\, V_i, V_j \in V_P,\, V_i \neq V_j,\, V_i \in \ell, V_j \in \ell\}$ be the set of lines containing at least one edge in E_P and $\overline{\mathcal{L}} := \mathcal{L} \setminus \mathcal{L}_P$ its complement. Finally, $G[V_i]$ is the graph induced by the nodes V_i, i. e., $G[V_i] := (V_i, E_i)$ with $E_i := \{e = \{u, v\} \in E : u, v \in V_i\}$.

Let $B : \mathcal{L}_P \to \mathcal{F} \cup \{0\}$ be an assignment of all lines in $\mathcal{L}_P$ to a positive frequency or zero. We call B a *P-band* and say that B is *valid* if

$$\sum_{e \in \delta_{G_P}(W)} \sum_{\ell : e \in \ell} \kappa_{\ell, B(\ell)} < d_W \quad \forall\, \emptyset \neq W \subset V_P$$

with $\delta_{G_P}(W) = \{\{u, v\} \in E_P : u \notin W, v \in W\}$, $i = 1, \ldots, k$, and

$$d_W := \max\{ \sum_{(s,t) \in D : s \in W, t \notin W} d_{st}, \sum_{(s,t) \in D : s \notin W, t \in W} d_{st} \}.$$

A P-band B is called maximal if there exists no valid P-band $\tilde{B}$ with $B(\ell) \leq \tilde{B}(\ell)$ for all $\ell \in \mathcal{L}_P$ and $B(\tilde{\ell}) < \tilde{B}(\tilde{\ell})$ for at least one $\tilde{\ell} \in \mathcal{L}_P$.

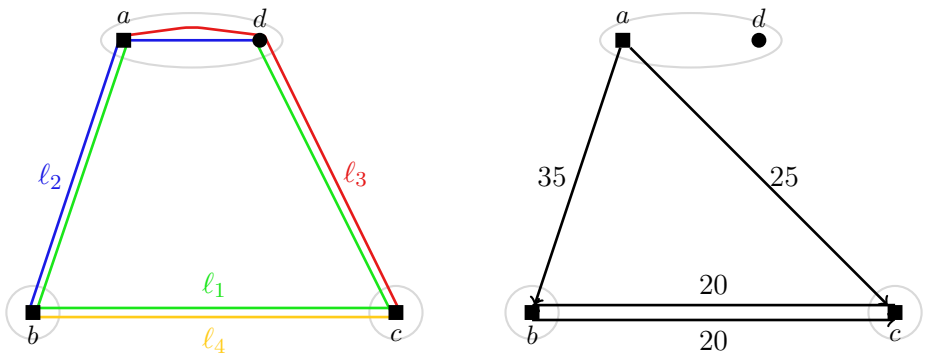


Figure 7.2: Example of a Steiner partition band inequality. *Left:* Graph with four lines ($\ell_1 = (a, b, c, d)$, $\ell_2 = (d, a, b)$, $\ell_3 = (a, d, c)$, $\ell_4 = (b, c)$) partitioned into three sets. The capacity of each line is 10, the possible frequencies are 1, 2, or 3. *Right*: Demand Graph: 35 passengers want to travel from a to b, 25 from a to c, 20 passengers between b and c in both directions. A valid band is given by setting the frequency of ℓ_1, ℓ_3, ℓ_4 to 1 and of ℓ_2 to 2. The corresponding Steiner partition band inequality is $2x_{\ell_1,2} + 2x_{\ell_1,3} + x_{\ell_2,3} + x_{\ell_3,2} + x_{\ell_3,3} + x_{\ell_4,2} + x_{\ell_4,3} \geq 2$.

Lemma 7.15. *Let P be a Steiner-partition and B a valid P-band. Furthermore, let $a_\ell := |\{i \in \{1, \dots, k\} : V_i \in V_P, V_i \in \ell\}| - 1$. Then the* Steiner partition band inequality *defined as*

$$\sum_{\ell \in \mathcal{L}_P} \sum_{\substack{f \in \mathcal{F} \\ f > B(\ell)}} a_\ell \, x_{\ell,f} \geq k - 1 \tag{7.12}$$

is valid for P_{LPP}.

Proof. Assume there exists a point $x^* \in P_{\text{LPP}}$ with

$$\sum_{\ell \in \mathcal{L}_P} \sum_{\substack{f \in \mathcal{F} \\ f > B(\ell)}} a_\ell \, x^*_{\ell,f} < k - 1.$$

The coefficient a_ℓ, $\ell \in \mathcal{L}_P$, counts the number of nodes in G_P that ℓ contains minus one, i. e., a_ℓ is the maximum number of edges that ℓ can contribute to a spanning tree in G_P. Therefore the solution x^* connects at most $k - 1$ nodes of G_P by lines with frequencies greater than the frequencies defined for the valid P-band B. This means that there is at least one node V_i that is not incident to lines with frequencies greater than the ones defined for B. Since B is a valid P-band we get

$$\sum_{e \in \delta_{G_P}(V_i)} \sum_{\ell : e \in \ell} \sum_{f \in \mathcal{F}} \kappa_{\ell,f} \, x^*_{\ell,f} < d_{V_i}.$$

This is a contradiction to x^* being a point in P_{LPP}. □

An example of a Steiner partition band inequality can be seen in Figure 7.2.

Denote by $T = \{v \in V : \exists w \in V \text{ s.t. } d_{vw} + d_{wv} > 0\} \subseteq V$ the set of nodes with positive supply or demand. Let $H = (T, F)$ be the (undirected) demand graph with

$F = \big\{\{u,v\} \,|\, u,v \in T,\, d_{uv} + d_{vu} > 0\big\}$. In the following we assume that the demand graph is connected. Then every solution for the line planning problem is a T-connecting set, compare with Chapter 1. Let

$$P_{\text{SCP}} := \text{conv}\{y \in \{0,1\}^{\mathcal{L}} : y \text{ satisfies all Steiner path cut constraints}\}$$

be the Steiner connectivity polytope as defined in Chapter 2, i. e., the polytope that contains all T-connecting sets. Then we have

$$x \in P_{\text{LPP}} \Rightarrow y \in P_{\text{SCP}} \quad \text{with } y_\ell = \sum_{f \in \mathcal{F}} x_{\ell,f} \;\forall \ell \in \mathcal{L}. \tag{7.13}$$

If we assume that the maximal frequency f_m is large enough, i. e., setting any line to the maximal frequency always satisfies the required demand for the edges on which the line is operated, we also get

$$y \in P_{\text{SCP}} \Rightarrow x \in P_{\text{LPP}} \quad \text{with } x_{\ell,f_m} = y_\ell,\; x_{\ell,f} = 0 \;\forall \ell \in \mathcal{L}, f \in \mathcal{F} \setminus \{f_m\}. \tag{7.14}$$

Stoer and Dahl [97] showed that a k-graph-partition band inequality is facet defining for the network design polytope if the band is maximal and the (underlying uncapacitated) Steiner partition inequality is facet defining for the (underlying) Steiner tree problem.

We generalize the properties to our case and get the following result.

Proposition 7.16. *Let P be a Steiner partition of G and B a valid P-band. The Steiner partition band inequality (7.12) is facet defining for P_{LPP} if the following properties are satisfied.*

1. *The underlying uncapacitated Steiner partition inequality $\sum_{\ell \in \mathcal{L}_P} a_\ell y_\ell \geq k-1$ is facet defining for P_{SCP}.*
2. *The maximal frequency is large enough, i. e., (7.14) is satisfied.*
3. *Each line contains at most two nodes in G_P, i. e., $a_\ell \in \{0,1\}$ for all $\ell \in \mathcal{L}$.*
4. *The P-band B is maximal.*

Recall that property 1 implies that the shrunk graph G_P is 2-node-path-connected for the case that the partition contains at least three sets (see Definition 3.10), compare with Proposition 2.18. If the partition contains only two sets, the shrunk graph is trivially 2-node-path-connected. Compare the following proof also with the one of Proposition 2.17.

Proof. Let $P = (V_1, \ldots, V_k)$ be a Steiner partition in G and consider the corresponding partition inequality $a^T x = \sum_{\ell \in \mathcal{L}_P} \sum_{f \in \mathcal{F}} a_{\ell,f} x_{\ell,f} \geq k-1$. Assume that properties 1 to 4 are satisfied. Let $b^T x = \beta$ be an equation such that

$$F_a = \{x \in P_{\text{LPP}} \,|\, a^T x = k-1\} \subseteq F_b = \{x \in P_{\text{LPP}} \,|\, b^T x = \beta\}$$

and such that F_b is a facet of P_{LPP}. Let further $F_a^{\text{SCP}} = \{y \in P_{\text{SCP}} \,|\, a^T y = k-1\}$ be a facet of P_{SCP} (property 1).

We first show that $b_{\ell,f} = 0$ for all $\ell \in \overline{\mathcal{L}}$ and $f \in \mathcal{F}$. Since the underlying uncapacitated Steiner partition inequality $\sum_{\ell \in \mathcal{L}_P} a_\ell y_\ell \geq k-1$ is facet defining for P_{SCP} (property 1), there exists $\mathcal{L}' \subseteq \mathcal{L} \setminus \{\ell\}$ with $\chi^{\mathcal{L}'} \in F_a^{\text{SCP}}$. According to property 2 we get $\chi^X \in P_{\text{LPP}}$ with $X = \{(\ell', f_m) : \ell' \in \mathcal{L}'\}$ and $\chi^X \in F_a$. Define $X_f := X \cup \{(\ell, f)\}$ then $\chi^{X_f} \in F_a$ and $a^T\chi^X = a^T\chi^{X_f} = k-1$. Thus, $b^T\chi^X = b^T\chi^{X_f}$ which implies $b_{\ell,f} = 0$.

With a similar construction, we get that $b_{\ell,f} = 0$ for all $\ell \in \mathcal{L}_P$ and $f \leq B(\ell)$.

Let $\ell_1 \in \mathcal{L}_P$ and $f_1, f_2 > B(\ell_1)$. Choose a set $\mathcal{L}' \subseteq \mathcal{L}_P$, $\ell_1 \in \mathcal{L}'$, such that $\chi^{\mathcal{L}' \cup \overline{\mathcal{L}}} \in F_a^{\text{SCP}}$, i. e., $|\mathcal{L}'| = k-1$. This is possible with property 1. Define

$$X_i = \{(\ell, f_m) : \ell \in (\mathcal{L}' \cup \overline{\mathcal{L}}) \setminus \{\ell_1\}\} \cup \{(\ell_1, f_i)\}, \quad i = 1, 2.$$

Then $\chi^{X_i} \in P_{\text{LPP}}$; this follows with $\chi^{\mathcal{L}' \cup \overline{\mathcal{L}}} \in P_{\text{SCP}}$, property 2, and the maximality of the band B. We also have $\chi^{X_i} \in F_a$, $i = 1, 2$, since $|\mathcal{L}'| = k-1$, and $0 = b^T\chi^{X_1} = b^T\chi^{X_2} = b_{\ell_1,f_1} - b_{\ell_1,f_2}$. Hence, the coefficients for one line and different frequencies higher than the frequency defined by the band are equal.

Let $\ell_3 \in \mathcal{L}_P \setminus \{\ell_1\}$. Consider the graph $\hat{G}_P = (V_P, \mathcal{L}_P)$ in which ℓ is an edge between V_i and V_j if it contains V_i and V_j (recall that $\ell \in \mathcal{L}_P$ contains exactly two nodes). Since G_P is 2-node-path-connected, $\hat{G}_P$ is 2-node-connected and there exists a cycle C in $\hat{G}_P$ containing ℓ_1 and ℓ_3. Let $\mathcal{L}''$ be a tree in $\hat{G}_P$ containing $C \setminus \{\ell_3\}$. With property 1 we have $\chi^{\mathcal{L}'' \cup \overline{\mathcal{L}}} \in F_a^{\text{SCP}}$, $\chi^{\mathcal{L}'' \setminus \{\ell_1\} \cup \{\ell_3\} \cup \overline{\mathcal{L}}} \in F_a^{\text{SCP}}$, i. e., $|\mathcal{L}''| = |\mathcal{L}'' \setminus \{\ell_1\} \cup \{\ell_3\}| = k-1$. Define

$$X_3 = \{(\ell, f_m) : \ell \in \overline{\mathcal{L}} \cup \mathcal{L}'' \setminus \{\ell_1\}\} \cup \{(\ell_1, f_1)\} \quad X_4 = \{(\ell, f_m) : \ell \in \overline{\mathcal{L}} \cup \mathcal{L}'' \setminus \{\ell_1\}\} \cup \{(\ell_3, f_3)\}$$

with $f_3 > B(\ell_3)$; recall $f_1 > B(\ell_1)$. Then $\chi^{X_i} \in P_{\text{LPP}}$, $i = 3, 4$; this follows with the same arguments as above. We also have $\chi^{X_i} \in F_a$, $i = 3, 4$, since $|\mathcal{L}''| = k-1$, and $0 = b^T\chi^{X_3} = b^T\chi^{X_4} = b_{\ell_1,f_1} - b_{\ell_3,f_3}$. Hence, the coefficients for different lines and frequencies higher than the frequency defined by the band are equal. It finally follows that $b^T x$ is a multiple of $a^T x$. This proves the claim. □

Proposition 7.17. *Properties 1 and 4 are also necessary conditions for a Steiner partition band inequality to be facet defining.*

Proof. We first show that property 1 is necessary. Let P be a Steiner partition of G and B a valid P-band. Assume the Steiner partition band inequality (7.12) is facet defining for P_{LPP}. Let $\{x^j \in P_{\text{LPP}}, j = 1, \ldots, |\mathcal{L}||\mathcal{F}|\}$, be a set of linearly independent vectors that satisfy inequality (7.12) with equality. We argue that the transformation given in (7.13) yields $|\mathcal{L}|$ linearly independent vectors for P_{SCP} satisfying the corresponding Steiner partition inequality with equality. Define a matrix $A \in \{0,1\}^{|\mathcal{L}||\mathcal{F}| \times |\mathcal{L}||\mathcal{F}|}$ with columns being x^j, $j = 1, \ldots, |\mathcal{L}||\mathcal{F}|$. Assume that $\mathcal{L} = \{1, \ldots, n\}$ and the rows $((i-1)|\mathcal{F}|+1)$ to $(i|\mathcal{F}|)$, $i = 1, \ldots, |\mathcal{L}|$, of A correspond to line i. Define a matrix $B \in \{0,1\}^{|\mathcal{L}| \times |\mathcal{L}||\mathcal{F}|}$ with $b_{ij} = 1$ for $j \in \{(i-1)|\mathcal{F}|+1, \ldots, i|\mathcal{F}|\}$ and $b_{ij} = 0$ otherwise. The multiplication of B and A , i. e., $BA \in \{0,1\}^{|\mathcal{L}| \times |\mathcal{L}||\mathcal{F}|}$, corresponds to the transformation given in (7.13).

We get $\operatorname{rank}(B \cdot A) \geq \operatorname{rank}(B) + \operatorname{rank}(A) - |\mathcal{L}||\mathcal{F}| = |\mathcal{L}|$ [48]. Hence there are $|\mathcal{L}|$ linearly independent vectors in BA and each such vector y satisfies $\sum_{\ell \in \mathcal{L}_P} a_\ell y_\ell \geq k - 1$ with equality, i. e., the Steiner partition inequality is facet defining for P_{SCP}.

Now we show that property 4 is necessary. Assume the P-band B is not maximal. Then there exists a valid P-band $\tilde{B}$ with $B(\ell) \leq \tilde{B}(\ell)$ for all $\ell \in \mathcal{L}_P$ and $B(\tilde{\ell}) < \tilde{B}(\tilde{\ell})$ for at least one $\tilde{\ell} \in \mathcal{L}_P$. Then the Steiner partition band inequality for B is dominated by the Steiner partition band inequality for $\tilde{B}$. □

Corollary 7.18. *A band inequality for a capacitated cutset inequality is facet defining for P_{LPP} if*

1. *The corresponding Steiner path cut inequality is facet defining for P_{SCP}, compare with Lemma 2.15.*
2. *The maximal frequency is large enough, i. e., (7.14) is satisfied.*
3. *The P-band B is maximal.*

The second property in the above corollary is not necessary. To see that, note that the metric inequality considered in Example 7.10 can be interpreted as a capacitated cutset inequality. The example then shows that a band inequality can be facet defining if property 2 is not satisfied.

Chapter 8

Solving the Line Planning Problem

In this chapter, we describe algorithms for the solution of the integrated line planning and passenger routing problem. We will compare the models (BD), (DC), and (UT) according to their computability and solution quality. It turns out that the direct connection model (DC) is the best computationally tractable line planning method that provides good estimates of direct travelers. We will show that its direct connection constraints strongly improve the number of direct travelers in comparison to the basic dynamic model (BD) which ignores a large number of transfers. Model (UT) only provides little better estimates of direct travelers than model (DC). However, it is, especially for large instances, much harder to solve than the models (DC) or (BD). We also tried to compute the change-and-go model (CG), however, even the root LP relaxation could only be solved for half of the instances within 10 hours.

We developed a branch-and-cut-and-price algorithm based on our investigation of the line planning models in Chapters 6 and 7. This involves heuristics, cutting planes, and pricing methods for the passenger path variables. In Section 8.1 we describe the test instances and some preprocessing of the data. We propose primal heuristics in Section 8.2, investigate approaches to find violated cutting planes in Section 8.3, and suggest a constraint branching in Section 8.4. We then compare the computational performance, e. g., the number of solved branching nodes, as well as primal and dual bounds, of the models (BD), (DC), and (UT) and evaluate the solutions according to the exact number of direct travelers using the change-and-go graph in Section 8.5.

8.1 Data and Preprocessing

We consider four transportation networks that we denote as China, Dutch, SiouxFalls, and Potsdam. The instance SiouxFalls uses the graph of the street network with the same name from the Transportation Network Test Problems Library of Bar-Gera [98]. Instances China, Dutch, and Potsdam correspond to public transportation networks.

The Dutch network was introduced by Bussieck in the context of line planning [28]. The China instance is artificial; we constructed it as a showcase example, connecting the twenty biggest cities in China by the 2009 high speed train network. The Potsdam instances are real-world multi-modal public transportation networks of the years 1998 and 2009; the Potsdam network distinguishes different modes for bus, tram, regional, and commuter trains. The Potsdam2010 instance arose within a project with the Verkehr in Potsdam GmbH (ViP) [96] to optimize the 2010 line plan [10, 24], see also Chapter 9.

The Potsdam data is given by a network file and an OD matrix in Visum format. We transformed the data for all other instances also into Visum format. Visum is a software system of the ptv AG [83] for "traffic analyses, forecasts, and data management".

For China, Dutch, and SiouxFalls all nodes are considered as termini, i. e., nodes where lines can start or end. We constructed a line pool by generating for each pair of termini all lines that satisfy a certain length restriction. To be more precise, the number of edges of a line between the two termini s and t must be less than or equal to k times the number of edges of the shortest path between s and t. For each network, we increased k in three steps to produce three instances with different line pool sizes. The Dutch and China instance number 3 contains all lines, i. e., all paths that are possible in the network. For instance Potsdam1998 we defined all nodes as termini that are termini of operating lines in the associated year. The line pools for the Potsdam network of 1998 are generated similar as for the other instances with one exception, namely, we considered the given turning restrictions on crossings. The termini for Potsdam2010 were explicitly given. The line pool contains all possible lines that fulfill the ViP requirements as well as given lines for regional and commuter traffic which are not operated by ViP. The instance Potsdam2010+ describes the final setting that we used for optimizing in the project Stadt+, compare with Chapter 9. It has the same set of lines as Potsdam2010, but restricts the set of possible frequencies for all tram lines. It further includes all additional requirements given by ViP such as a minimum frequency requirement for a set of stations/stops. These additional constraints are discussed in Chapter 9. We do not consider the two big instances Anaheim and Chicago from the Transportation Network Test Problems Library of Bar-Gera [98] as we have done for the SCP, because they lead to very artificial looking super-large scenarios of questionable significance. Indeed, taking all nodes as termini, as for the Dutch, China, and SiouxFalls instances, produces hundreds of thousands of line paths, even if we allow only shortest paths. The real-world instance Potsdam2010 has only 3 433 lines, although the underlying transportation network is larger than that of Anaheim. We feel that a substitution of missing data by a complete enumeration of all possibilities exceeds its reasonable limits here.

Recall that we assume lines to operate in both directions. This means that one-way streets are an obstacle; they only occur in the Potsdam data. We assume that relaxing one-way streets has little influence on the optimal solution. More precisely, we allow to operate lines on one-way streets in both directions in the optimization model and, if necessary, reroute the lines to satisfy the geographical restrictions afterwards. We further implemented some checking routines for the given data as well as several routines

to remove dispensable information before starting with the optimization. Note that we do not modify the data. The preprocessing may reduce the size of the public transportation network. It involves the following ideas:

- We remove isolated nodes. Sometimes the data contains nodes in the network that are not connected to other nodes. These nodes are called isolated.
- We remove unused OD nodes and uncovered network nodes with all incident arcs. Unused OD nodes have no traffic, i. e., no supply or demand is given for these nodes. Uncovered network nodes are not contained in any line and cannot be reached by foot, i. e., are also not relevant for the passenger traveling paths. Checking for uncovered network nodes requires that the set of lines is given.
- We remove parallel edges/arcs of the same transportation mode in the public transport network and keep the edge/arc with the smallest travel time.
- We remove a node and its incident arcs if it is a non-terminus and non-OD node with exactly one neighbor of the same transportation mode.
- We contract intermediate nodes, i. e., nodes v with exactly two neighbors u and w of the same mode; the arcs (u, v) and (v, w) are replaced by the arc (u, w) with travel time $\tau_{uw} = \tau_{uv} + \tau_{vw}$ and length $l_{uw} = l_{uv} + l_{vw}$.
- We contract nodes that do not correspond to stops/stations, i. e., passengers cannot change lines at these nodes. Such nodes correspond, e. g., to crossings or light signals.

The preprocessing is mainly important for real-world data since those data often include more information than needed (e. g., the position of light signals) and is prone to inconsistencies. Indeed, the preprocessing reduces only (but largely) the Potsdam network. For Potsdam1998 (Potsdam2010) the number of nodes can be reduced by around 55.8% (61.8%), the number of OD nodes can be reduced by around 3.6% (7.8%), and the number of edges can be reduced by around 16.5% (37.3%).

Table 8.1 gives some statistics about the test instances after preprocessing. The columns labeled $|D|$, $|V|/|V_O|$, $|A|$, and $|\mathcal{L}|$ list the number of OD pairs with nonzero demand, nodes/OD nodes (note that $V_O \cap V = \emptyset$), arcs, and lines after the preprocessing. The columns labeled x-vars and cons (dc-cons) give the number of x-variables for the frequency set $\mathcal{F} = \{3, 6, 9, 18\}$ and the number of constraints for the models (BD), (DC), and (UT) after the presolving of SCIP. The numbers in brackets in column cons (dc-cons) list the number of additional direct connection constraints of models (DC) and (UT). We will come back to this in Section 8.5 when explaining computational results. The last two columns give the number of nodes and arcs in the corresponding change-and-go graph. Especially the transfer arcs can highly enlarge the size of the change-and-go graph since we have to insert transfer arcs for every two lines that operate the same station. If an important transfer station is contained in many lines, say k lines, then a straight forward implementation of the change-and-go graph includes a complete graph on k nodes for this transfer station. An equivalent but more efficient construction is based on inserting an additional auxiliary transfer node. This requires only $2k$ transfer arcs instead of $k(k-1)$ arcs. For $k \geq 4$ this is an improvement since $2k < k(k-1)$, $k \geq 4$. In formulas, the idea

Table 8.1: Statistics on the line planning instances. The first column lists the instances. The next three columns list the number of nonzero OD pairs, number of nodes/OD nodes, and number of edges of the preprocessed passenger routing graph. Columns 5, 6, and 7 show the size of the line pool and give statistics on the number of variables and constraints for the models (BD), (DC), and (UT) after the presolving of SCIP. The last two columns list the number of nodes and arcs of the change-and-go graph.

instance	$\|D\|$	$\|V\|/\|V_O\|$	$\|A\|$	$\|\mathcal{L}\|$	x-vars	cons (dc-cons)	$\|\mathcal{V}\|$	$\|\mathcal{A}\|$
Dutch1	420	23/23	106	402	1 608	1 080 (1 832)	1 880	10 200
Dutch2	420	23/23	106	2 679	10 716	3 341 (7 544)	18 573	105 804
Dutch3	420	23/23	106	7 302	29 208	7 945 (9 736)	61 270	352 740
China1	379	20/20	98	474	1 896	1 178 (2 754)	2 742	15 264
China2	379	20/20	98	4 871	19 484	5 457 (8 162)	44 431	256 604
China3	379	20/20	98	19 355	77 420	19 931 (12 443)	229 702	1 339 262
SiouxFalls1	528	24/24	124	866	3 464	1 779 (4 400)	4 868	27 188
SiouxFalls2	528	24/24	124	9 397	37 588	10 197 (16 844)	77 078	443 386
SiouxFalls3	528	24/24	124	15 365	61 460	16 145 (21 220)	146 515	848 072
Potsdam1998a	7 734	344/107	2 746	207	756	8 894 (3 538)	2 615	20 218
Potsdam1998b	7 734	344/107	2 746	1 907	7 560	10 606 (25 916)	41 474	366 153
Potsdam1998c	7 734	344/107	2 746	4 342	17 300	12 981 (38 107)	126 666	1 096 732
Potsdam2010	4 443	851/236	5 542	3 433	13 696	10 158 (33 166)	93 897	609 411
Potsdam2010+	4 443	851/236	5 542	3 433	13 624	10 350 (33 166)	93 897	609 411

is as follows. Let $\mathcal{L}(u)$ be the set of lines containing node u.

- If $|\mathcal{L}(u)| < 4$ then we insert transfer arcs $((u, \ell_i)(u, \ell_j))$ and $((u, \ell_j)(u, \ell_i))$ for all lines $\ell_i, \ell_j \in \mathcal{L}(u)$.
- If $|\mathcal{L}(u)| \geq 4$ then we insert an additional auxiliary node v_u and transfer arcs $((u, \ell), v_u)$ and $(v_u, (u, \ell))$ for all lines ℓ containing u. The transfer penalties are only added on the arcs $(v_u, (u, \ell))$.

The average number of lines containing the same node, i. e., for which transfer arcs have to be included, is 80 for Dutch1, 135 for China1, 200 for SiouxFalls, 13 for Potsdam1998a, and 367 for Potsdam2010, i. e., the above construction is already an improvement for the smallest instances.

We tried to compute the root LP relaxation of model (CG) for all instances listed in Table 8.1 on computers with an Intel(R) Xeon(R) CPU X5672 with 3.20 GHz, 12 MB cache, and 48 GB of RAM. The root LP could be solved for Dutch1, Dutch2, China1, China2, SiouxFalls1, and Potsdam1998a within 5 hours. The root LP of all other instances, which make 7 instances out of 13, could not be solved within 10 hours. We, therefore, restrict ourselves to computing solutions for (BD), (DC), and (UT) and describe the application of our solution methods to these models.

8.2 Primal Heuristics

The objective functions of the line planning models contain two competing parts, the costs of a line plan and the travel times of the passengers. In this section, we introduce primal heuristics to round the line variables according to one part in the objective. The first idea is to round the frequencies of the lines such that the travel times of the passengers do not increase. The second idea is to reduce the number of lines in an LP solution as long as the increase in travel time is smaller than the cost of the deleted line.

We distinguish between fast rounding heuristics and diving heuristics. All heuristics are independent of the computation of direct travelers and can therefore be applied to models (BD), (DC), and (UT).

Rounding Heuristics

Rounding heuristics set all integer or binary variables to an integral value in a first step and then recompute the resulting LP to find the best values for all continuous variables in a second step, see, e. g., Berthold [7]. Such rounding heuristics are usually fast. In the following we describe a rounding heuristic for line planning models. We assume that we are given a feasible LP solution.

The simplest idea is to set the frequency for each fractional line to the next integral frequency which is higher than or equal to the sum of the frequencies of the line in the LP solution. In this way, the passenger paths do not have to be changed since the so-defined capacities of the lines suffice to cover all passenger paths. The travel times of the resulting IP solution are, therefore, not higher than the travel times of the initial LP solution. We slightly improve this idea by setting the frequency of a fractional line to the minimal value such that the resulting capacity reaches either the capacity of the line of the LP solution or the maximal number of passengers on an arc of the line that are not covered by lines whose frequencies have already been fixed. Note that an arc can be contained in several lines and the sum of the capacity of all these lines has to be greater than or equal to the number of passengers on the arc. The lines are rounded in the order as they are defined. We call this heuristic `round`. We further modify this heuristic and consider a different order of the lines. The order is induced by a *weight* $s(\ell)$ for each line $\ell \in \mathcal{L}$. We will use as weights the sums of the pseudocosts of the associated variables (for all frequencies) of the lines. A pseudocost is a statistic on the cost value of the variable if it is rounded down, i. e., these pseudocosts are getting more meaningful during the branching process. We get them by a SCIP method. The lines are sorted in decreasing order w. r. t. their weights, and we first round the line with the highest weight. We call this modification `roundSorted`. Algorithm 8.10 gives a detailed description of the heuristic. Note that it starts by reducing the arc weights by the capacity of all integral line variables before it continues with the rounding of fractional lines. We also considered to sort the lines according to the number of passengers or the

Algorithm 8.10: Algorithm for heuristic `roundSorted`. The algorithm for heuristic `round` is the same without the sorting step in line 13.

Input : LP solution (x^*, y^*) of (BD)/(DC)/(UT); weights $s(\ell)$ for all $\ell \in \mathcal{L}$.
Output: IP solution $(x^\star, y^*)$ of (BD)/(DC)/(UT).

1 Set arc weights $w_a := \sum_{p:a\in p} y^*_p$, $\bar{\mathcal{L}} := \emptyset$
2 **for** $\ell \in \mathcal{L}$ **do**
3 **for** $f \in \mathcal{F}$ **do**
4 **if** $x^*_{\ell,f} = 1$ **then**
5 $x^\star_{\ell,f} := 1$
  `// reduce arc weights according to the capacity of line` ℓ
6 $w_a := \max\{0, w_a - \kappa_{\ell,f}\}$ for all $a : e(a) \in \ell$
7 **end**
8 **if** $x^*_{\ell,f}$ is fractional **then**
9 $\bar{\mathcal{L}} := \bar{\mathcal{L}} \cup \{\ell\}$
10 **end**
11 **end**
12 **end**
13 Sort lines in $\bar{\mathcal{L}}$ such that $s(\ell_1) \geq s(\ell_2) \geq \ldots \geq s(\ell_{|\bar{\mathcal{L}}|})$
14 **for** $i = 1 \ldots |\bar{\mathcal{L}}|$ **do**
15 $\tilde{f} := \min_{f\in\mathcal{F}\cup\{0\}} \left\{ \kappa_{\ell_i,f} \geq \sum_{f\in\mathcal{F}} \kappa_{\ell_i,f} x^*_{\ell_i,f} \text{ or } \kappa_{\ell_i,f} \geq \max\{w_a \,|\, a : e(a) \in \ell_i\} \right\}$
16 $x^\star_{\ell_i,\tilde{f}} := 1$
17 $w_a := \max\{0, w_a - \kappa_{\ell_i,\tilde{f}}\}$ for all $a : e(a) \in \ell_i$ `// update arc weights`
18 **end**

number of different passenger paths using arcs of the line. However, this sorting is more time consuming and yields no better results.

Diving Heuristics

A diving heuristic fixes a subset of variables that are not fixed so far and resolves the modified LP, see again Berthold [7]. It then fixes the next subset of variables and so on. The procedure stops if one of the following conditions is satisfied:

- The modified LP is infeasible.
- The objective value of the modified LP is not better than the best known IP solution.
- All variables are fixed.

We will describe two diving heuristics for our line planning problem which always yield feasible solutions for the line planning models (BD), (DC), and (UT) as defined in Chapter 6. Both heuristics try to find a promising subset of lines with positive LP value,

i. e., all lines with 0 frequencies in the LP solution are fixed to 0. The first rounds the variables successively up, the second rounds the variables successively down.

Algorithm 8.11: Algorithm for heuristic `diveSorted`.

Input : LP solution (x^*, y^*) of (BD)/(DC)/(UT); weights $s(\ell)$ for all $\ell \in \mathcal{L}$.
Output: IP solution $(x^\star, \cdot)$ of (BD)/(DC)/(UT).

$\bar{\mathcal{L}} := \{\ell \in \mathcal{L} \,|\, \exists\, f \in \mathcal{F} \text{ with } 0 < x^*_{l,f} < 1\}$, $x^\star_{\ell,f} := x^*_{\ell,f}\ \forall \ell \in \mathcal{L} \setminus \bar{\mathcal{L}}$
while $\bar{\mathcal{L}} \neq \emptyset$ **do**
  $\tilde{\ell} = \text{argmax}\ \{s(\ell) \,|\, \ell \in \bar{\mathcal{L}}\}$
  $\tilde{f} := \min_{f \in \mathcal{F}}\{\kappa_{\tilde{\ell},f} \geq \sum_{f \in \mathcal{F}} \kappa_{\tilde{\ell},f} x^\star_{\tilde{\ell},f}\}$
  $x^\star_{\tilde{\ell},\tilde{f}} := 1$ (constraint for modified LP)
  $(x^*, y^*) :=$ solution of modified LP (for $x^\star$)
  $\bar{\mathcal{L}} := \{\ell \in \mathcal{L} \,|\, \exists\, f \in \mathcal{F} \text{ with } 0 < x^*_{l,f} < 1\}$ `// update` $\bar{\mathcal{L}}$
end

Algorithm 8.12: Algorithm for heuristic `roundFreqObj`.

Input : LP solution (x^*, y^*) of (BD)/(DC)/(UT).
Output: IP solution $(x^\star, \cdot)$ of (BD)/(DC)/(UT).

$\bar{\mathcal{L}} := \{\ell \in \mathcal{L} \,|\, \exists\, f \in \mathcal{F} \text{ with } 0 < x^*_{l,f} < 1\}$, $x^\star_{\ell,f} := x^*_{\ell,f}\ \forall \ell \in \mathcal{L} \setminus \bar{\mathcal{L}}$
while $\bar{\mathcal{L}} \neq \emptyset$ **do**
  $\tilde{\ell} := \text{argmin}\ \{\sum_{f \in \mathcal{F}} x^*_{\ell,f} \cdot f : \ell \in \bar{\mathcal{L}}\}$
  $\tilde{f} := \min_{f \in \mathcal{F}}\{\kappa_{\tilde{\ell},f} \geq \sum_{f \in \mathcal{F}} \kappa_{\tilde{\ell},f} x^*_{\tilde{\ell},f}\}$
  $C_{\tilde{\ell}} = c_{\tilde{\ell},\tilde{f}} - \sum_{f \in \mathcal{F}} c_{\tilde{\ell},f} x^*_{\tilde{\ell},f}$ `// additional operating cost`
  $x^\star_{\tilde{\ell},f} := 0$ for all $f \in \mathcal{F}$ (constraint for modified LP)
  $(x^{**}, y^{**}) :=$ solution of modified LP for $(x^\star)$
  if $v_{LP}(x^{**}, y^{**}) > v_{LP}(x^*, y^*) + 0.9 \cdot C_{\tilde{\ell}} \cdot \lambda$ **then**
    $x^\star_{\tilde{\ell},\tilde{f}} := 1$ `// backtracking`
    $(x^{**}, y^{**}) :=$ solution of modified LP for $(x^\star)$
  end
  $(x^*, y^*) := (x^{**}, y^{**})$
  $\bar{\mathcal{L}} := \{\ell \in \mathcal{L} \,|\, \exists\, f \in \mathcal{F} \text{ with } 0 < x^*_{l,f} < 1\}$ `// update` $\bar{\mathcal{L}}$
end

The idea of the first diving heuristic is the same as for `roundSorted`. It just resolves the LP after each fixing of a line to a frequency. More precisely, we choose a line ℓ whose frequency is not fixed so far and for which the weight $s(\ell)$ is maximal. We set the frequency to the minimal value such that the capacity of the line of the (current) LP solution is reached. Note that this capacity is always smaller than or equal to the maximal number of passengers on an arc of ℓ that are not covered by lines already fixed if we have no other constraints than those in the models (BD),(DC), and (UT). This is

Table 8.2: Time in seconds and integrality gap for the four different primal heuristics after solving the root node of model (DC) ($\lambda = 0.8$).

	round		roundSorted		diveSorted		diveObj	
instance	time	gap	time	gap	time	gap	time	gap
N1	<1	1.34%	<1	0.72%	1	0.61%	2	0.53%
N2	16	2.93%	15	1.66%	26	1.80%	36	1.38%
N3	54	2.75%	55	1.74%	92	2.02%	102	1.41%
China1	1	2.28%	1	2.35%	2	0.79%	3	0.95%
China2	93	4.33%	92	1.50%	124	0.54%	146	1.09%
SiouxFalls1	3	0.28%	2	0.28%	4	0.22%	6	0.11%
SiouxFalls2	88	0.24%	103	0.24%	166	0.24%	198	0.12%
SiouxFalls3	209	0.20%	209	0.20%	1859	0.20%	427	0.13%
Potsdam1998a	54	0.48%	56	0.46%	126	0.33%	550	0.27%
Potsdam1998b	501	1.74%	500	1.15%	1803	0.55%	3380	0.38%
Potsdam1998c	1788	1.90%	1784	1.01%	8055	0.56%	7909	0.40%
Potsdam2010	331	3.17%	334	2.75%	1279	2.24%	894	1.26%
Potsdam2010+	239	-	245	-	4 497	16.84%	753	0.93%

Table 8.3: Time in seconds and integrality gap for the four different primal heuristics after solving the root node of model (BD) ($\lambda = 0.8$).

	round		roundSorted		diveSorted		diveObj	
instance	time	gap	time	gap	time	gap	time	gap
N1	<1	1.10%	<1	0.65%	<1	0.38%	<1	0.88%
N2	1	1.37%	1	1.10%	1	0.36%	2	0.60%
N3	3	1.32%	2	1.01%	3	0.36%	4	0.79%
China1	< 1	1.32%	<1	1.10%	<1	0.27%	<1	0.66%
China2	1	1.37%	1	1.17%	2	0.27%	2	0.56%
China3	8	1.25%	8	1.34%	11	0.26%	11	0.73%
SiouxFalls1	<1	0.10%	<1	0.10%	<1	0.10%	<1	0.03%
SiouxFalls2	6	0.08%	5	0.08%	8	0.08%	7	0.06%
SiouxFalls3	12	0.11%	13	0.11%	16	0.11%	15	0.02%
Potsdam1998a	19	0.49%	19	0.45%	27	0.29%	223	0.27%
Potsdam1998b	9	0.97%	9	0.97%	18	0.38%	168	0.44%
Potsdam1998c	12	0.90%	12	0.81%	26	0.42%	202	0.43%
Potsdam2010	8	4.04%	9	4.41%	19	2.76%	82	1.96%
Potsdam2010+	10	-	10	-	17	2.75%	62	1.00%

in contrast to the rounding heuristic `roundSorted` and because we resolve the LP after each fixing. A detailed description of the heuristic gives Algorithm 8.11. We call this heuristic `diveSorted`.

The aim of the second diving heuristic is to reduce the number of lines of the LP solution. In each step we choose the line with the smallest fractional value and set all variables of this line to 0. Let x^* be the LP solution for the line variables. Let further $C_\ell = c_{\ell,\tilde{f}} - \sum_{f\in\mathcal{F}} c_{\ell,f} x^*_{\ell,f}$ be the additional operating cost for line ℓ, i. e., the increase in the cost when the line is rounded to frequency $\tilde{f} := \min_{f\in\mathcal{F}}\{\kappa_{\ell,f} \geq \sum_{f\in\mathcal{F}} \kappa_{\ell,f} x^*_{\ell,f}\}$. If

the objective value of the resulting LP exceeds the old objective value plus 90% of the additional operating cost, we set the variable of the line with frequency $\tilde{f}$ to 1. A detailed description of the heuristic is given by Algorithm 8.12. We call this heuristic `diveObj`.

Tables 8.2 and 8.3 show computation times and integrality gaps for the different heuristics for model (DC) and (BD), respectively. For model (DC) we skip the China3 instance, since it can only be solved if a start solution is given. It can be seen that the best solutions are found by the diving heuristics which also take much more computation time than the rounding heuristics; the increase in computation times can be up to a factor of 10, on average it is 2. For model (DC), the heuristic `diveObj` finds the best solutions for all instances but the China instances. It also uses only little more time than the heuristic `diveSorted` with the exception of large instances, e. g., Potsdam1998c and Potsdam2010. For the Potsdam2010+ instance, only the heuristic `diveObj` finds a good solution in acceptable time; we cannot explain why the heuristic `diveSorted` needs so much more time than `diveObj` but several computations lead to a similar result. The rounding heuristics do not consider the additional constraints, such as the minimum frequency requirement, which are included in Potsdam2010+. Hence, the fixing of the variables generally does not yield a valid solution (in the root node). In model (BD) the best solutions are mainly found by the heuristic `diveSorted`. However, extensive computations show that applying the three heuristics `roundSorted`, `diveSorted`, and `diveObj` in the root node and the two heuristics `roundSorted` and `diveObj` every 10 th and 5 th depth, respectively, is a good choice for both models. We will use this strategy in the following. For the Potsdam2010+ instance we will not consider the heuristic `diveSorted`; the rounding heuristic will be included because it needs not much time and there is a little chance that it yields a valid solution. We will use the same choice also for model (UT) without testing other calibrations explicitly for this model.

8.3 Cutting Planes

In the following, we will analyze different strategies to find additional cutting planes based on the concepts of Chapter 7. Since the separation problems for the band inequalities and the Steiner partition (band) inequalities are $\mathcal{NP}$-hard, we consider heuristic approaches to find promising capacitated cutset inequalities and violated Steiner partition inequalities; these approaches are based on solution approaches for the Steiner connectivity problem, see Chapter 2. We will see that the main effects on the dual bound can be achieved by some improved line cutset inequalities and some mixed integer rounding inequalities on capacitated cutset inequalities added in a preprocessing step. We additionally identified further violated inequalities that strengthen the relation between passenger path variables and line variables. These inequalities and the preprocessing cuts yield, in general, a remarkable increase of the dual bounds.

Max Flow Separation and Preprocessing Cuts. The separation problem for violated line cutset inequalities (7.10) is equal to the separation problem for Steiner path

cut constraints in the cut formulation of the Steiner connectivity problem, compare with Section 2.1.1, i. e., it can be solved in polynomial time by a max flow algorithm in the Steiner connectivity digraph, see Section 1.2. Here, terminal nodes correspond to OD nodes and paths to lines. Similar as for the Steiner connectivity problem, the construction of the Steiner connectivity digraph is too time- and memory consuming. Therefore, we relax the problem and search for violated "weak line cutsets", i. e., we use a max flow algorithm in the passenger routing graph in which the capacity of an arc is set to the sum of the line variables (x-variables) containing this arc. The advantage of this method is that we do not have to construct the Steiner connectivity digraph. The disadvantage is that a line may be counted several times in the computed cut. If the so-defined value of a cut that disconnects an OD pair is less than 1, we compute the total travel demand over this cut. In this way, we get a capacitated cutset inequality (7.9). We use this inequality to generate improved line cutset inequalities (7.11) and to apply mixed integer rounding, compare with Chapter 7. We call all violated constraints that are generated in this way *max flow cuts*.

Recall that we initialized the computations for the Steiner connectivity problem by (directed) Steiner path cuts around the terminal nodes. In the line planning setting, this approach corresponds to considering a cut around each OD node with positive supply or demand, i. e., a cut $\delta^-(W)/\delta^+(W)$ in the passenger routing graph G with $W = \{s\} \cup \{v \in V : (s,v) \in A_O\}$ for each $s \in V_O$ such that there exists $t \in V_O$ with $d_{st} > 0$ or $d_{ts} > 0$. We, further, consider *OD bridges* in the passenger routing graph. An arc a is an OD bridge if its removal would disconnect at least one OD pair, i. e., there exist $W \subset V$, $s \in W$, $t \notin W$ such that $d_{st} > 0$ and $\delta^-(W) = \{a\}$. In both cases we compute the total travel demand for the resulting cut. This yields a capacitated cutset inequality. We again use this inequality to generate improved line cutset inequalities (7.11) and to apply mixed integer rounding. We call all additional constraints that are generated in this way *preprocessing cuts*.

One short remark to the mixed integer rounding technique. Theorem 7.12 limits the set of multipliers that lead to different mixed integer rounding inequalities. We decided to divide the capacitated cutset inequality (7.9) by each coefficient of the left hand side. This is also common practice in, e. g., SCIP. We usually have few different coefficients in the capacitated cutset inequality (7.9), namely, we often have $\alpha_\ell \leq 3$, $|\mathcal{F}| \leq 5$, and $\kappa_{\ell_1,f} \neq \kappa_{\ell_2,f}$ only if ℓ_1 and ℓ_2 do not have the same transportation mode, i. e., multiplying the capacitated cutset inequality (7.9) with $\frac{1}{\alpha_\ell \kappa_{\ell,f}}$ for each coefficient $\alpha_\ell \kappa_{\ell,f}$ in (7.9) and then considering the mixed integer rounding inequality (7.6) can be done in constant time.

In computational experiments, we found out that including preprocessing cuts or max flow cuts increases the dual bound. However, we did not find any further violated max flow cuts if we included all preprocessing cuts. Note that the default settings of SCIP already provide several cutting plane separation methods, e. g., mixed integer rounding cuts, Gomory cuts, and flowcover cuts, see Achterberg [1] and Wolter [107]. These separation methods might imply some valid inequalities that we also would find by our

separation methods. Anyway, we include the preprocessing cuts but no flow cuts in the following.

Separation by Graph Shrinking. We have considered a heuristic separation for the Steiner partition inequalities, the SPI separation, in Section 4.1.1, which is based on a graph shrinking procedure. A valid Steiner partition band inequality for the line planning problem can be derived from the trivial band, i. e., $B(\ell) = 0$ for all $\ell \in \mathcal{L}$. We, therefore, consider this separation method also for the line planning problem.

We further consider a slightly different graph shrinking procedure. The idea is similar to a shrinking procedure considered in telecommunication network design to determine cuts, see, e. g., Raack et al. [85]; Dix [44] considered a similar shrinking procedure for line planning with continuous frequencies. It works as follows. For a given LP solution (x^*, y^*) of the line planning problem let $\omega_a := \sum_{\ell \in \mathcal{L}: e(a) \in \ell} \sum_{f \in \mathcal{F}} \kappa_{\ell,f} x^*_{\ell,f} - \sum_{p \in \mathcal{P}, a \in p} y_p$, for all $a \in A$, be some arc weights in the passenger routing graph representing the unused capacity of an arc. Here, we set $w_a = \infty$ for all OD arcs $a \in A_O$. Sort the arcs in decreasing order of their weights and shrink the arcs in this order until the graph contains $N \in \mathbb{N}$ components and each of the N components contains at least one OD node. We then enumerate all cuts in the shrunk graph and consider the capacitated cutset inequality for each cut. We again use this inequality to generate the improved cutset inequality (7.11) and to apply mixed integer rounding. We call this separation method *capacitated shrinking separation*.

The SPI separation method, that was quite good for the Steiner connectivity problem, did not find any violated cuts in all our computations for the line planning problem. The capacitated shrinking separation method found violated cuts only for few instances. We considered the values 3, 4, and 5 for the number of components N. We nevertheless apply this latter separation method in the root node and set $N = 5$.

Coupling Passenger Paths and Lines. All cuts considered so far were motivated by the investigation of the line planning polytope, compare with Chapter 7. They describe possible line capacities to support the given demand, i. e., the cuts only involve the x-variables associated with line paths. In the following, we will consider valid inequalities that also incorporate the passenger flow which is to computed. Inequalities containing the y-variables associated with passenger paths usually affect the pricing problem of the passenger variables. We, nevertheless, consider in the following inequalities that couple passenger path variables and line path variables since these inequalities, albeit slowing the passenger path pricing algorithm down, have a strong effect on the dual bound.

The idea is as follows. If an arc a is covered by a passenger path, at least one line has to cover arc a, i. e., in formulas,

$$\sum_{p \in \mathcal{P}_{st}: a \in p} y_p > 0 \Rightarrow \sum_{\ell \in \mathcal{L}: e(a) \in \ell} \sum_{f \in \mathcal{F}} x_{\ell,f} \geq 1 \qquad \forall\, (s,t) \in D, \forall\, a \in A. \tag{8.1}$$

We, further, have

$$\sum_{p\in\mathcal{P}_{st}} y_p = d_{st} \Rightarrow \sum_{\substack{p\in\mathcal{P}_{st}\\ a\in p}} y_p \leq d_{st} \Rightarrow \sum_{\substack{p\in\mathcal{P}_{st}\\ a\in p}} \frac{y_p}{d_{st}} \leq 1 \quad \forall\,(s,t)\in D,\,\forall\,a\in A. \tag{8.2}$$

Combining (8.1) and (8.2) yields

$$\sum_{p\in\mathcal{P}_{st}:a\in p} \frac{y_p}{d_{st}} \leq \sum_{\ell:e(a)\in\ell} \sum_{f\in\mathcal{F}} x_{\ell,f} \qquad \forall\,(s,t)\in D,\,\forall\,a\in A. \tag{8.3}$$

The same idea can also be applied to direct connection passenger paths, i. e., if an arc a is covered by direct st-connection travelers, at least one direct st-connection line has to cover arc a. We then get the following constraints

$$\sum_{p\in\mathcal{P}^{0+}_{st}(a)} \frac{y_{p,0}}{d_{st}} \leq \sum_{\ell\in\mathcal{L}^{0}_{st}(a)} \sum_{f\in\mathcal{F}} x_{\ell,f} \qquad \forall\,(s,t)\in D,\,\forall\,a\in A_{st}. \tag{8.4}$$

We call inequalities (8.3) *coupling constraints* and inequalities (8.4) *coupling constraints for direct connections* or just *dc-coupling constraints*. Violated coupling constraints (8.3) can be added to model (BD) as well as to model (DC). In the latter case, we have $y_p = y_{p,0} + y_{p,1}$. However, computations show that it is better to search only for violated dc-coupling constraints (8.4) in model (DC). We will also separate dc-coupling constraints (8.4) in model (UT).

Including dc-coupling constraints (8.4) in model (DC) with associated dual variable ξ only adds an additional summand in the first inequality of the dual program (6.55). The reduced cost for a direct connection st-passenger path changes as follows

$$\bar{\tau}_{p,0} = -\pi_{st} + \sum_{a\in p}\left(\mu_a + \nu_{a,[s,t]_a} + \frac{1}{d_{st}}\xi_a + (1-\lambda)\tau_a\right) \quad \forall\,(s,t)\in D,\,\forall\,p\in\mathcal{P}^{0+}_{st}, \tag{8.5}$$

i. e., only the arc weights differ. The pricing problem remains to find a shortest st-rdcpath with arc weights now set to $\mu_a + \nu_{a,[s,t]_a} + \frac{1}{d_{st}}\xi_a + (1-\lambda)\tau_a$. The pricing problem for model (UT) with dc-coupling constraints (8.4) also only differs in the arc weights.

The changes in the pricing problem for model (BD) caused by the coupling constraints (8.3) are more drastic. The reduced cost for a passenger path variable becomes

$$\bar{\tau}_{p} = -\pi_{st} + \sum_{a\in p}\left(\mu_a + \frac{1}{d_{st}}\xi_a + (1-\lambda)\tau_a\right) \quad \forall\,(s,t)\in D,\,\forall\,p\in\mathcal{P}_{st}, \tag{8.6}$$

i. e., now, the arc weights depend on the considered OD pair which was not the case without coupling constraints (8.3). This means that the computation time for the pricing problem for model (BD) increases. Without coupling constraints (8.3) we could compute a shortest paths tree from one node to all other nodes. Including coupling constraints (8.3), we have to compute the shortest path for every two nodes individually since the arc weights may differ.

Table 8.4: Time in seconds and dual bound for model (DC) after solving the root node without additional cuts, with preprocessing cuts, and with coupling constraints for passenger paths and lines for $\lambda = 0.8$. The SCIP default separation routines are included.

	without cuts		prepro cuts		coupl. cons.	
instance	time	dual	time	dual	time	dual
N1	<1	2606076	1	2607200	2	2611485
N2	13	2592857	15	2594781	49	2606166
N3	50	2592026	65	2594032	157	2605251
China1	1	2599683	3	2599718	4	2601112
China2	89	2510624	102	2510738	136	2511057
SiouxFalls1	2	645226	4	645226	4	645247
SiouxFalls2	81	639191	71	639191	101	639214
SiouxFalls3	193	638578	228	638578	326	638580
Potsdam1998a	48	1020617	49	1020869	81	1021189
Potsdam1998b	459	982631	500	982838	2027	984141
Potsdam1998c	1696	981268	1942	981476	7787	982778
Potsdam2010	302	207423	202	207958	4079	208743
Potsdam2010+	227	210976	244	211003	7984	211911

Table 8.5: Time in seconds and dual bound for model (BD) after solving the root node without additional cuts, with preprocessing cuts, and with coupling constraints for passenger paths and lines for $\lambda = 0.8$. The SCIP default separation routines are included.

	without cuts		prepro cuts		dc-coupl. cons.	
instance	time	dual	time	dual	time	dual
N1	<1	2583227	1	2588250	1	2588315
N2	<1	2583152	4	2587685	4	2587691
N3	2	2583152	5	2587310	6	2587310
China1	<1	2510271	2	2510669	2	2510728
China2	<1	2509617	3	2509880	3	2509880
China3	7	2509409	9	2509526	9	2509526
SiouxFalls1	<1	641703	1	641703	1	641703
SiouxFalls2	4	639173	7	639173	7	639173
SiouxFalls3	11	638578	14	638578	15	638578
Potsdam1998a	17	971252	20	971947	32	972063
Potsdam1998b	6	956775	10	957460	51	957660
Potsdam1998c	9	956695	13	957374	99	957538
Potsdam2010	7	195477	6	197207	115	197412
Potsdam2010+	6	198945	7	199253	1340	199929

Tables 8.4 and 8.5 show times and dual bounds for the root LP without additional cuts, with the preprocessing cuts, and with preprocessing cuts and the coupling constraints for the models (DC) and (BD), respectively. For model (DC), we skip the China3 instance, since it can only be solved if a start solution is given. The capacitated shrinking method finds few cuts for only few instances with very little effect on the dual bound. The results are, therefore, not presented in the tables. Nevertheless, we will include this separation method (only in the root node) in our computations, since it takes little separation time.

The preprocessing constraints improve the root LP of model (DC) by around 0.02% for the Potsdam1998 instances and by around 0.07% for the Dutch instances. The biggest improvement is for Potsdam2010 with around 0.26%. The improvement for the China instances is in the order of per mill; the SiouxFalls instances could not be improved. The coupling constraints (8.4) together with the preprocessing cuts improve the root LP value by around 0.1% to 0.5% for the Dutch and Potsdam instances. The improvement for the China and SiouxFalls instances stays in the order of per mill. The amount of the improvement for model (BD) is similar, it is however mainly achieved by the preprocessing cuts and little by the coupling constraints (8.3). We include the separation of the coupling constraints in our computations in each branching node.

8.4 Branching Rule

The SCIP framework offers only a branching on variables. However, the SOS frequency constraints, e. g., (6.40) for model (DC), produce unbalanced the branch-and-bound trees with this strategy. This is due to the fact that $x_{\ell,f} = 0$ prohibits just one frequency for the line while $x_{\ell,f} = 1$ fixes one frequency with the effect that all other frequencies for this line have to be 0. A better strategy is to branch on the SOS frequency constraints. This *constraint branching* is a branching on the minimum resp. maximum frequency of a line, i. e., a line has to be operated with at least a minimum frequency or a line can be operated with at most a maximum frequency. This prohibits or permits a (small) set of frequencies. However, a straight forward implementation (as we did) of such an SOS constraint branching is in most cases not better than the sophisticated branching rules implemented in the SCIP framework. But we found out that a constraint branching in a broader sense leads to an improvement of the LP bound during the branching process. The idea is to branch on the number of lines with a minimum/maximum frequency on an arc. For this purpose, we included additional auxiliary *branching variables* $h_{a,i} \in \mathbb{Z}_{\geq 0}$, $a \in A$, $i \in \mathcal{F}$, that account for the number of lines with frequency greater than or equal to i on arc a, and the corresponding *branching constraints*

$$\sum_{\ell \in \mathcal{L}: e(a) \in \ell} \sum_{f \in \mathcal{F}: f \geq i} x_{\ell,f} = h_{a,i} \quad \forall\, a \in A, \forall\, i \in \mathcal{F}.$$

Including these branching variables and constraints, the branching rules implemented in the SCIP framework branch either on a variable $x_{\ell,f}$, $\ell \in \mathcal{L}$, $f \in \mathcal{F}$, or on a branching variable $h_{a,i} \in \mathbb{Z}_{\geq 0}$, $a \in A$, $i \in \mathcal{F}$. The latter choice corresponds to a constraint branching. In this way, we combine the sophisticated SCIP branching rules with a constraint branching.

Table 8.6: Transportation capacities (in passengers) and costs (in €/km) for the line planning instances according to the available modes. The networks for the Potsdam instances contain different transportation modes; the costs for the transportation modes not operated by the local public transport company of Potsdam (ViP) are set to 0. The numbers for the Potsdam instances have been set after consultation with ViP. For the year 1998 ViP gave us the capacities for a medium utilization of the transportation modes; the numbers for the year 2010 correspond to a maximum utilization for bus, tram, and ferry. The capacities for the other instances are equal to the Potsdam1998 numbers for the corresponding mode.

instances	mode capacities (in passengers/line)					costs (in €/km)				
	ferry	bus	tram	S-Bahn	IC/IR	ferry	bus	tram	S-Bahn	IC/IR
Dutch	-	-	-	-	600	-	-	-	-	100
China	-	-	-	-	600	-	-	-	-	100
SiouxFalls	-	57	-	-	-	-	1.96	-	-	-
Potsdam1998	39	57	114	536	600	0	1.96	2.34	0	0
Potsdam2010	100	125	170	536	600	0	1.96	2.34	0	0

8.5 Computational Comparison of Line Planning Models

We will now present computational results for the models (BD), (DC), and (UT). We set the parameters as follows. The possible frequencies for all lines are 3, 6, 9, and 18; this corresponds to a cycle time of 60, 30, 20, and 10 minutes in a time horizon of 3 hours. The line pools of the Potsdam instances contain also lines for regional and commuter trains. These lines are not operated by ViP and we therefore fix them to given frequencies in our computations. The number of x-variables with the above frequency set is equal for the models (BD), (DC), and (UT); Table 8.1 lists these numbers for the different instances. The number of all constraints but the direct connection constraints is also equal for all models. Table 8.1 shows these numbers for the different instances in the column cons (dc-cons). The numbers in brackets give the number of direct connection constraints which are the additional constraints in models (DC) and (UT). One can see that the number of direct connection constraints is higher than the number of all other constraints for most instances; for the bigger Potsdam1998 instances the factor is 2.5. We set the costs of the lines proportional to their lengths plus a fixed cost term that is used to reduce the number of lines. More precisely, we set $c_{\ell,f} = C + f \cdot c^i \cdot \sum_{e \in \ell} l_e$ with fixed cost $C = 100$ and operating cost c^i for line ℓ with transportation mode i as given in Table 8.6. Table 8.6 also lists capacities for the different instances. The objective weighing parameter is set to $\lambda = 0.8$ and the transfer penalty to $\sigma = 15$ minutes. This choice was influenced by the computations for the Potsdam line plan 2010, see Chapter 9. Moreover, the costs are usually of a smaller order of magnitude than the total travel times. Hence, choosing $\lambda > 0.5$ is a reasonable choice for a balanced objective function.

The instances were solved using the constraint integer programming framework SCIP version 2.1.0, see [2, 95] with Cplex 12.3 [61] as LP-solver. We used the default settings of SCIP with two exceptions. The first is to use only "fast SCIP heuristics". The second is to change the LP pricing parameter to "quickstart steepest edge" for the China instances and the Potsdam instances in all computations since the number of computed branching nodes increased for this parameter. Moreover, China3 could not be solved for models

Table 8.7: Statistics on the computations for model (DC) including branching variables (bv) and without branching variables. The columns list the instance, the number of branching nodes, the LP value at the end of the computations, the best solution value, and the integrality gap for both runs. The computation time was limited to 10 hours except for the run labeled as Potsdam2010+*, which was solved to optimality using the branching variables after 12 hours. Comparing both runs, we highlighted the better values by a green color.

	(DC) with bv				(DC) without bv			
instance	nodes	LP-val.	best sol	gap	nodes	LP-val.	best sol	gap
Dutch1	363	2613065	2613065	opt.	403 886	2613065	2613065	opt.
Dutch2	11 760	2608329	2608329	opt.	128 360	2607016	2608526	0.06%
Dutch3	32 438	2607039	2608009	0.04%	26 228	2606221	2608526	0.09%
China1	412 908	2603726	2610301	0.25%	607 403	2604021	2610826	0.26%
China2	7 082	2511251	2520677	0.38%	7 247	2511363	2522842	0.46%
China3	843	2510045	2520677	0.42%	770	2510136	2522842	0.45%
SiouxFalls1	262 557	645393	645708	0.05%	297 289	645317	645707	0.06%
SiouxFalls2	10 071	639280	639748	0.07%	10 051	639221	639536	0.05%
SiouxFalls3	4 380	638623	638811	0.03%	4 722	638581	638843	0.04%
Potsdam1998a	9 355	1021881	1022793	0.09%	3 482	1021513	1022605	0.11%
Potsdam1998b	1 281	984249	986111	0.19%	322	984423	985811	0.14%
Potsdam1998c	212	982784	984954	0.22%	105	982908	984848	0.20%
Potsdam2010	1 119	208864	209350	0.23%	455	208810	209380	0.27%
Potsdam2010+	1 028	212136	212209	0.03%	710	212056	212291	0.11%
Potsdam2010+*	1 447	212175	212175	opt.				

(DC) and (UT) using the default LP pricing algorithm which is "steepest edge", but it can be solved with a starting solution and the "quickstart steepest edge" pricing. For the Dutch and SiouxFalls instances the number of solved branching nodes not necessarily increased for the quickstart steepest edge LP pricing due to numerical problems. We, therefore, kept the default settings for these instances. Line/frequency variables were enumerated, for the pricing of the passenger path variables we implemented Dijkstra's shortest path algorithm and a labeling algorithm for the constrained shortest paths case for model (DC).

We set a time limit of 10 hours for all instances. All computations were done on computers with an Intel(R) Xeon(R) CPU X5672 with 3.20 GHz, 12 MB cache, and 48 GB of RAM.

Tables 8.7, 8.8, and 8.9 show statistics on the number of branching nodes, dual bound, best solution, and the integrality gap for models (DC), (UT), and (BD). We made computations including the additional branching variables, see Section 8.4, and without these branching variables. The results for both runs are listed in the tables. In both runs we used the solution of an instance with smaller line pool as starting solution for the instance with larger line pool, e. g., Dutch2 is initialized with a starting solution corresponding to the solution of Dutch1. We first discuss each table individually and then compare the results for the different models.

The integrality gaps for model (DC), see Table 8.7, are very small, namely, far below 1%. Around half of the instances are solved to optimality and very close to optimality.

Table 8.8: Statistics on the computations for model (UT) including branching variables (bv) and without additional branching variables. The columns list the instance, the number of branching nodes, the LP value at the end of the computations, the best solution value, and the integrality gap for both runs. The computation time was limited to 10 hours, except for instances with a superscript[2]. Computations for instances marked by the superscript[1] do not involve the diving heuristic. Potsdam1998b[2] was solved with the diving heuristic `diveObj`; we stopped the computation after the root node was solved (including separators), which took approximately 20 hours. Comparing both runs, we highlighted the better values by a green color.

	(UT) with bv				(UT) without bv			
instance	nodes	LP-val.	best sol	gap	nodes	LP-val.	best sol	gap
Dutch1	479	2613065	2613065	opt.	125 182	2612547	2613065	0.02%
Dutch2	4 379	2607596	2608329	0.03%	3 719	2606768	2609443	0.10%
Dutch3	553	2606029	2608329	0.09%	519	2605342	2609430	0.16%
China1	18 206	2602727	2610600	0.30%	19 701	2603092	2609821	0.26%
China2	207	2511074	2523100	0.48%	254	2511189	2526324	0.60%
China3	3	-	2523100	-	33	2510049	2525041	0.60%
SiouxFalls1	17 170	645592	646109	0.08%	14 503	645505	646028	0.08%
SiouxFalls2	166	639223	639763	0.07%	169	639216	639671	0.07%
SiouxFalls3	56	638580	638972	0.06%	50	638580	638986	0.06%
Potsdam1998a	4 146	1024810	1026489	0.16%	1 857	1024715	1026089	0.13%
Potsdam1998b[1]	72	982243	988394	0.63%	95	982279	990079	0.79%
Potsdam1998b[2]					1	981934	985173	0.33%
Potsdam1998c[1]	5	980745	988924	0.83%	10	980513	990051	0.97%
Potsdam2010[1]	26	210372	215240	2.31%	54	210433	214989	2.17%
Potsdam2010+[1]	20	213670	-	-	56	213500	225117	5.44%

Table 8.9: Statistics on the computations for model (BD) including branching variables (bv) and without additional branching variables. The columns list the instance, the number of branching nodes, the LP value at the end of the computations, the best solution value, and the integrality gap for both runs. The computation time was limited to 10 hours except for Potsdam2010+* which was solved after 10 hours and 10 minutes to optimality. Comparing both runs, we highlighted the better values by a green color.

	(BD) with bv				(BD) without bv			
instance	nodes	LP-val.	best sol	gap	nodes	LP-val.	best sol	gap
Dutch1	4 071 625	2590913	2591303	0.02%	6 235 018	2588611	2592090	0.13%
Dutch2	499 380	2589654	2591303	0.06%	1 655 950	2587996	2592090	0.16%
Dutch3	92 539	2588770	2591242	0.10%	230 454	2587415	2592090	0.18%
China1	1 482 899	2511481	2514522	0.12%	4 745 360	2511043	2515181	0.16%
China2	137 396	2510277	2514522	0.17%	700 773	2509956	2515181	0.21%
China3	27 325	2509639	2514522	0.21%	78 434	2509526	2515181	0.23%
SiouxFalls1	643 713	641846	641846	opt.		out of memory		
SiouxFalls2	58 115	639267	639344	0.01%	58 215	639175	639343	0.03%
SiouxFalls3	78 260	638631	638632	0.00%	32 502	638578	638704	0.02%
Potsdam1998a	9 342	973021	973839	0.08%	7 243	972491	973236	0.08%
Potsdam1998b	1 409	958119	959735	0.17%	1 546	958010	959945	0.20%
Potsdam1998c	1 912	957822	959735	0.20%	1 003	957828	959732	0.20%
Potsdam2010	2 472	198287	199004	0.36%	414	198026	199114	0.55%
Potsdam2010+	3 253	200234	200315	0.04%	3 613	200178	200353	0.09%
Potsdam2010+*	4 662	200315	200315	opt.				

Two instances can be solved to proven optimality in 10 hours if the additional branching variables are included. We solved the Potsdam2010+ instance also to proven optimality after 12 hours in the run with the additional branching variables. Without the additional branching variables only Dutch1 can be solved to proven optimality within 10 hours. Considering, however, the values of dual bound and best solution for all instances, including the additional branching variables does not result in a clear win over the computations without the branching variables. The detailed picture is as follows. For the Dutch instances, the SiouxFalls instances, and the two Potsdam2010 instances including the branching variables is a better choice than skipping these additional variables. The Dutch instances and the Potsdam2010 instances benefit most: Dutch1 and Dutch2 can be solved to proven optimality within 10 hours. Potsdam2010+ can be solved to proven optimality after 12 hours, last row in Table 8.7. Considering just the number of solved branching nodes, the computations are getting not necessarily harder if the branching variables are included. All Potsdam instances even seem to be easier to solve with the additional branching variables since in these computations the number of solved branching nodes is significantly higher. However, for Potsdam1998b and Potsdam1998c the better solutions and the better dual bounds are achieved for the computation without branching variables. A better solution was also found for Potsdam1998a in the computations without the branching variables.

Model (UT), Table 8.8, is hard to solve. For the bigger instances, i. e., Potsdam1998c, Potsdam2010, and Potsdam2010+, we had to turn off the diving heuristic `diveObj` due to a large memory consumption of more than 20 GB; even solving only the root node including the diving heuristic for Potsdam1998b (marked with[1]) took more than 20 hours. We only made this computation once, without branching variables, since we only solved the root node. A comparison of the table entries for Potsdam1998b[1] and Potsdam1998b[2] shows that the diving heuristic `diveObj` closes more than 50% of the integrality gap that we have after 10 hours without this heuristic. Comparing LP value, best solution, and integrality gap for the computations with and without the additional branching variables, the picture looks similar as for model (DC). The instance Dutch1 can be solved to proven optimality when the branching variables are included. The inclusion of the additional branching variables is also recommended for the other Dutch instances. For nearly all other instances, one method finds the better solution within 10 hours while the other computes a stronger LP bound.

In contrast to model (DC) and (UT), it is clearly preferable to include the branching variables for model (BD). In all but one instance (which is Potsdam1998c), this method computes a stronger LP bound. Moreover, the difference between the LP bound for Potsdam1998c computed with branching variables and without branching variables is very small. The SiouxFalls1 instance could even be solved to optimality with branching variables while the computations without the branching variables lead to insufficient memory after solving more than 1.3 million branching nodes since nearly as much branching nodes are still open. The best solution for SiouxFalls3 is close to be optimal for the computations with branching variables after 10 hours. Moreover, the Potsdam2010+ instance could be solved to proven optimality after 10 hours and 10 minutes if the branching vari-

ables are added. Albeit there are three instances for which the run without branching variables found better solutions, the difference to the solutions in the run with branching variables is small, namely, in the order of per mill and only for one instance 0.06%.

Comparing the results for the different models, model (UT) is much harder to solve than model (DC). The number of solved branching nodes within 10 hours with model (UT) is for each instance much lower than the number of solved branching nodes with model (DC), e. g., for China1 and China2 these numbers are for model (UT) below 5% of the numbers for model (DC). Moreover, all integrality gaps are smaller for model (DC) than for model (UT). It was not possible to compute solutions with the diving heuristic `diveObj` for model (UT) for the instances Potsdam1998b, Potsdam1998c, and both Potsdam2010 instances within 10 hours. At first glance, model (DC) seems to be harder to solve than model (BD) since the number of solved branching nodes is usually smaller for (DC) than for (BD). However, the integrality gaps are similar for (DC) and (BD). The Dutch instances 1 and 2 are even solved to optimality with model (DC) while SiouxFalls1 is solved to optimality with model (BD). Moreover, the Potsdam2010+ instance could be solved to optimality for both models after little more than 10 hours and 12 hours, respectively. Considering the computational tractability of all three models, models (BD) and (DC) perform very good and much better than model (UT) for large instances.

We evaluate the quality of the solutions of model (DC), (UT), and (BD) by computing an optimal passenger routing, including penalties for all transfers, in a change-and-go graph similar to that of Schöbel and Scholl [93]. Namely, for the best solution of the associated model we construct nodes and arcs for each line individually, i. e., the change-and-go graph contains a copy of each node and arc for every line that contains this node and arc. Further transfer arcs are added between two nodes of different lines whenever a transfer between these two lines is possible on this node. The travel time of every arc is set to the travel time of the associated arc in G, all transfer arcs are penalized by σ. We then fix the frequencies of the lines according to the best solution and route the passengers to minimize the total travel and transfer times, i. e., we compute the number of transfers for all passengers in a system optimum routing. Table 8.10 shows the results of this evaluation for the best solutions computed with model (DC), (UT), and (BD). It lists the travel times in minutes in the change-and-go graph, the costs of the line plan, the objective, i. e., 0.8·costs + 0.2·travel times, the number of direct travelers predicted by the considered model, the ("exact") number of direct travelers computed in the system optimum in the change-and-go graph, and the over- or under-estimation in percent of the predicted number of direct travelers to the "exact" number of direct travelers, i. e., this number is positive if the number of direct travelers was over estimated by the considered model.

It can be seen that the exact number of direct travelers is very close to the number of direct travelers predicted by models (DC) and (UT), which is exactly what we wanted to achieve. Although the difference is in general a little smaller for model (UT) than for model (DC), the numbers for over/underestimation are very similar for both models.

Table 8.10: Evaluation of the best solutions of models (DC), (UT), and (BD) of Tables 8.7, 8.8, and 8.9 in the change-and-go graph. The columns list travel times (in minutes), costs, objective value, number of direct travelers[1] predicted by the considered model, number of direct travelers[2] by computing a system optimum passenger routing in the change-and-go graph, and over- or under-estimation of predicted direct travelers in comparison to direct travelers in the change-and-go graph.

problem	travel times	costs	obj.	dir. trav.[1]	dir. trav.[2]	over/under est.
Dutch1 (DC)	$1.279 \cdot 10^7$	68 900	2613305	179 496	179 496	0
Dutch1 (UT)	$1.279 \cdot 10^7$	69 500	2613305	179 596	179 656	-0.03%
Dutch1 (BD)	$1.325 \cdot 10^7$	57 800	2696781	183 582	151 552	+21.11%
Dutch2 (DC)	$1.279 \cdot 10^7$	67 500	2612122	180 644	179 544	+0.61%
Dutch2 (UT)	$1.279 \cdot 10^7$	66 900	2612138	180 484	179 384	+0.61%
Dutch2 (BD)	$1.325 \cdot 10^7$	57 800	2696781	183 582	151 552	+21.11%
Dutch3 (DC)	$1.281 \cdot 10^7$	65 000	2614374	179 384	178 206	+0.66%
Dutch3 (UT)	$1.279 \cdot 10^7$	66 900	2612138	180 484	179 384	+0.61%
Dutch3 (BD)	$1.338 \cdot 10^7$	57 700	2721286	183 582	149 584	+22.73%
China1 (DC)	$1.254 \cdot 10^7$	265 287	2720586	749 461	719 112	+4.22%
China1 (UT)	$1.259 \cdot 10^7$	263 985	2729728	749 920	713 690	+5.08%
China1 (BD)	$1.527 \cdot 10^7$	234 536	3242545	759 950	532 804	+42.63%
China2 (DC)	$1.250 \cdot 10^7$	240 852	2692979	759 950	709 595	+7.10%
China2 (UT)	$1.246 \cdot 10^7$	245 656	2689361	759 902	719 802	+5.57%
China2 (BD)	$1.527 \cdot 10^7$	234 536	3242545	759 950	532 804	+42.63%
China3 (DC)	$1.250 \cdot 10^7$	240 852	2692979	759 950	709 595	+7.10%
China3 (UT)	$1.246 \cdot 10^7$	245 656	2689361	759 902	719 802	+5.57%
China3 (BD)	$1.527 \cdot 10^7$	234 536	3242545	759 950	532 804	+42.63%
SiouxFalls1 (DC)	$3.294 \cdot 10^6$	8 896	666011	360 600	357 079	+0.99%
SiouxFalls1 (UT)	$3.253 \cdot 10^6$	8 998	657781	360 600	359 300	+0.36%
SiouxFalls1 (BD)	$3.622 \cdot 10^6$	8 295	730966	360 600	335 559	+7.46%
SiouxFalls2 (DC)	$3.418 \cdot 10^6$	5 389	687911	360 600	351 812	+2.50%
SiouxFalls2 (UT)	$3.457 \cdot 10^6$	5 596	695832	359 400	352 295	+2.02%
SiouxFalls2 (BD)	$3.807 \cdot 10^6$	5 179	765451	360 600	324 182	+11.23%
SiouxFalls3 (DC)	$3.361 \cdot 10^6$	4 501	675840	360 400	354 287	+1.73%
SiouxFalls3 (UT)	$3.351 \cdot 10^6$	4 690	673894	360 600	356 153	+1.25%
SiouxFalls3 (BD)	$4.017 \cdot 10^6$	4 278	806879	360 600	312 601	+15.35%
Potsdam1998a (DC)	$5.056 \cdot 10^6$	27 509	1033218	70 574	71 038	-0.65%
Potsdam1998a (UT)	$5.042 \cdot 10^6$	27 221	1030204	71 409	71 445	-0.05%
Potsdam1998a (BD)	$5.101 \cdot 10^6$	29 081	1043379	83 926	68 825	+21.94%
Potsdam1998b (DC)	$4.834 \cdot 10^6$	30 180	990991	78 543	79 192	-0.82%
Potsdam1998b (UT)	$4.826 \cdot 10^6$	31 779	990591	79 276	79 219	+0.07%
Potsdam1998b (BD)	$4.940 \cdot 10^6$	29 354	1011392	84 894	74 213	+14.39%
Potsdam1998c (DC)	$4.852 \cdot 10^6$	29 645	994024	78 839	78 890	-0.06%
Potsdam1998c (UT)	$4.813 \cdot 10^6$	39 873	994458	79 560	79 580	-0.03%
Potsdam1998c (BD)	$4.974 \cdot 10^6$	28 951	1017874	84 874	72 702	+16.74%
Potsdam2010 (DC)	$1.025 \cdot 10^6$	9 017	212257	38 255	38 232	+0.06%
Potsdam2010 (UT)	$1.019 \cdot 10^6$	15 257	215906	38 377	38 375	+0.01%
Potsdam2010 (BD)	$1.067 \cdot 10^6$	8 820	220412	41 089	35 647	+15.27%
Potsdam2010+ (DC)	$1.026 \cdot 10^6$	12 661	215258	38 112	38 102	+0.03%
Potsdam2010+ (UT)	$1.020 \cdot 10^6$	27 055	225565	38 207	38 358	-0.39%
Potsdam2010+ (BD)	$1.051 \cdot 10^6$	10 730	218712	41 097	36 740	+11.86%

The only bigger over/underestimations (of nearly 7% for (DC) and 5% for (UT)) are in the China instances. However, the China instances also display the largest prediction improvement in comparison to model (BD), namely, around 40%. But the predicted number of direct travelers can also be significantly improved with model (DC) in comparison to model (BD) for all other instances; the improvement is around 7% for the Potsdam and SiouxFalls instances and around 15% to 20% for the Dutch instances. Model (UT) further improves the number of direct travelers for all instances except for Dutch2 and China1. But the improvement is relatively small compared to the improvement between (BD) and (DC), namely, less than 1.6% for China2 and China3, and less than 1% for all other instances. Moreover, considering the objective values (4th column), the solution for the instances SiouxFalls2, Potsdam1998c, Potsdam2010, and Potsdam2010+ computed with (UT) are worse than the corresponding solutions computed with (DC). Recall that model (UT) is computationally hard to handle for large instances such the Potsdam instances and, hence, computing good solutions (in reasonable time) is difficult.

We therefore conclude that (DC) is currently the best computationally tractable model for the integrated line planning and passenger routing problem. It provides good estimates of the number of direct travelers.

Chapter 9

Potsdam 2010 – Line Optimization in Practice

In this chapter, we report on the project *Stadt+* to optimize the 2010 line plan for Potsdam, a medium sized city in Germany. This project lasted from March 2009 until February 2010 and was organized within the project B15 (Service Design in Public Transport) of the DFG Research Center MATHEON *Mathematics for key technologies* together with the ViP Verkehrsbetriebe Potsdam GmbH. ViP is the public transport company of Potsdam and in charge of 50 buses and 55 trams to operate 496 bus stops, 126 tram stations, and 2 ferry stops in a network with a total length of 389 km. Around 27 million passengers per year use the public transport in Potsdam.

A reorganization of the line plan in Potsdam became necessary when ViP took over six additional bus lines that were formerly operated by Havelbus Verkehrsgesellschaft mbH in the year 2009. A new line plan should be established in the year 2010 with the following requirements

- no higher cost than the cost of the old line plan including the six additional bus lines,
- service improvement, i. e., minimization of travel times and number of transfers,
- compliance with a set of practical requirements, e. g., a minimum cycle time for the tram, minimum frequency requirements for each station, minimal and maximal lengths for lines with respect to travel time and distance.

ViP had planned all former line plans "by Hand" on the basis of experience. The aim of the project Stadt+ was to prove that a new line plan can also be constructed with the support of combinatorial optimization methods and that such a plan can even improve a "hand-made" line plan.

In an iterative process of computation and communication, we fine-tuned our model and set up a data base that was suitable for the application of optimization methods. We included all additional planning requirements in the model. The final optimized solution

reduced the cost by around 4% and the perceived travel time by around 6%. ViP also certified that this line plan was indeed practicable. However, our solution moved some transport from the tram-network to the bus-network which ViP did not want for fear of demand reductions. The Potsdam line plan for 2010 that ViP finally established in practice, therefore, slightly deviates from our optimized solution. However, we could show that mathematical optimization can indeed support line planning. It offers a useful setting to classify solutions according to travel times and costs with bounds on the best possible values for both objectives. Main parts of this chapter are published in [10, 22, 24].

We briefly explain the preparation of the data in Section 9.1. Then we describe the optimization model including the definition of the parameters and the implementation of additional requirements for the Potsdam line plan in Section 9.2. In Section 9.3, we will compare our final solution with the ViP solution for 2010 using the traffic planning software Visum [83]. We end this chapter with some concluding remarks in Section 9.4.

9.1 Data

ViP uses the traffic planning software Visum of the ptv AG [83]. Visum can be used to analyze, visualize, plan, and simulate public and individual traffic. The network data for the city of Potsdam and the OD data were also given in Visum format, and we decided to exchange the Potsdam data between ZIB and ViP in this format. The input for the optimization tool is the network data and the OD data of Potsdam, the output is again the network data including all computed lines. Hence, it is possible to analyze our solution using Visum. In a first step, we established the interface for input and output which existed in a rudimentary state but had not been tested and needed improvements.

At the beginning of the project Stadt+, it turned out that the given network data needed a major reorganization. This was finished in August 2009. More precisely, the given network data represented the status quo of the year 2007. Some stations and connections were missing, some other data were imprecise or not reasonable. The modification of the data required, of course, knowledge of the city and the current public transportation network. It was, hence, done by ViP and included the following aspects:

- combining different OD-nodes and reconnecting them to the network,
- refining the network and defining additional connections for certain transportation modes and/or pedestrians,
- updating of turning relations,
- updating the locations of stations and connections.

When one investiagtes a real world problem the first time and makes computations that have to "survive" in practice, the identification of additional requirements that an optimized solution has to fulfill, generally, takes some time. It needs some practice for the "optimizer" to translate all practical requirements into a form that fits with the optimization tool as well as for the "practitioner" to communicate all important requirements,

especially if these are seen as natural. For some of the requirements the data also has to be enriched, e. g., to define a *minimum frequency requirement* for some stations/stops, i. e., how often the station/stop should be served by a line. Indeed, adjusting the data and/or the optimization tool was an iterative process which was not finished until our final optimization and evaluation at the end of the project. Fortunately, the Visum data interface offers a good environment to include new "attributes" for stops/stations and connections. In close cooperation with ViP we extended the data by some definitions that are needed for the computation to fulfill the requirements on a line plan for Potsdam, which are,

- defining termini, i. e., stations as starting (or end) stations for lines,
- defining a minimum frequency requirement for some stations,
- blocking some stations for operation with new lines,
- defining important transfer stations.

The detailed meaning of these definitions is explained in the next section.

9.2 Model Specification

The goal of the project Stadt+ was to find a line plan that minimizes the travel time and the number of transfers at a cost being not higher than that of the old plan including the six additional bus lines. ViP emphasized the importance of a minimal number of transfers. We decided to use the change and go approach, see Section 6.4, for our main computations since at this time we had not invented the direct connection model of Section 6.5. For the computation of a cost minimal line plan and other lower bounds (Pareto curve) we also used the basic dynamic model (BD), see Section 6.3. It turned out that the number of all relevant lines for Potsdam was below 3500, and, therefore, the change and go approach was a computational nightmare but possible. The computation of just one solution took several days. Indeed, the project Stadt+ motivated the investigation of our new model, the direct connection model of Section 6.5.

In the following, we describe the calibration of the optimization tool including the generation of new lines, the inclusion of all practical requirements in the optimization model as well as the calibration of parameters such as cost, capacity, and weighing parameter for the objective function.

Generating new Lines. New lines should satisfy the following requirements.

- Termini, i. e., start and end of a line are predefined stations/stops.
- A line has to fulfill all turning restrictions.
- A line is not allowed to serve blocked stations/stops.
- The travel time for each direction is limited to 45 minutes.
- Railroad crossings are prohibited.
- A line has to serve at least 5 stations.

- A line has to contain at least one of the important transfer nodes defined in the data (see Section 9.1).
- The route of a bus line is only allowed to be at most 45% parallel to the route of a tram line.
- Some possible turning restrictions are not applied if this would omit a serving of a nearby important transfer station.
- The maximal number of stations/stops of a line is 80.
- The length of a line is at most four times the length of a shortest path between the end stations.

The last two requirements have rather technical reasons to reduce the computation time for generating all lines satisfying the other above defined requirements. We made a preprocessing afterwards and removed a line, if it serves exactly the same OD nodes as another line. In this way, the number of lines was reduced by around 60% and we ended up with a little less then 3500 valid lines.

Practical Requirements. ViP operates around 20 bus and tram lines in Potsdam. The basic cycling time over the whole day is 20 minutes in the city, in the peripheral area it can also be 30 or even 60 minutes. Considering the time from 6 to 9 am (3 hours), we get a frequency set of $\mathcal{F} = \{3, 6, 9, 18\}$. For some stations, the line plan has to fulfill a minimum frequency requirement of 3, 6, 9, or 18, i. e., the station has to be served by a line every 60, 30, 20, or 10 minutes. We denote by $F_v \in \{3, 6, 9, 18\}$ the minimum frequency requirement for station v. Here, an operation of every 30 and every 10 minutes can also be done by two lines with a cycling time of 60 minutes, and by two lines with a cycling time of 20 minutes, respectively. The other two values for the minimum frequency requirement have to be fulfilled by at least one line with exactly the needed frequency. In formulas, we get the following constraints for our model:

$$\sum_{\ell: v \in \ell} \sum_{f \geq F_v} x_{\ell,f} \geq 1 \qquad \forall\, v \in V_1 \quad (F_v \in \{3, 9\}) \tag{9.1}$$

$$\sum_{\ell: v \in \ell} (x_{\ell,(\frac{F_v}{2})} + \sum_{f \geq F_v} 2x_{\ell,f}) \geq 2 \qquad \forall\, v \in V_2 \quad (F_v \in \{6, 18\}). \tag{9.2}$$

The set V_1 denotes the stations/stops with a minimum frequency requirement of 3 or 9, (i. e., the service has to be every 60 or 20 minutes) and the set V_2 denotes the stations/stops with a minimum frequency requirement of 6 and 18 minutes (i. e., the service has to be every 30 or 10 minutes).

Tram lines are more important for ViP than bus lines. Reasons for this are that tram lines are of higher prestige, offer usually more comfort, and are more favored by the passengers compared to bus lines. Hence, we included a rule that the minimum frequency requirements for all stations served by bus and tram lines have to be satisfied by tram lines only, i. e., if v is a station that is served by bus and tram lines, we consider only tram lines in the inequalities (9.1) and (9.2). Additional requirements for the tram lines are that the minimum cycling time is always 20 minutes and that parallel traffic of bus lines should be avoided. The first requirement can be achieved by reducing the set of

possible frequencies for tram lines, the latter is considered during the generation of new lines.

Costs and Capacities. ViP calculates with costs per kilometer to estimate the expected cost of vehicle and duty operation. This is exactly the same approach as in our model. The calculated costs of ViP of August 2009 are 2.34 € per kilometer for tram lines and 1.96 € per kilometer for bus lines. The capacity of a bus was fixed to 125 passengers and the capacity of a tram to 170 passengers. In our computations we considered also lines of other transport companies if they are operated in the city of Potsdam since passengers transfer from and to those lines; this includes, e. g., the local railway traffic and the S-Bahn of Berlin. The frequencies of these lines were fixed to the (at that time) current frequency and we assumed a capacity of 600 passengers for local railway traffic and 536 passengers for the S-Bahn Berlin.

Finally, we included a fixed cost term of 100 € per line. This number expresses no real cost. We only included a fixed cost term to also minimize the number of lines in a line plan. We chose a number which is of similar dimension as the operating cost of a line in the considered time horizon.

Optimization Model. For the optimization model, we refer to the notation given in Section 6.2 and the definition of the change-and-go model given in Section 6.4. The final optimization model is

$$\min \lambda \sum_{\ell \in \mathcal{L}} \sum_{f \in \mathcal{F}} c_{\ell,f}\, x_{\ell,f} + (1-\lambda) \sum_{p \in \mathcal{P}} \tau_p\, y_p$$

$$\sum_{p \in \mathcal{P}_{st}} y_p = d_{st} \qquad \forall\, (s,t) \in D \tag{9.3}$$

$$\sum_{p : a \in p} y_p \leq \sum_{f \in \mathcal{F}} \kappa_{\ell(a),f}\, x_{\ell(a),f} \qquad \forall\, a \in \mathcal{A}_L \tag{9.4}$$

$$\sum_{f \in \mathcal{F}} x_{\ell,f} \leq 1 \qquad \forall\, \ell \in \mathcal{L} \tag{9.5}$$

$$\sum_{\ell : v \in \ell} \sum_{f \geq F_v} x_{\ell,f} \geq 1 \qquad \forall\, v \in V_1 \quad (F_v \in \{3,9\}) \tag{9.1}$$

$$\sum_{\ell : v \in \ell} (x_{\ell,(\frac{F_v}{2})} + \sum_{f \geq F_v} 2x_{\ell,f}) \geq 2 \qquad \forall\, v \in V_2 \quad (F_v \in \{6,18\}) \tag{9.2}$$

$$x_{\ell,f} \in \{0,1\} \qquad \forall\, \ell \in \mathcal{L},\, \forall\, f \in \{3,6,9,18\} \tag{9.6}$$

$$y_p \geq 0 \qquad \forall\, p \in \mathcal{P}. \tag{9.7}$$

The objective is a weighted sum of line operating cost and total passenger travel and transfer time. We add a transfer penalty of 15 minutes on all transfer arcs. The weighing parameter is set to $\lambda = 0.8$, see the discussion below. Inequalities (9.3) ensure that the given demand is satisfied. Inequalities (9.4) require enough capacity for each arc $a \in \mathcal{A}_L$ in the network. Recall that an arc $a \in \mathcal{A}_L$ is associated with exactly one line $\ell(a)$.

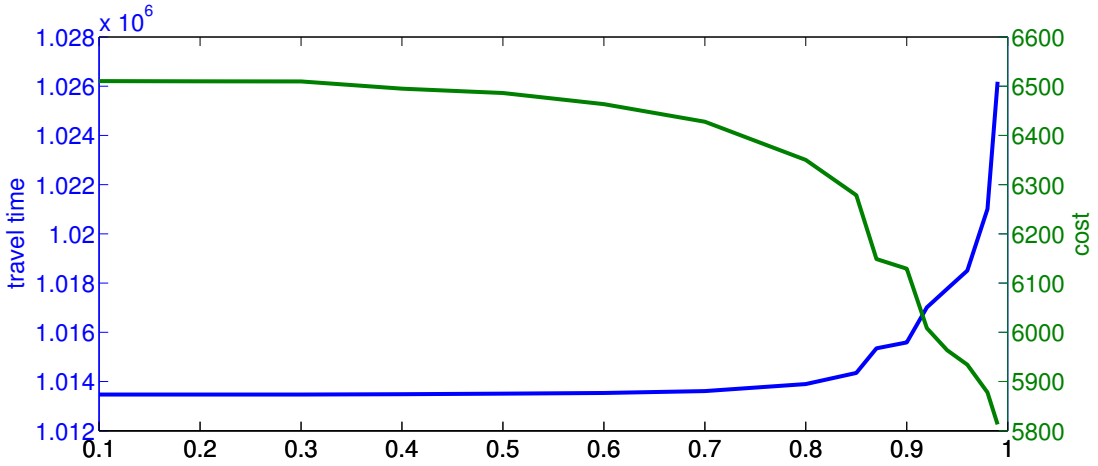


Figure 9.1: Pareto curve. The green (blue) curve represents the cost in € (travel time in minutes) in dependence of λ. A value of $\lambda = 0$ omits costs, a value of $\lambda = 1$ omits travel time.

Constraints (9.5) ensure that each line is operated by at most one frequency. Finally, inequalities (9.1) and (9.2) are the minimum frequency requirements introduced above.

Pareto Curve. To investigate the influence of the parameter λ on the objective value, we computed the optimal objective value of the LP relaxation for 17 different values of λ. Since the size of the cost is much lower than the size of the total travel times, we set half of the samples between $\lambda = 0.8$ and $\lambda = 1$. The results are plotted in Figure 9.1. This figure shows the total traveling time and the total line cost depending on λ. The solution visualized on the left of the x-axis ($\lambda = 0.1$) corresponds to a minimization of mainly the travel time; the solution visualized on the right of the x-axis corresponds mainly to a cost minimization. We did not include the pure extreme points in the graphic, i. e., a pure cost minimization and a pure travel time minimization. In both extreme solutions the part not being considered in the objective explodes because there is no need to bound this value. This would distort the picture. Therefore, we skipped these cases to keep track of the relevant cases.

The two curves show the following: A shift of the weighing parameter from a solution that minimizes the travel time to a solution that minimizes the cost results in a hardly remarkable increase of travel times in the beginning but a more remarkable decrease of costs. At the value of $\lambda = 0.8$ the increase in travel time gets higher. In fact, more detailed computations for $\lambda = 0.8$ yields solutions with costs in the range ViP expected. We, therefore, used $\lambda = 0.8$ for our computations.

9.3 Results

We needed more than 2000 minutes to compute a first solution of the above defined optimization model. We made some modifications and further computations to improve

the solution. We were not able to prove that this solution is optimal but the optimality gap was less than 3%. In recent computations with the new model (DC), compare with Chapter 8, we could solve the instance to optimality within 12 hours. The optimal solution of model (DC) improves the final solution of this project by 0.4%, see also the remark at the end of this section.

In this section we want to analyze our final solution (OPT). To this purpose, we also computed a line plan that minimizes the cost and a line plan that minimizes the travel times. We denote by P10 the line plan that ViP implemented as line plan in Potsdam in the year 2010. All four line plans fulfill the requirements of the preceding section and are visualized in Figure 9.2. In Subsection 9.3.2, we will compare these four line plans to analyze costs, total travel time, and number of transfers with respect to the extreme cases. We did a detailed evaluation of the line plans OPT and P10 by analyzing the optimization results, see also [22, 24], and by using the software Visum, see [10], respectively. Both methods lead to similar results which will be discussed in Subsection 9.3.1. An evaluation with Visum, however, is the accepted practice by public transport companies. We will, therefore, present a detailed comparison of the line plans P10 and OPT using Visum in Subsection 9.3.3.

9.3.1 Optimization vs. Simulation

In the following, we want to compare the passenger routing computed by our optimization method with the passenger routing computed by a Visum simulation. To this purpose, we briefly describe both methods first. Then we globally compare the passenger paths resulting from both methods.

Since the optimization model integrates a passenger routing, the costs and the total travel time as well as the number of transfers can be read off the computed solution, easily. The final routing in the optimization model is a system optimum. This means that a passenger routing is computed that minimizes the total travel and transfer times for all passengers. Hence, it is possible that some passengers have to use detours if the capacities of the shortest paths are utilized by other passenger routes. For the Potsdam instance, however, at most 2 paths are computed for every OD pair. Omitting the capacities of the lines yield similar results concerning travel times and number of transfers. Hence, for the given demand data the passengers rarely have to use long detours.

The software Visum offers different methods to compute a distribution of the passengers based on a line plan and the OD data. In this way, detailed travel path patterns can be identified which yield statistics on the utilization of the lines, on the travel times, and on the number of transfers. We used a passenger distribution that is qualified for urban networks and for line plans not containing an exact timetable and included a transfer penalty of 15 minutes equal to the penalization in the optimization model.

Visum computes nearly all possible passenger paths for each OD pair which are sometimes more than 100 possibilities. These paths differ mainly in start and end stations

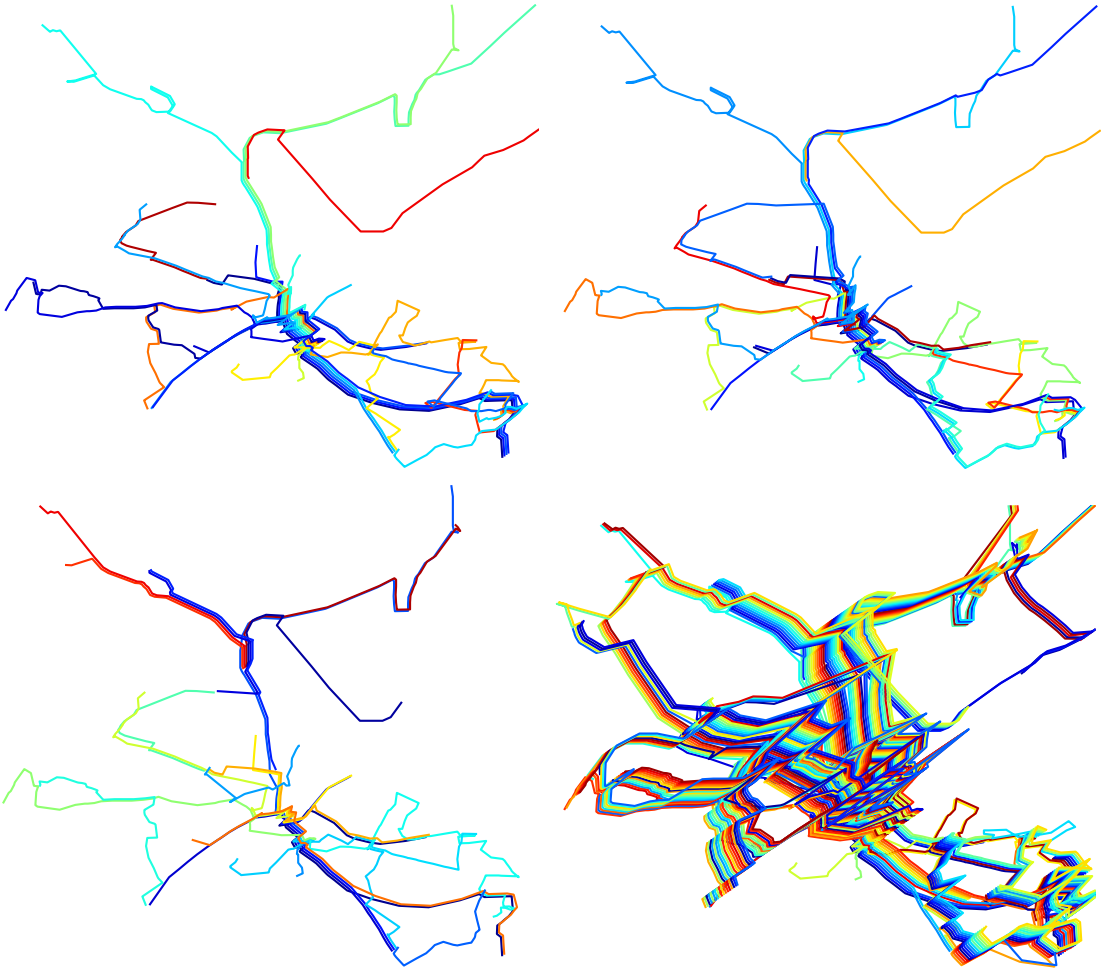


Figure 9.2: Visualization of four different line plans for Potsdam: *Upper left:* P10, *Upper right:* OPT, *Lower left:* cost minimal line plan, *Lower right:* travel time minimal line plan.

(e. g., if the OD node has more than one neighboring station), transfer stations, as well as a different choice of lines that operate parallel routes. Considering the line plan OPT, our optimization yields 4 457 paths (for 4 443 OD pairs); 3 984 (89%) of them are also computed by Visum with a traveler volume of 32 453 passengers, out of 47 743 in total, which is 68%. This shows that the passenger distribution in the optimization model and in a Visum simulation is not equal but similar.

In the following we compare the line plans OPT and P10 according to costs and travel times. In a global comparison, in Subsection 9.3.2, i. e., a comparison of the minimum possible travel time and the minimum possible cost, we used the results given by our optimization method. For the detailed comparison of the line plans OPT and P10, in Subsection 9.3.3, we present the results derived with Visum.

	P10	OPT	min. cost	min. time
no. of lines (bus)	16	16	17	120
no. of lines (tram)	7	7	6	18
km (bus)	2392	2497	1644	10565
km (tram)	1440	1207	1054	3156
cost	8057	7717	5688	28091
travel+transfer time	6.2484	6.1927	6.7344	6.0812
travel time	5.1641	5.1550	5.2889	5.1422

Table 9.1: Characteristic numbers (computed by optimization tool) of the line plans P10, OPT, as well as for the cost minimal, and the travel time minimal solutions. The considered time horizon is one day from 6 to 9 am. The time is given in seconds $\cdot 10^7$. Transfer times include a penalization of 15 minutes per transfer.

no. transfers	P10	OPT	min. cost	min. time
0	35876	36355	32492	37425
1	11689	11249	14477	10205
2	178	139	741	113
3	1	1	34	1

Table 9.2: Number (computed by optimization tool) of transfers for the four line plans. The numbers in the first column give the number of passengers that have to transfer 0, 1, 2, or 3 times. Here, the passengers are distributed according to a system optimum, i. e., as long as capacities are not used by other passengers each passenger can travel on the travel time shortest path (including a transfer penalty of 15 minutes).

9.3.2 Cost vs. Travel Time

We first analyze costs and travel times by comparing the line plans OPT and P10 with the line plans that minimize cost and travel time, respectively. Here, we consider the numbers computed by the optimization model, since we have no Visum evaluation for the extreme solutions. However, it will turn out that the "Visum numbers" for P10 and OPT are similar to the optimization numbers (as it was also argued above). The characteristic numbers computed by our optimization method for all four line plans are given in Table 9.1. A visualization of the four line plans is shown in Figure 9.2.

The cost minimal solution is optimal, i. e., 5688 € is a lower bound on the daily cost for operating a line plan between 6 and 9 am that fulfills all requirements. The solution of the minimal travel time is nearly optimal. The difference to an optimal solution is at most 0.4%, i. e., a lower bound on the travel time including transfer penalties for all passengers is $6.06 \cdot 10^7$ seconds.

The solutions P10 and OPT offer a compromise between these extreme variants. In both cases a slight increase of the travel time compared to the minimum travel time yields a notable reduction in costs. Compared to the cost minimal solution, the line plans P10 and OPT reduce mainly the number of transfers. The numbers for OPT are a little better than for P10: The travel time is around 1% below the travel time of P10 whereas

	OPT	P10
average total travel time	36min 3s	36min 39s
average time in vehicle	13min 8s	14min 36s
average transfer waiting time	1min 30s	1min 29s
average start waiting time	13min 23s	12min 32s
average walking time	1min 38s	1min 37s
average perceived travel time	26min	27min 37s
average distance	9,862	9,868
total travel time	28685h 41min 12s	29165h 8min 37s
total time in vehicle	10444h 19min 42s	11616h 48min 51s
total transfer waiting time	1187h 58min 42s	1181h 51min 3s
total start waiting time	10642h 48min 26s	9970h 53min 15s
total walking time	1297h 15min 48s	1286h 23min 45s
total perceived travel time	20691h 40min 3s	21979h 27min 34s
total distance	470828,526	471129,845
total number of transfers	10595	11141
passengers with 0 transfer	37338	36851
passengers with 1 transfer	10088	10503
passengers with 2 transfers	243	306
passengers with more than 2 transfers	7	9

Table 9.3: Statistics for the line plans OPT and P10 generated with Visum.

the cost can be reduced by around 4%. The lower travel time is mainly achieved by a reduction of transfers, see Table 9.2. The lower costs are a result of the reduction of the tram network.

9.3.3 OPT vs. P10

In the following, we will present the results of a detailed evaluation of the line plans OPT and P10 with Visum.

The number of operated kilometers computed with Visum is 2 321 bus-km and 1 440 tram-km for P10 and 2 393 bus-km and 1 207 tram-km for OPT. This yields a cost of 7 918 € for P10 and 7 514 € for OPT which is a cost reduction of around 5% for OPT compared to P10. Comparing these numbers to those computed by optimization, see Table 9.1, the only deviation is in the number of operated bus kilometers. This yields a slightly higher cost in the optimization numbers in both line plans. Hence, comparing different line plans results in nearly equal statements.

Table 9.3 shows all relevant numbers of the passenger distribution computed by Visum. The travel time in the OPT plan is slightly below the travel time in the P10 plan, the improvement is 1.6%. But more remarkable is the reduction of around 10% for

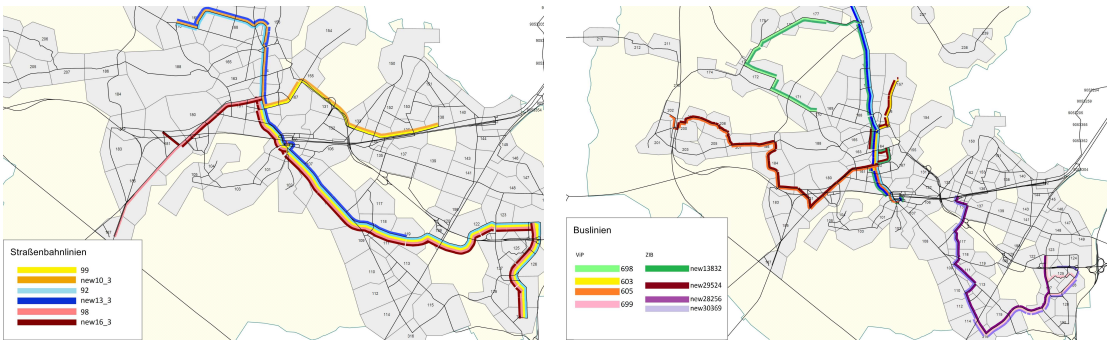


Figure 9.3: Tram lines (*left*) and bus lines (*right*) that are different in the OPT plan and the P10 plan; graphics generated with Visum [83].

the in-vehicle-time, i. e., the time passengers spend driving in busses, trams, etc. The in-vehicle-time improvement is not compensated by more transfers. In fact, the total number of transfers even decreases by around 5%. To be more precise, considering the total of 47 743 passengers, then 487 additional passengers reach their destination without transfer in the OPT plan; the number of passengers that have to transfer once, twice, and more than twice can be reduced by 415, 63, and 2, respectively. Only the walking time increases a little in the OPT solution by around 1%, and the start waiting time increases by around 6%. This increase is probably due to a slight reduction of the cycling time, i. e., some stations are served a little less in the OPT plan than in the P10 plan. The average perceived travel time which is a weighted sum of the different parts of the travel time and an important criterion to quantify the travel paths, amounts to 26 minutes in the OPT plan and to 27 minutes and 37 seconds in the P10 plan, an improvement of around 6%.

Line Routes

Considering routes and frequencies of the lines, then 11 of 16 bus lines and 4 of 7 tram lines are identical in OPT and P10. Figure 9.3 illustrates the line routes of the 5 bus lines and 3 tram lines that are different in OPT and P10. Lines in the OPT plan that are not in P10 are denoted by the prefix "new". In the following, we compare the effects of the different line routes on OD pairs with average or high demand. In our case, these are relations with 10 and more passengers. We consider each line in P10 and its counterpart in OPT if both are different.

Tram line 98. The routes of the P10 line 98 and the OPT line new16 are almost identical. Schloss Charlottenhof is an endpoint of the line new16 while the line 98 operates three more stations and has its endpoint at Bhf Pirschheide. The shortening of the line in the OPT plan results in an additional transfer for a connection between the south-east and the south-west of Potsdam. However, the demand for such connections is relatively small; the cost can be reduced by around 82 €.

Tram line 92. The route of the tramline new13 in the OPT plan is also a shortened version of the tramline 92 in the P10 plan. Line new13 has its endpoint at Bisamkiez while the line 92 is operated beyond the Bisamkiez including Johannes-Kepler-Platz and ends at Marie-Juchacz-Str. Therefore, the line 92 offers a direct connection between Kirschallee and the south-east of Potsdam. Such a direct connection is not present in the OPT plan, however, there are satisfactory alternatives with one transfer and less travel time. One transfer is needed to travel from OD node Konrad-Wolf-Allee to OD node Kirschallee (38 passengers) with a travel time of 39 minutes and 25 seconds. The connection in the P10 plan is transfer free but a travel time of 45 minutes and 50 seconds is needed. The difference in the tram kilometers between 92 and new13 is 5 km per route which corresponds to a cost reduction of 224 € in the OPT plan.

Tram line 99. The examples of lines in the P10 plan and lines in the OPT plan considered so far only differ in the length of their routes. More differences exist between the routes of the line 99 in the P10 plan and the line new10 in the OPT plan. Both lines have one of their endpoints in Fontanestr. but at Platz der Einheit the routes split. The route of new10 proceeds into the north-west with endpoint at Kirschallee while the route of 99 proceeds to the south-east with endpoint at Marie-Juchacz-Str.

The operation of line new10 instead of line 99 reduces the cost by around 239 € but it also offers a direct connection between Babelsberg and Bornstedter Feld resp. Kirschallee. The demand for this connection is much higher than the demand for direct connections covered by line 99. Moreover, there are good alternatives for all relations which are covered by the line 99 in the P10 plan, e. g., a transfer free connection between origin node Kirchsteigfeld Nord and destination node S-Bahnhof Babelsberg (11 passengers) has a travel time of 47 minutes and 4 seconds, a connection with one transfer has a travel time of 44 minutes and 5 seconds. In the OPT plan passengers need 24 minutes and 9 seconds for a connection with one transfer. The omission of line 99 in the OPT plan is therefore compensated by good alternatives. In contrast to that, the line new10 offers direct connections that are not compensated by satisfactory alternatives in the P10 plan. In the OPT plan 83 Passengers from origin node Kirschallee reach their destination node Rathaus Babelsberg without transfers in 27 minutes and 17 seconds, in the P10 plan they have to transfer at least once and have a travel time of at least 38 minutes and 54 seconds.

Bus line 638. The routes of the lines 638 and new1163 differ only slightly, compare with the left of Figure 9.4. The bus line 638 operates seven additional stations in Groß Glienicke. These stations are also operated by a third line that is present in both line plans OPT and P10. The passengers of these stations are, therefore, also transported in the OPT plan, however, the frequency at this stations is reduced. The cost reduces by around 30 €.

Bus line 698. The bus line 698 which is a feeder line for the tram in the P10 plan has the longer line new13832 as its counterpart in the OPT plan. The station Weißer See is an endpoint of the line 698 while the line new13832 serves additional important transfer stations such as Reiterweg/Alleestr. and Platz der Einheit and ends at Hauptbahnhof.

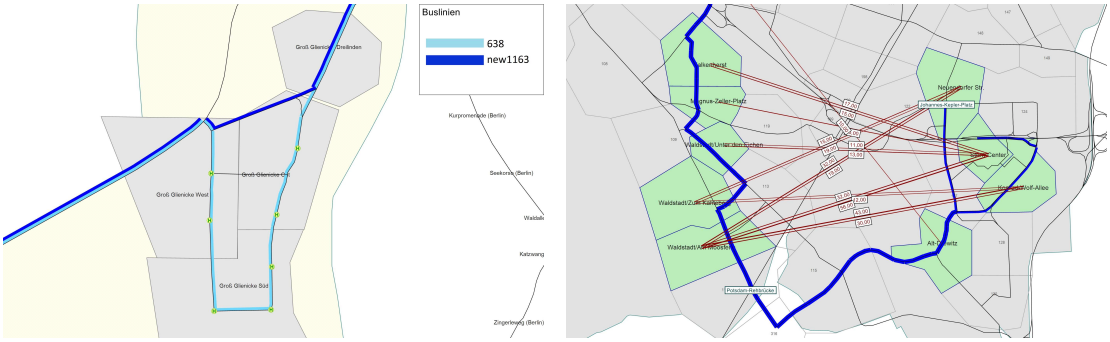


Figure 9.4: *Left:* Comparing the routes of the line 638 in the P10 plan and the line new1163 in the OPT plan. *Right:* OD pairs with a direct (i. e., transfer free) connection provided by the bus lines new28256 and new30369. The numbers on the red lines correspond to the demand of the associated OD pair; graphics generated with Visum [83].

This line offers 40 passengers of the OD nodes Max-Eyth-Allee and Schneiderremise to reach the Hauptbahnhof without transfers. The line is extended from 6 to 13 km whereas the frequency is reduced to an operation of once per hour. Therefore, the cost increases by only 17 € but also 4 stations in the OPT plan are only served once per hour.

Bus lines 603 and 605. The former HVG line 605 connects Golm with the inner-city of Potsdam. The line new29524 also starts in Golm but at Bahnhof Golm/Universität instead of Wissenschaftspark. At Platz der Einheit the route proceeds to the north passing Reiterweg/Alleestr. and ends at Höhenstr. similar as line 603 in the P10 plan. The line new29524, therefore, corresponds to a merged version of the lines 603 and 605 and reduces the cost by around 46 €. However, in the OPT plan a direct connection between OD node Altes Rad and Hauptbahnhof is missing. 23 passengers can use the line 605 for a direct connection to the Hauptbahnhof and 42 passengers reach the OD node Eiche from Hauptbahnhof in 36 minutes and 40 seconds. At least one transfer and 42 minutes and 6 seconds are needed in the OPT plan. On the opposite, 18 passengers between Rathaus and Brandenburger Vorstadt get a transfer free connection. In the P10 plan they have to transfer at least once which is disproportionate to the average travel time of 6 minutes.

Bus line 699. The last two bus lines in the OPT plan which are different to the P10 plan are the lines new28256 and new30369 which we want to consider as variants. They have one endpoint in Babelsberg/Post and the other at Stern-Center in the south-east and therefore have a longer route than the line 699. Between Bhf Rehbrücke and Konrad-Wolf-Allee/Sternstr. the route of the line 699 is identical to new28256 and new30369. The cost increases by around 201 € when operating new28256 and new30369 instead of 699. However, these two lines offer a direct connection for several OD pairs with high demand, see the right of Figure 9.4. The passengers on these OD pairs have to transfer at least once in the P10 plan.

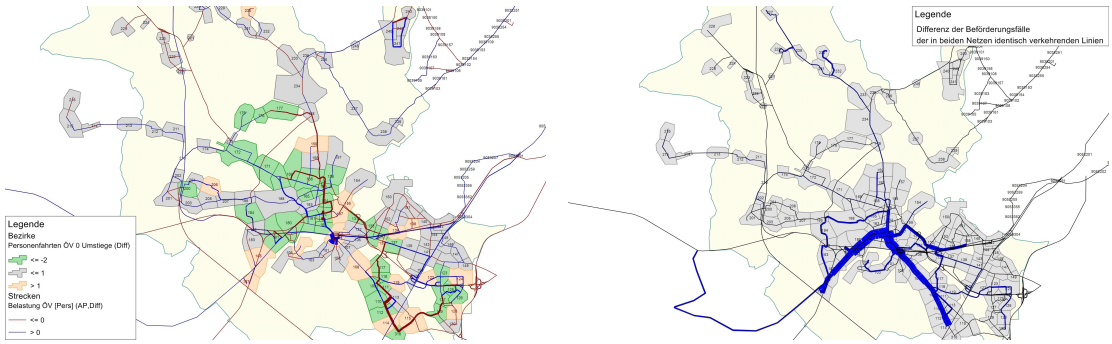


Figure 9.5: *Left:* Comparing transfer free connections according to OD nodes and the difference in the utilization of the capacity of the edges. A green district (corresponding to an OD node) symbolizes that more passengers can reach their destination in the OPT plan without a transfer than in the P10 plan; a red district symbolizes the converse relation. The red edges correspond to more used capacity in the OPT plan, the blue edges correspond to more used capacity in the P10 plan. *Right:* Considering the used capacity of the lines that are equal in both plans. The bigger the blue edges the bigger is the difference in used capacity; graphics generated with Visum [83].

Passenger Routes

The left of Figure 9.5 shows a comparison of the passengers traveling directly, i. e., without transfers, between the P10 plan and the OPT plan: The OPT plan offers more transfer free connections in the green-marked OD nodes while the P10 plan offers more transfer free connections in the red-marked OD nodes. It can be seen that for the majority of OD nodes the OPT plan offers more transfer free connections than the P10 plan; this especially pertains to the OD nodes where the "new"-lines are operated.

The total number of lines operating in Potsdam is 41. This number also includes lines of other operators than ViP such as Deutsche Bahn and Berliner Verkehrsbetriebe. Although only 8 of these 41 lines in Potsdam are different in the OPT plan and the P10 plan, this has a noticable effect on the passenger behavior in terms of different passenger routes and on the capacity utilization in the entire network. The left of Figures 9.5 shows the capacity utilization for all edges in the network. The red edges are used by more passengers in the OPT plan compared to the P10 plan while the blue edges are used by more passengers in the P10 plan compared to the OPT plan. The right of Figures 9.5 shows the difference in the utilized capacity for lines that are equal in the OPT plan and the P10 plan. The bigger the blue edges, the bigger is the difference, i. e., changes of a line route do not only concern the passengers that have used the line before but also other passenger that may be attracted by the new line route. We also detected this effect in the evaluation with the optimization tool. Figure 9.6 shows the utilization of the tram line 96 in the P10 plan (left) and in the OPT plan (right). In both plans the line operates in the same way. But the interaction of all lines in the OPT plan seems to improve the attractivity of line 96 compared to the P10 line plan.

Remark 9.1. *Using the new direct connection model (DC), the line planning problem for Potsdam 2010 including all practical requirements with the above defined parameters can*

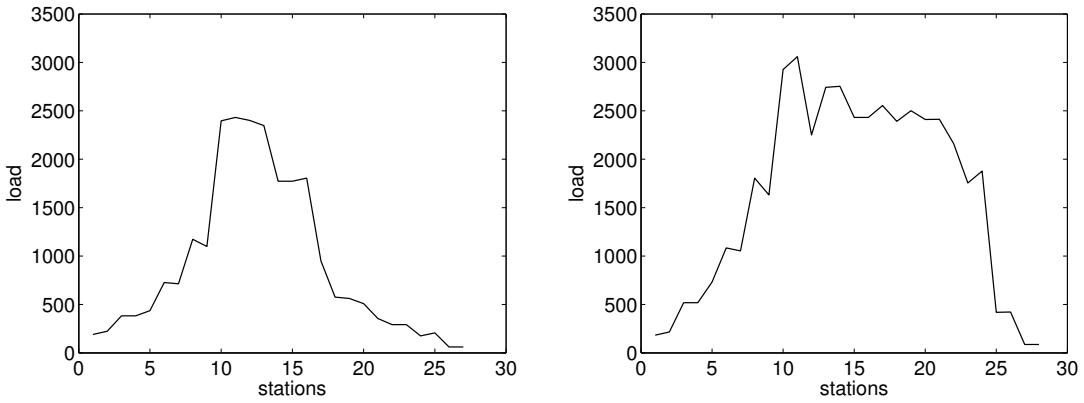


Figure 9.6: Utilization of the tram line 96 in the P10 plan (*left*) and in the OPT plan (*right*). The x-axis represents the stations from Viereckremise to Marie-Juchacz-Str. in the order of service.

be solved to optimality within 12 hours. The costs of the optimal solution DC-OPT are 7 936 €, and, therefore, still lower than for P10 but higher than for OPT. The total travel times of DC-OPT are smaller than for OPT; evaluating both solutions in the change-and-go graph shows that DC-OPT improves OPT in total by around 0.13%. The number of direct travelers increases by around 1.8%. A brief comparison of the DC-OPT line plan with the P10 line plan shows that it only differs in one tramline but replaces 5 bus lines by 12 new (shorter) ones.

9.4 Conclusion

The evaluations with Visum corroborate the results of the line optimization. The passenger distribution in the optimization model yields similar results concerning travel and transfer times as the simulation with Visum. Hence, a better solution, e. g., with respect to travel and transfer times, in the optimization model is also a better solution in practice. The OPT plan fulfills all planning requirements and improves the P10 plan in all important aspects such as travel time, number of transfers, and cost. The savings in costs are achieved by reducing the number of operated kilometers. This can be seen especially well in the tram network and in the reduction of some frequencies. For many OD-relations the OPT plan offers better travel alternatives than the P10 plan; for relations that are affected by the reduction the OPT plan offers satisfactory alternatives or the demand is very small and, therefore, longer travel paths can be accepted.

The potential of optimization and simulation depends of course on the quality of the given data. The OD matrix is a weak point since it represents an aggregation of the data for a considered time horizon, usually a day. Moreover, the demand is also influenced by the offered line plan. In the case of Potsdam, a reduction of the tram service may result in a reduction of passenger demand for public transport. This is, so far, not considered in the line planning models, and this was also one argument that ViP, in the end, established

partly different line routes than those we had computed.

Nevertheless, the advantage of optimization tools is the comprehensive consideration of all given data and the possibility to include a variety of requirements that have to be fulfilled by a solution. A careful disposition, analysis, and maintenance of the data results in the computation of line plans that offer globally good connections for the passengers. New and highly demanded lines are generated while lines with little demand are reduced or canceled. Furthermore, it is possible to compute different variants by moving the focus between cost minimization and quality maximization while all planning requirements are satisfied automatically. In this way, a good compromise between quality of service and cost for operating lines can be identified and justified. The project Stadt+ has shown that mathematical programming and optimization tools make a valuable contribution to plan and to improve public transportation.

Chapter 10

Concluding Remarks for Part II

In the second part of this thesis we investigated the integrated line planning and passenger routing problem. We proposed a new model for this problem, the direct connection model (DC), and discussed two existing models, namely, the change-and-go model (CG) of Schöbel and Scholl and the basic dynamic model (BD) of Borndörfer, Grötschel, and Pfetsch. As objective for all three models we considered a weighted sum of minimizing the line operating costs and the passenger travel times. A comparison of the three models yielded the following picture

$$\begin{aligned} v_{\mathrm{IP}}(\mathrm{CG}) &\geq v_{\mathrm{IP}}(\mathrm{DC}) \geq v_{\mathrm{IP}}(\mathrm{BD}), \\ v_{\mathrm{LP}}(\mathrm{CG}) &\geq v_{\mathrm{LP}}(\mathrm{DC}) \geq v_{\mathrm{LP}}(\mathrm{BD}), \end{aligned}$$

where $v_R(M)$ is the optimal objective value of relaxation R of an integer programming model M. Hence, the theoretically best among the three models for line planning with integrated passenger routing is the change-and-go model (CG). However, this model is also of largest size. It is defined on the change-and-go graph whose number of nodes (arcs) is approximately the product of the number of lines and nodes (arcs) of the underlying graph for the direct connection model and the basic dynamic model, respectively. Table 10.1 lists the size of the change-and-go graph compared to the underlying graph of model (BD) and (DC). It also lists the number of capacity constraints and the running time of a pricing round for the passenger path variables since these are the numbers that differ in all three models. Note that the number of capacity constraints for model (DC) includes all direct connection constraints which can be seen as capacity constraints for direct travelers. In contrast to models (BD) and (DC), the capacity constraints and the pricing problem for the passenger path variables of model (CG) strongly depend on the size of the line pool. Indeed, our computations for the project Stadt+ to compute the line plan for Potsdam showed that model (CG) is a computational nightmare. We needed several days to compute a good solution. Using the new direct connection model (DC) the same setting can be solved to optimality within 12 hours. Evaluating both solutions in the change-and-go graph shows that the DC-solution improves the best CG-solution in total by around 0.13%. This shows that model (DC) is computationally tractable for

Table 10.1: Comparing the sizes and the computational results of models (CG), (DC), and (BD); [1]geometric mean on 14 instances after 10 hours of computation; [6]root LP for 8 of 14 instances could not be solved within 10 hours; [3]difference of direct travelers predicted by model and exact direct travelers in system optimum.

	(CG)	(DC)	(BD)
graph	$G_{CG} = (\mathcal{V}, \mathcal{A})$ $\lvert\mathcal{V}\rvert \in \mathcal{O}(\lvert\mathcal{L}\rvert\lvert V\rvert)$ $\lvert\mathcal{A}\rvert \in \mathcal{O}(\lvert\mathcal{L}\rvert\lvert A\rvert)$	$G = (V, A)$	$G = (V, A)$
capacity constr.	$\lvert\mathcal{A}\rvert \in \mathcal{O}(\lvert\mathcal{L}\rvert\lvert A\rvert)$	$\lvert A\rvert + \mathcal{O}(\lvert A\rvert\lvert D\rvert)$	$\lvert A\rvert$
pricing pass. path var.	$\mathcal{O}(\lvert V_O\rvert(\lvert\mathcal{V}\rvert \log \lvert\mathcal{V}\rvert + \lvert\mathcal{A}\rvert))$ shortest path tree $\forall v \in V_O$	$\mathcal{O}(\lvert D\rvert(\lvert V\rvert \log \lvert V\rvert + \lvert A\rvert))$ weights for st-rdcpaths depend on $(s,t) \in D$	$\mathcal{O}(\lvert V_O\rvert(\lvert V\rvert \log \lvert V\rvert + \lvert A\rvert))$ shortest path tree $\forall v \in V_O$
optimality gap[1]	inf [2]	0.041%	0.039%
Δ direct traveler[1,3]	0.0%	0.39%	19.26%

real world instances and that it yields good solutions concerning passenger travel times and line operating costs.

As far as we know, public transport line planning in practice is currently not supported by mathematical optimization tools. In this thesis we have shown that this is possible. The direct connection model is a suitable tool to support the solution of a first strategic planning problem in public transport by mathematical optimization. This improves the entire planning process of a public transport system from line planning to the subsequent operational planning steps. Line optimization has a leverage effect, since a line plan provides the basis for vehicle and duty scheduling. As optimization tools for vehicle and duty scheduling are already integrated in planning software for public transport, line optimization tools can extend this tool chain from operational planning to strategic planning.

Appendix A

MATHEON *Adventskalender* – Exercises

The *Digitaler Adventskalender* [43] of the DFG research center MATHEON provides mathematical riddles or exercises. Every day, from December 1 st to December 24 th., a door on the calender webpage opens at 18:00 to reveal the exercise of the day which has to be solved correctly and if possible until midnight to have the greatest chance of winning a prize. The idea is to popularize mathematics and to give the general public an understanding of the research done within the MATHEON projects, and indeed the large number of participants, more than 12 000 in the year 2012, made the calender a big success. The exercises were conceived by the MATHEON researchers; I also contributed riddles for the years 2008 to 2012. They are all related to the Steiner connectivity problem and/or to the line planning problem. The riddles of the years 2011 and 2012 were voted by the participants as the "ultimative Kalenderaufgabe" 2011 and 2012, respectively. In the following I list these exercises, each followed by the solution on a subsequent page.

Figure A.1: Digitaler Adventskalender of MATHEON [43].

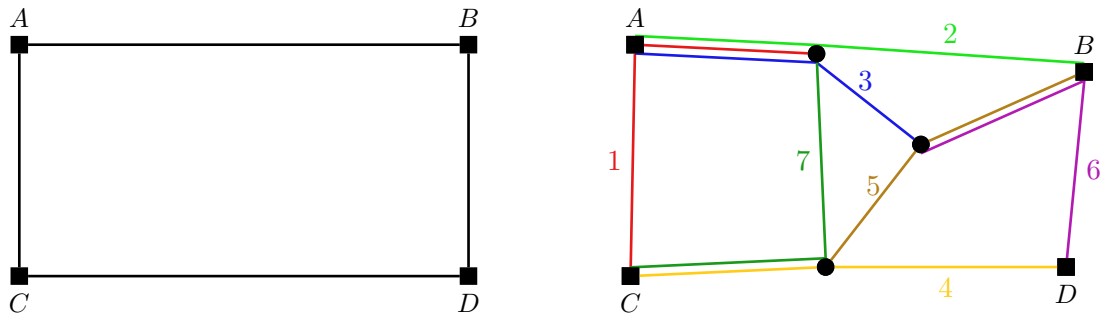


Figure A.2: *Links:* Geschenkedepots, eine Strecke zwischen zwei Depots gibt an, dass dazwischen Geschenke ausgetauscht werden müssen. *Rechts:* Liniennetz der Turboschlittenfirma Schneeflocke.

A.1 Die Turboschlitten Rettung (2008)

Der Weihnachtsmann rauft sich die Haare. Das aber auch jedes Jahr immer irgendetwas schief gehen muss! Diesmal sind aus irgendeinem Grund die Geschenke in den Depots durcheinander gekommen. Es gibt vier wichtige Depots, von denen die Geschenke am Weihnachtsabend in die ganze Welt verteilt werden. Die Verteilung der Geschenke ist bereits bis ins kleinste Detail geplant. Es gibt einen straffen Zeitplan, der eingehalten werden muss. Und erst jetzt, wenige Tage vor Weihnachten, fällt der pragmatischen Weihnachtswichtelin Linda auf, das jemand geträumt haben muss und die Geschenke zum Teil in falsche Depots geliefert wurden. Seufzend schaut der Weihnachtsmann das immermüde Rentier Kalle an, das gerade aus Depot B kam. Aber es hilft ja nichts, die Geschenke müssen an ihre eigentlichen Depots gebracht werden. Der Weihnachtsmann beruft eine außerordentliche Versammlung mit allen weihnachtlichen Helfern ein und erklärte das Problem:

„Ich habe im ersten Bild (Figure A.2 links) skizziert, zwischen welchen Depots Geschenke ausgetauscht werden müssen und zwar sind es genau die Depots, die in dem Bild direkt miteinander verbunden sind, also zwischen A und B, A und C, B und D und zwischen C und D. Um die Geschenke noch schnell vor Weihnachten in die richtigen Depots zu bringen, müssen wir wohl die Dienste der Turboschlittenfirma Schneeflocke in Anspruch nehmen. Sie haben sieben verschiedene Turboschlittenlinien, die verschiedene Stationen anfahren. Ich habe das Turboschlitten-Liniennetz im zweiten Bild (Figure A.2 rechts) skizziert. Wichtig sind für uns die rechteckig markierten Stationen. Da stehen unsere vier Geschenkedepots A, B, C und D. Die Linien verkehren jeweils in Hin- und Rückrichtung. Außerdem ist die Kapazität jeder Linie groß genug alle Geschenke zu transportieren. Wir können jede einzelne der sieben Linien mieten. Allerdings kostet jede Linie, die wir mieten extra. Es ist vielleicht auch nötig, die Geschenke unterwegs von einer Linie auf eine andere umzuladen, wenn es keine direkte Linienverbindung zwischen zwei Depots gibt oder unser Budget nicht für eine umladefreie Verbindung ausreicht. Leider habe ich bei der Firma Schneeflocke bisher nur den Rätselkönig Andi erwischt. Er hat mir natürlich nicht die Kosten (in Euro) direkt verraten, sondern als Rätsel aufgegeben:“

- Die Kosten der Linie 3 sind doppelt so groß wie die von Linie 5.
- Die Kosten der Linie 1 sind doppelt so großwie die von Linie 7.
- Die Kosten der Linie 4 sind um 5 größer als die von Linie 3 und um 5 kleiner als die von Linie 1.
- Die Kosten von Linie 7 sind $\frac{2}{3}$ so groß wie die von Linie 3.

„Die Kosten von Linie 2 und Linie 6 wollte mir Andi nicht verraten," seufzt der Weihnachtsmann. „Er meinte, ich solle mir überlegen, wie viel ich bereit wäre zu zahlen. Wenn ich ihm meine Antwort stichhaltig begründen kann, dann bekommen wir 50% Rabatt auf alles."

Sofort bricht ein allgemeines Gemurmel aus. Viele sind empört, dass so kurz vor Weihnachten der Rätselkönig Andi sein Spiel mit ihnen treibt. Dennoch beginnen viele über mögliche Verbindungen mit den Turboschlittenlinien zu diskutieren. Der Weihnachtsmann ist erstaunt, wie viele (mehr oder weniger kluge) Aussagen zusammen kommen. Er notiert sich einige Punkte, von denen er sicher ist, dass sie bei der Auswahl der Linien von Nutzen sein können bzw. helfen den Rabatt zu erhalten. Auf eine Aussage sollte sich der Weihnachtsmann aber lieber nicht verlassen, wenn er bei seiner Entscheidung zur Linienwahl Kosten und/oder Bequemlichkeit berücksichtigen will. Welche?

Antwortmöglichkeiten:

1. Die pragmatische Weihnachtswichtelin Linda meint: „Wenn wir Linie 4 nicht nehmen, müssen wir Linie 6 wählen."
2. Das schlaue Rentier Karla sagt: „Es kann keine Lösung mit nur zwei Linien geben."
3. Der träge Weihnachtself Olaf brummelt: „Keine der Linien ist so wichtig, dass wir sie unbedingt wählen müssen, um alle Geschenke transportieren zu können."
4. Die großäugige Weihnachtswichtelin Babette meint: „Ich glaube, Linie 1 ist nie in einer kostenminimalen Lösung, egal wie die Kosten von Linie 2 und Linie 6 sind."
5. Benno, der Kopfrechenkönig unter den Rentieren, sagt: „Eine kostenminimale Lösung ohne Linien 2 und 6 kostet 80 Euro."
6. Das immermüde Rentier Kalle murmelt: „Falls wir wirklich mindestens drei Linien brauchen, müssen nicht alle der gewählten Linien zwei verschiedene Depots miteinander verbinden."
7. Die selbstbewusste Weihnachtselfin Gerda sagt: „Falls die Kosten der Linie 2 kleiner gleich 35 sind, gibt es eine kostenminimale Lösung, die Linie 2 enthält."
8. Der extravagante Weihnachtswichtel Bob kontert: „Falls die Kosten der Linie 6 kleiner gleich 30 sind, gibt es eine kostenminimale Lösung, die Linie 6 enthält."
9. Das aufgeweckte Rentier Charlotte meint: „Wenn die Linien 2 und 6 jeweils nur halb so teuer sind wie Linie 7, können wir für weniger als 100 Euro alle Geschenke ohne Umladen transportieren. Und da ist noch nicht mal der Rabatt eingerechnet."
10. Der faule Weihnachtself Paul meint: „Zum Glück müssen wir keine Geschenke zwischen Depot B und C transportieren. Das wäre nie ohne Umladen gegangen."

Richtige Lösung: Antwort 7

Zur Berechnung der Linienkosten ergibt sich folgendes Gleichungssystem (c_ℓ Kosten von Linie ℓ)

$$\begin{aligned} c_3 &= 2 \cdot c_5 \\ c_1 &= 2 \cdot c_7 \\ c_4 &= c_3 + 5 \\ c_4 &= c_1 - 5 \\ c_7 &= \tfrac{2}{3} c_3 \end{aligned}$$

Es ergibt sich

$$\begin{aligned} c_1 &= 40 \\ c_3 &= 30 \\ c_4 &= 35 \\ c_5 &= 15 \\ c_7 &= 20 \end{aligned}$$

Es ist relativ leicht zu sehen, dass die Depots nicht mit zwei oder weniger Linien miteinander verbunden werden können. Daher sind mindestens drei Linien notwendig. Wenn es nur um Kosten geht, wird man nicht mehr als drei Linien auswählen.

Seien x und y die Kosten der Linien 2 und 6. Man kann sich nun alle Lösungen mit drei Linien anschauen und folgendes feststellen:

- Eine kostenminimale Lösung ohne Linien 2 und 6 ist die Wahl der Linien 3, 4, 5 mit Kosten 80 (Antwort 5 ist eine richtig Aussage).
- Eine kostenminimale Lösung mit Linie 2 (aber ohne 6) ist die Wahl von den Linien 2, 4, 5 mit Kosten $50 + y$.
- Eine kostenminimale Lösung mit Linie 6 (aber ohne 2) ist die Wahl von den Linien 3, 6, 7 mit Kosten $50 + x$
- Eine kostenminimale Lösung mit Linie 2 und Linie 6 ist die Wahl der Linien 2, 6, 7 mit Kosten $20 + x + y$.

Es ergibt sich für die 10 Aussagen

1. Aussage richtig: In Depot D fahren nur die Linien 2 und 6. Um Depot D anzufahren, muss wenigstens eine der beiden Linien gewählt werden.
2. Aussage richtig: Alle Linien fahren höchstens zwei Depots an. Alle Linien die zwei Depots anfahren, haben entweder ein Depot gemeinsam oder treffen sich nicht. Daher braucht man wenigstens 3 Linien.
3. Aussage richtig: In jedem Schnitt im Liniennetzgraph, der zwei Depots voneinander trennt, gibt es mindestens zwei Linien, d. h., es gibt immer mindestens zwei verschiedene Möglichkeiten zwei Depots miteinander zu verbinden.
4. Aussage richtig: Ergibt sich aus den obigen Feststellungen.
5. Aussage richtig: Ist die erste Feststellung.
6. Aussage richtig: Siehe z. B. Linien 2, 4 und 5.
7. Aussage falsch: Wenn sowohl Linie 2 als auch Linie 6 über 30 € kosten, ist die Wahl der Linien 3, 4, 5 am kostengünstigsten.

8. Aussage richtig: Kostet Linie 6 weniger oder gleich 30 € und Linie 2 mehr als 30 € ist die Wahl der Linien 3, 6, 7 am kostengünstigsten. Falls auch die Kosten von Linie 6 kleiner gleich 30 sind, ist die Wahl der Linien 2, 6, 7 am besten. In jedem Fall ist in einer kostenminimalen Lösung die Linie 6 enthalten.
9. Aussage richtig: Eine Lösung ohne umladen ist 1, 2, 4, 6 mit Kosten $40 + 10 + 35 + 10 = 95$.
10. Aussage richtig: Es gibt keine Linie, die Depot B und C direkt miteinander verbindet.

Figure A.3: *Links:* Ein Beispiel mit 7 Standorten und 9 Straßen. *Rechts:* Eine Lösung mit zwei Linien, blau und rot.

A.2 Linienplanung (2009)

Nächstes Jahr wird alles anders! Das hat der vorweihnachtliche Verwaltungsrat bereits beschlossen. Um die ständig wachsende Anzahl an Weihnachtswünschen erfüllen zu können, soll es nächstes Jahr mehrere Standorte geben, in denen Geschenke verpackt werden. Die genaue Zahl an Standorten ist noch nicht fest, aber es soll wenigstens vier geben. Die Standorte werden durch ein Straßennetz verbunden werden. Auch wie das aussehen wird, ist noch nicht ganz klar. Da kommt es auf die geografischen Gegebenheiten an. Auf eines kann man sich aber verlassen, alle Standorte sind über das Straßennetz miteinander verbunden, und es gibt keine parallelen Straßen. Ein Beispiel, wie es aussehen könnte, ist in Abbildung A.3, links, dargestellt.

Damit die Verteilung der Geschenke zwischen den Standorten gut funktioniert, sollen auf dem Straßennetz Transportlinien eingerichtet werden. Diese Linien sollen so gewählt werden, dass jeder Standort von jedem anderen durch die Nutzung der Linien erreicht werden kann. Umsteigen geht an jedem Standort, der von den betreffenden Linien bedient wird. Linien können überall definiert werden, und verschiedene Linien können sich eine Straße teilen. Es gibt dabei folgende Bedingungen an die Linien:

1. Eine Linie soll stets genau vier Standorte miteinander verbinden und darf nur auf den gegebenen Straßen fahren.
2. Linien fahren in beide Richtungen und in jeder Richtung an jedem der vier Standort nur einmal vorbei.

Das rechte Bild in Abbildung A.3 zeigt eine Lösung für das Beispiel auf der linken Seite.

Zur Vereinfachung wird angenommen, dass die Kapazitäten der Linien und die Länge der Straßen keine Rolle spielen.

Der Weihnachtsmann hat sich bezogen auf ein gegebenes Straßennetz und gegebene Standorte bereits überlegt, wie man diese Linien definieren kann. Hier ist seine Idee:

1. Zu Beginn wähle eine beliebige Linie, die vier Standorte miteinander verbindet. Die Standorte, die von dieser Linie angefahren werden, werden notiert.
2. Wenn es noch Standorte gibt, die nicht notiert sind, dann füge eine Linie dazu, die mindestens einen notierten Standort und mindestens einen nicht notierten Standort

enthält. Notiere alle neu angefahrenen Standorte. Wiederhole diesen Schritt so lange, wie es noch nicht notierte Standorte gibt.

Der Weihnachtsmann stellt das Problem und seinen Lösungsvorschlag fünf seiner wichtigsten Helfer vor. Hier sind ihre Aussagen:

- Anton: Es gibt Konstellationen von Standorten und Straßennetz für die es keine Lösung gibt.
- Beate: Falls es eine Lösung gibt, dann liefert der Lösungsvorschlag vom Weihnachtsmann eine Lösung.
- Claudia: Gibt es zwischen je zwei Standorten bezogen auf des Straßennetz (ohne Linien) zwei verschiedene Wege, dann bekommen wir mit dem Algorithmus eine Lösung mit minimaler Anzahl an Linien.
- Doreen: Ist das Problem lösbar und gilt für die Anzahl der Standorte n, dass $n-1$ durch drei teilbar ist, dann gibt es eine Lösung, so dass keine zwei Linien, die gleiche Straße benutzen.
- Emil: Wenn es eine Lösung gibt, dann werden mindestens $\lceil \frac{n-1}{3} \rceil$ viele Linien benötigt (n ist die Anzahl der Standorte).

Der Weihnachtsmann denkt kurz nach und ist der Meinung, dass nur drei Aussagen stimmen. Welche sind es?

Antwortmöglichkeiten:

1. Anton, Beate, Claudia
2. Anton, Beate, Doreen
3. Anton, Beate, Emil
4. Anton, Claudia, Doreen
5. Anton, Claudia, Emil
6. Anton, Doreen Emil
7. Beate, Claudia, Doreen
8. Beate, Claudia, Emil
9. Beate, Doreen, Emil
10. Claudia, Doreen, Emil

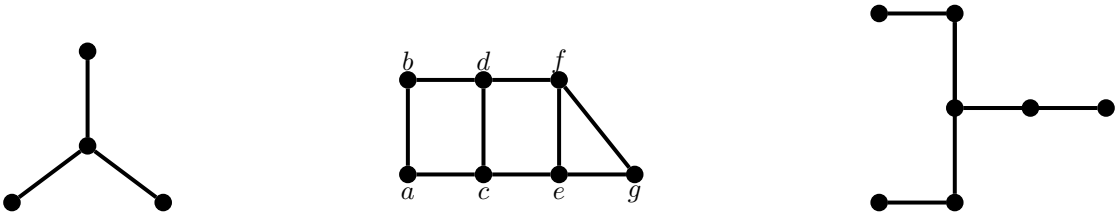


Figure A.4: Beispielnetze.

Richtige Lösung: Antwort 3

- Die Aussage von Anton ist korrekt. Das Beispiel in der linken Abbildung A.4 hat vier Standorte und alle sind über Straßen miteinander verbunden. Aber man kann keine Linien finden, die genau vier Standorte anfahren und jeden in jeder Richtung nur einmal.
- Die Aussage von Beate ist richtig. Wenn es eine Lösung gibt, dann kann man eine Linie, die alle Bedingungen erfüllt finden. Gibt es noch nicht angefahrene Standorte, so muss einer dieser Standorte direkt durch eine Straße von einen der vier (von der ersten Linie angefahrenen) Standorte erreichbar sein. In jedem Fall kann man eine Linie finden, die drei von den bereits angefahrenen Standorten enthält und den neuen Standort.
  Die Behauptung folgt dann per Induktion.
- Die Aussage von Claudia stimmt nicht. Betrachte dazu die mittlere Abbildung A.4. Es gibt zwischen je zwei Standorten zwei verschiedene Wege. Eine Minimallösung enthält zwei Linien, z.B. $\ell_1 = \{a, c, e, g\}$ und $\ell_2 = \{b, d, f, g\}$. Wenn der Algorithmus aber mit Linie $\{f, d, c, e\}$ beginnt, werden noch mindestens zwei Linien benötigt, um die restlichen Standorte zu bedienen.
- Die Aussage von Doreen stimmt nicht. Betrachte dazu die rechte Abbildung A.4. Es sind mindestens drei Linien nötig, um alle Standorte miteinander zu verbinden. Da es aber nur sechs Straßen gibt, müssen sich die Linien Straßen teilen.
- Die Aussage von Emil stimmt. Eine Linie, die genau vier Standorte enthält, fährt über genau drei Straßen. Um n Standorte miteinander zu verbinden, muss man über $n - 1$ Straßen fahren, d. h. $\frac{n-1}{3}$ ist eine untere Schranke an die Anzahl benötigter Linien. Da wir Linien ganz oder gar nicht nehmen, kann dieser Zahl aufgerundet werden.

Damit haben Anton, Beate und Emil recht.

A.3 Der lange Weg nach Hause (2010)

Letztes Jahr war die Weihnachtsfeier wieder sehr unterhaltsam. Susanne hatte über's Jahr viele Witze gesammelt und auf eine ihr unnachahmlich komischen Weise zum Besten gegeben. Kalles Parodie vom Chef, der mal wieder früher gehen musste, war auch nicht von schlechten Eltern, und Claudias Rumkugeln hielten, was sie versprachen, im wahrsten Sinn des Wortes. Nur die Heimfahrt war durch Schneegestöber und vereiste Züge gestört und zog sich bis zum Morgengrauen hin. Zum Glück hatten alle das gleiche Ziel und so zog sich nicht nur der Heimweg sondern auch die Feier bis zum Morgengrauen.

Dieses Jahr haben Susanne, Claudia und Kalle überlegt, ob sie eine Fahrgemeinschaft bilden. Aber dann muss einer auf die leckeren Rumkugeln verzichten. Das will und kann keiner versprechen. Stattdessen sinnieren alle drei über den Nahverkehr und die möglichen Probleme. Hier sind ihre Aussagen:

- Insgesamt gibt es fünf Strecken (direkte Verbindungen zwischen zwei Stationen).
- Erst wenn mindestens zwei Strecken durch Schneeverwehungen nicht mehr passierbar sind, dann kommen wir nicht mehr an unser Ziel.
- Es verkehren mehrere Linien. Eine Linie ist dabei immer eine Menge von Strecken. Für diese Strecken gilt, dass der Bus an jeder Station, an der er vorbeikommt, auch hält. (Es gibt also keine Expresslinien.)
- Schade eigentlich, dass wir immer mindestens einmal umsteigen müssen.
- Es müssen mindestens 3 der 6 Linien, die wir nutzen können, ausfallen, damit wir nicht mehr ans Ziel kommen.
- Falls die Linien 1, 4 und 5 ausfallen, kommen wir nicht an unser Ziel.
- Das gleiche gilt, wenn Linien 2, 3 und 6 ausfallen.
- Ebenso, falls Linien 1, 2 und 3 ausfallen oder 2, 5 und 6.
- Selbst wenn Linie 5 fährt aber nicht die Linien 1,3 und 4, kommen wir nicht an.

Eine 3er Kombination an Linien fehlt noch, die bei Ausfall dazu führt, dass Susanne, Claudia und Kalle nach der Weihnachtsfeier nicht mehr nach Hause kommen. Welche?

Antwortmöglichkeiten:

1. 1, 2, 6
2. 1, 4, 6
3. 1, 2, 5
4. 2, 3, 4
5. 2, 4, 5
6. 2, 4, 6
7. 3, 4, 5
8. 3, 4, 6
9. 3, 5, 6
10. 4, 5, 6

Hinweis: Weil im Forum die Fragen aufkamen:

- Linien können in beide Richtungen fahren.

- Gründe fuer das Nichtnachhausekommen sind:
  - Streckenausfälle (mindestens 2)
  - Linienausfälle (mindestens 3)

  Beide sind voneinander unabhängig. Der Fahrer einer Linie kann also trotz freier Strecken im Spekulatiusrausch liegen. Und wenn eine Strecke verschneit ist, dann scheitert das Nachhausekommen nicht an den Linien, sondern am Schnee. Schnee legt also keine Linien lahm sondern Strecken. Für die Bestimmung der fehlenden Linienkombination kann davon ausgegangen werden, dass der Schnee keine Rolle spielt.

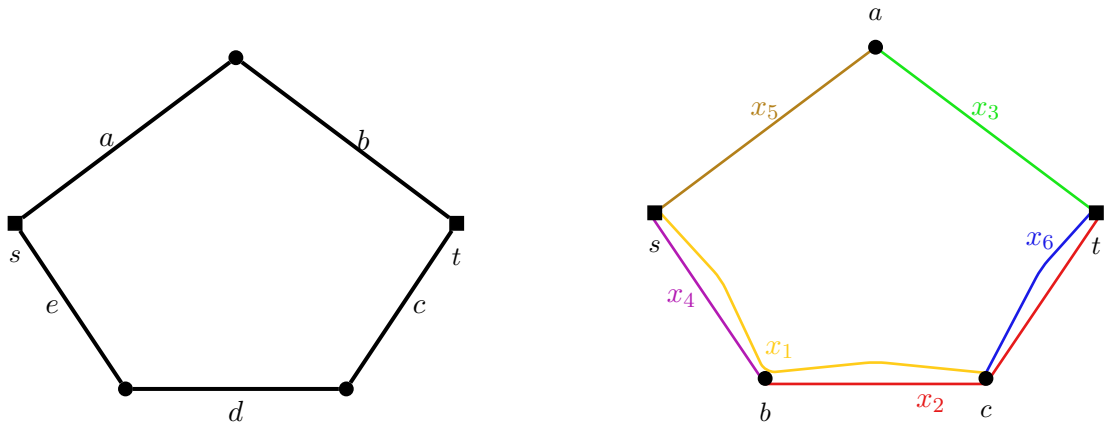


Figure A.5: *Links:* Nahverkehrsnetz mit 5 Strecken (Kanten), das alle Voraussetzungen der Aufgabe erfüllt.

Richtige Lösung: Antwort 3

Das Nahverkehrsnetz hat 5 Strecken/Kanten. Da mindestens zwei Kanten ausfallen müssen, um vom Start s zum Ziel t zu kommen, ist das Netz zweifach-zusammenhängend, d. h., es gibt mindestens zwei kantendisjunkte Wege von s nach t. Weiterhin muss es mindestens sechs verschiedene minimale Schnitte im Netz geben, da es sechs Kombinationen von je drei Linien gibt, die durch Ausfall keine Verbindung mehr von s nach t zulassen. Als einzige Möglichkeit gibt es daher nur das Netz in Abbildung A.5, links.

Es gibt 6 Linien, d. h., auf mindestens einer Strecke verkehren 2 Linien. Weiterhin ist Voraussetzung, dass mindestens 3 der 6 Linien ausfallen müssen, so dass es keine Verbindung mehr von s nach t gibt. Im Netz müssen daher entweder auf den oberen zwei Kanten je zwei Linien verkehren oder auf den unteren drei Kanten. Schaut man sich die in der Aufgabe gegebenen „Linienschnitte“ an, kommt Linie 4 immer in Verbindung mit 1 und Linie 6 immer in Verbindung mit Linie 2 vor. Durch ein wenig probieren kann man feststellen, dass sich Linie 1 und 4 eine Strecke teilen müssen sowie 2 und 6. Des weiteren kommen Linie 1 und 2 ebenfalls zusammen in einem Schnitt vor. Das heißt, Linie 1 und 2 müssen jeweils zwei Kanten enthalten. Durch ein wenig probieren, erhält man (bis auf Symmetrien) das Liniennetz in Abbildung A.5, rechts. Der einzige mögliche Linienschnitt mit drei Linien, der in der Aufgabe nicht erwähnt wird, ist der mit Linie 1, 2 und 5.

Ausgehend vom Netz aus Abbildung A.5, links, kann man auch ein Gleichungssystem aufstellen, um die Belegung der Kanten durch Linien herauszufinden. Wir betrachten dazu folgende Mengen

- $A = \{a, b, c, d, e\}$ Kanten im Netz
- $L = \{1, 2, 3, 4, 5, 6\}$ Menge der Linien
- $I = \{\{a, e\}, \{a, d\}, \{a, c\}, \{b, e\}, \{b, d\}, \{b, c\}\}$ Kantenkombination, die zu minimalen Schnitten führen
- $J = \{\{1, 4, 5\}, \{1, 3, 4\}, \{2, 3, 6\}, \{1, 2, 3\}, \{2, 5, 6\}\} \cup \{j_0\}$ Linienschnitte aus der Aufgabe zusammen mit dem gesuchten Schnitt

und zwei Variablentypen.

- x_{ij} Kantenkombination i enthält Linienschnitt j
- $y_{\ell a}$ Linie ℓ fährt auf Kante a

Folgendes Gleichungssystem führt zu verschiedenen (symmetrischen) Belegungen von Linien zu Kanten, alle Lösungen führen aber zum gleichen fehlenden Schnitt

$$\begin{array}{lll}
(i) & \sum_{j\in J} x_{ij} & = 1 \qquad \forall\, i \in I \\
(ii) & \sum_{i\in I} x_{ij} & = 1 \qquad \forall\, j \in J \\
(iii) & x_{ij} - \sum_{a\in i} y_{\ell a} & \leq 0 \qquad \forall\, \ell \in L \\
(iv) & \sum_{a\in i} \sum_{\ell\in L} y_{\ell a} & = 3 \qquad \forall\, i \in I \\
(v) & x_{ij} & \in \{0,1\} \quad \forall\, i \in I, \forall\, j \in J \\
(vi) & y_{\ell a} & \in \{0,1\} \quad \forall\, \ell \in L, \forall\, a \in A
\end{array}$$

Erklärung zu den einzelnen Zeilen des Gleichungssystems

- (i) Jeder Kantenkombination wird genau ein Linienschnitt zugeordnet.
- (ii) Jedem Linienschnitt wird genau eine Kantenkombination zugeordnet.
- (iii) Wenn Linienschnitt i der Kantenkombination j zugeordnet wird, muss jede Linie aus dem Linienschnitt auf wenigstens einer Kante der Kantenkombination fahren
- (iv) Für jede Kantenkombination gilt, dass genau drei Linien auf Kanten dieser Kombination fahren (da jeder Linienschnitt nur drei Linien hat)
- (v) Entschneidungsvariable, ob Kantenkombination i Linienschnitt j enthält ($x_{ij} = 1$) oder nicht ($x_{ij} = 0$)
- (vi) Entschneidungsvariable, ob Linie ℓ auf Kante a fährt

Man erhält als Lösung welche Kanten den gesuchten Linienschnitt enthalten und wie die Linien auf den Kanten fahren. Daraus kann man leicht ablesen, dass der fehlende Linienschnitt die Linien 1, 2 und 5 enthält.

A.4 Gesprächige Wichtel (2011)

Es sollte eine tolle Überraschung werden. In den letzten Jahren hat es die viele Arbeit immer nicht zugelassen. Aber dieses Jahr will der Osterhase seinen guten Freund, den Weihnachtsmann, am Weihnachtsabend besuchen und hat sich schon sehr auf das überraschte Gesicht von ihm gefreut. Doch irgendwie haben drei Wichtel, Franz, Rita und Ole, von dem geplanten Besuch erfahren. Leider können sie nichts für sich behalten. Sobald sich Wichtel treffen, reden sie über alles, was sie gehört haben. Selbst, wenn der Weihnachtsmann dabei ist, werden alle Informationen ausgetauscht. Bei folgenden Aktivitäten treffen Wichtel bzw. Wichtel und Weihnachtsmann aufeinander.

- Franz und Hella treffen sich bei einer heißen Schokolade.
- Ole, Hella und Emil gehen schwimmen.
- Emil und Theo treffen sich in der Sauna.
- Theo und Erika diskutieren gerne bei einem Glas Wein.
- Ina, Harry und Nelly spielen zusammen Skat.
- Rita und Harry treffen sich beim Tischtennis.
- Erika, Nelly, Alfons und Caro spielen zusammen Poker.
- Theo Hanna und der Weihnachtsmann gehen zusammen rodeln.
- Ina, Caro und der Weihnachtsmann spielen zusammen Basketball.
- Alfons und der Weihnachtsmann fahren zusammen Ski.

Welche Aktivitäten müssten bis zum Weihnachtsabend ausfallen, damit der Weihnachtsmann sicher nichts vom geplanten Besuch erfährt?

Antwortmöglichkeiten:

1. Schokolade trinken und Schwimmen
2. Wein trinken und Tischtennis
3. Poker und Rodeln
4. Skat und Sauna
5. Skat und Poker
6. Basketball und Ski
7. Tischtennis und Rodeln
8. Sauna und Basketball
9. Ski und Schwimmen
10. Wein trinken und Schokolade trinken

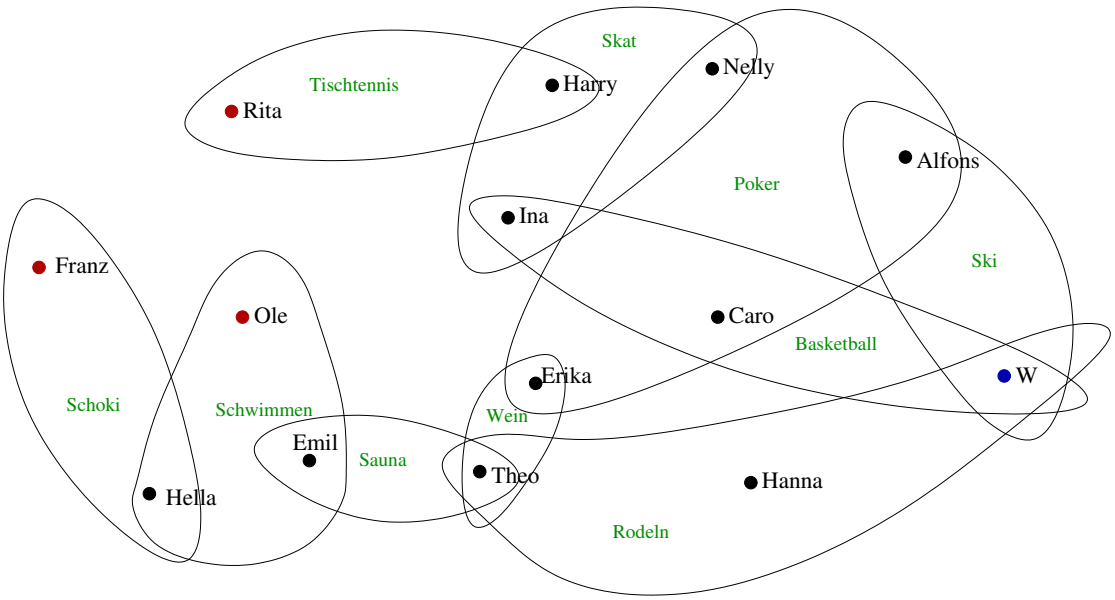


Figure A.6: Konstellationen in denen Wichtel bzw. Wichtel und Weihnachtsmann aufeinander treffen.

Richtige Lösung: Antwort 4

Eine Darstellung der Wichtel und ihre Teilnahme an den verschiedenen Aktivitäten ist in Figure A.6 dargestellt

Der Weihnachtsmann hört von dem Besuch, wenn es eine Kette von sich schneidenden Aktivitäten/Flächen gibt, die den Weihnachtsmann (blauer Punkt) und mindestens einen der drei Wichtel Franz, Rita, Ole (rote Punkte) enthält.

Man sieht sofort, dass bei Ausfallen von Skat und Sauna (Antwort 4) keine solche Kette mehr existiert. Nun kann man sich noch vergewissern, dass bei allen anderen Antwortmöglichkeit immer wenigstens eine Kette bestehen bleibt.

A.5 Wer war es? (2012)

Der Weihnachtsmann ist verärgert. Eine leckere Sahnetorte sollte auf ihn im Teehaus warten. Er hatte sich schon den ganzen Tag darauf gefreut, sie in seiner Pause am Nachmittag zu essen. Aber, was für eine Überraschung, neun Weihnachtswichtel hatten sich bereits im Teehaus versammelt, als er ankam. Und von einer Sahnetorte war weit und breit nichts zu sehen. Die Weihnachtswichtel, sonst immer gesprächig, schauten verdruckst auf den Boden. Niemand wollte was wissen oder gesehen haben. Der Weihnachtsmann schaute sie lange an und überlegte. Es blieb ja nicht viel geheim vor ihm. Der Besuch vom Osterhasen letztes Jahr war auch keine Überraschung. Und dieses Jahr war eine Theateraufführung geplant. Sollte wohl auch eine Überraschung werden, aber irgendwie hat der Weihnachtsmann schon einiges in Erfahrung bringen können. Er wusste, dass alle Wichtel den Tag über mit Vorbereitungen für das Theaterstück beschäftigt waren. Die Vorbereitungen beinhalten Einstudieren, Vorsprechen, Nase pudern, Dekorieren des Bühnenbildes und Anprobe. Jede Aktivität befindet sich in einem anderen Haus. Zwischen je zwei Häusern gibt es eine Verbindung, die mit dem Schlitten bewältigt werden kann. Entweder geht es auf dieser Verbindung bergauf oder bergab. Wenn es in eine Richtung bergauf geht, geht es natürlich in die Gegenrichtung bergab. An diesem Tag hat jeder der neun Wichtel drei der fünf Aktivitäten besucht. Die Wege dazwischen haben alle mit dem Schlitten zurückgelegt. Nach der letzten Aktivität sind alle mit ihrem Schlitten ins Teehaus gefahren. Auch diese Verbindungen verlaufen entweder bergauf oder bergab, je nachdem, von wo man kommt. Folgende weitere Informationen hat der Weihnachtsmann:

- Alle neun Wichtel haben die gleiche Anzahl an bergauf-Verbindungen gehabt. (Die Fahrt zum Teehaus ist dabei mitgezählt.)
- Idrin und Tol sind die Einzigen, die von ihrer letzten Aktivität bergab zum Teehaus rodeln konnten.
- Die erste Station von Idrin war die Anprobe, danach fuhr er hinauf zum Dekorieren.
- Anprobe und Dekorieren waren die 2. und 3. Station für Sharna und Rorrina.
- Zarna begann mit Dekorieren, fuhr danach zur Anprobe und als letztes zum Einstudieren.
- Miira begann mit der Anprobe. Ihre 2. Station war das Nase pudern.
- Tinder begann wie Sharna mit dem Einstudieren. Beide hatten die gleiche 2. Station.
- Erden begann wie Rorrina mit Nase pudern. Die 2. Station von Erden war Dekorieren, zu der er hinunterfuhr.
- Der einzige, der mit dem Vorsprechen begann, war Nor.
- Vom Vorsprechen geht es in alle Richtungen bergab.

Folgendes kann vorausgesetzt werden: Alle Wichtel haben gleichzeitig ihre erste Aktivität begonnen. Geht es von der ersten Aktivität zur zweiten bergauf, braucht jeder Wichtel 10 Minuten, geht es bergab, braucht jeder Wichtel 5 Minuten. Für die Verbindungen danach gilt: bergab braucht jeder Wichtel 2 Minuten weniger als für die letzte Verbindung

davor, bergauf braucht jeder Wichtel doppelt so lange wie für die letzte Verbindung davor. Alle Wichtel brauchen bei jeder Aktivität im Haus die gleiche Zeit. Wer war unter diesen Voraussetzungen der/die erste, der/die im Teehaus ankam und damit die beste Möglichkeit hatte, unbemerkt die Sahnetorte zu verspeisen?

Antwortmöglichkeiten:

1. Zarna
2. Idrin
3. Miira
4. Tinder
5. Sharna
6. Tol
7. Erden
8. Rorrina
9. Nor
10. Es kann keine eindeutige Aussage gemacht werden.

Richtige Lösung: Antwort 3

Die Informationen, die der Weihnachtsmann kennt, ergeben folgendes Bild:

	Aktivitäten						
Name	1	2	3	Ende	1. Verbindung	2.Verbindung	3.Verbindung
Zarna	D	A	E	T	bergab	bergauf	bergauf
Idrin	A	D		T	bergauf	bergauf	bergab
Miira	A	N		T	bergauf	bergab	bergauf
Tinder	E	A		T	bergab	bergauf	bergauf
Sharna	E	A	D	T	bergab	bergauf	bergauf
Tol				T	bergauf	bergauf	bergab
Erden	N	D		T	bergab	bergauf	bergauf
Rorrina	N	A	D	T	bergab	bergauf	bergauf
Nor	V			T	bergab	bergauf	bergauf

Die grün markierten Infos können aus den anderen hergeleitet werden, z.B. Sharna und Rorrina fahren als zweites bergauf, da Idrin von A nach D bergauf fährt. Da alle die gleiche Anzahl an Bergauf-Verbindungen haben, fahren Sharna und Rorrina als erstes bergab. Es folgt, dass Miira zuerst bergauf fährt, da sie die ersten beiden Aktivitäten in umgekehrter Reihenfolge besucht als Rorrina, als zweites also bergab, usw.

Es sind nur die Fahrzeiten auf den Verbindungen relevant. Es ergibt sich aus dem Text:

- bergauf, bergauf, bergab: $10 + 20 + 18 = 48$
- bergauf, bergab, bergauf: $10 + 8 + 16 = 34$
- bergab, bergauf, bergauf: $5 + 10 + 20 = 35$

Da Miira die einzige ist, die bergauf, bergab, bergauf fährt, ist sie als erste im Teehaus.

Bibliography

[1] Tobias Achterberg. *Constraint Integer Programming.* PhD thesis, TU Berlin, 2007. Cited on page 160.

[2] Tobias Achterberg. SCIP: Solving Constraint Integer Programs. *Mathematical Programming Computation*, 1(1):1–41, 2009. Cited on pages 7, 81, 92, 165.

[3] Ravindra K. Ahuja, Thomas L. Magnanti, and James B. Orlin. *Network Flows: Theory, Algorithms, and Applications.* Prentice-Hall, Inc., 1993. Cited on page 67.

[4] Anantaram Balakrishnan, Thomas L. Mangnanti, and Prakash Mirchandani. Network design. In Mauro Dell'Amico, Francesco Maffioli, and Silvano Martello, editors, *Annotated Bibliographies in Combinatorial Optimization*, chapter 18, pages 311–334. Wiley, Chichester, 1997. Cited on pages 1, 135.

[5] Egon Balas and Shu Ming Ng. On the set covering polytope: I. All the facets with coefficients in {0,1,2}. *Mathematical Programming*, 43:57–69, 1989. Cited on page 51.

[6] Gregor Baudis, Clemens Gröpl, Stefan Hougardy, Till Nierhoff, and Hans Jürgen Prömel. Approximating minimum spanning sets in hypergraphs and polymatroids. Technical report, HU Berlin, 2000. Cited on page 18.

[7] Timo Berthold. Primal heuristics for mixed integer programs. Diploma thesis, TU Berlin, 2006. Cited on pages 155, 156.

[8] Ralf Borndörfer. *Aspects of Set Packing, Partitioning, and Covering.* PhD thesis, TU Berlin, 1998. Cited on page 64.

[9] Ralf Borndörfer, Ivan Dovica, Ivo Nowak, and Thomas Schickinger. Robust tail assignment. Technical Report 10-08, ZIB, 2010. Cited on page 4.

[10] Ralf Borndörfer, Isabel Friedow, and Marika Karbstein. Optimierung des Linienplans 2010 in Potsdam. *Der Nahverkehr*, 30(4):34–39, 2012. Cited on pages 5, 152, 174, 179.

[11] Ralf Borndörfer, Martin Grötschel, and Ulrich Jaeger. Planungsprobleme im öffentlichen Verkehr. In Martin Grötschel, Klaus Lucas, and Volker Mehrmann, editors, *PRODUKTIONSFAKTOR MATHEMATIK – Wie Mathematik Technik und Wirtschaft bewegt*, acatech DISKUTIERT, pages 127–153. acatech – Deutsche Akademie der Technikwissenschaften und Springer, 2008. Cited on page 4.

[12] Ralf Borndörfer, Martin Grötschel, and Ulrich Jaeger. Planning problems in public transit. Technical Report 09-13, ZIB, Takustr.7, 14195 Berlin, 2009. Cited on page 4.

[13] Ralf Borndörfer, Martin Grötschel, and Marika Karbstein. Angebotsplanung im öffentlichen Nahverkehr. In *DFG Forschungszentrum* MATHEON, Projekt B15, 2006–2014. Website at http://www.zib.de/Optimization/Projects/TrafficLogistic/Matheon-B15/index.de.html. Cited on page 3.

[14] Ralf Borndörfer, Martin Grötschel, and Andreas Löbel. Duty scheduling in public transit.

In Willi Jäger and Hans-Joachim Krebs, editors, *MATHEMATICS – Key Technology for the Future*, pages 653–674. Springer Verlag, Berlin, 2003. Cited on page 4.

[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. Cited on pages 7, 101, 102, 104, 105, 109, 110, 111, 112.

[16] Ralf Borndörfer, Martin Grötschel, and Marc E. Pfetsch. Models for line planning in public transport. In Mark Hickman, Pitu Mirchandani, and Stefan Voß, editors, *Computer-aided Systems in Public Transport (CASPT 2004)*, volume 600 of *Lecture Notes in Economics and Mathematical Systems*, pages 363–378. Springer-Verlag, 2008. Cited on pages 102, 110.

[17] Ralf Borndörfer and Marika Karbstein. A direct connection approach to integrated line planning and passenger routing. In Daniel Delling and Leo Liberti, editors, *ATMOS 2012 - 12th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems*, OpenAccess Series in Informatics (OASIcs), pages 47–57, Dagstuhl, Germany, 2012. Schloss Dagstuhl–Leibniz-Zentrum für Informatik. Cited on page 101.

[18] Ralf Borndörfer and Marika Karbstein. A note on Menger's theorem for hypergraphs. ZIB Report 12-03, ZIB, 2012. Cited on page 63.

[19] Ralf Borndörfer, Marika Karbstein, and Marc E. Pfetsch. Models for fare planning in public transport. *Discrete Applied Mathematics*, 160(18):2591–2605, 2012. Cited on pages 4, 5.

[20] Ralf Borndörfer, Marika Karbstein, and Marc E. Pfetsch. The Steiner connectivity problem. *Mathematical Programming*, 2012. Cited on pages 11, 39, 45, 94.

[21] Ralf Borndörfer, Andreas Löbel, and Steffen Weider. A bundle method for integrated multi-depot vehicle and duty scheduling in public transit. In Mark Hickman, Pitu Mirchandani, and Stefan Voß, editors, *Computer-aided Systems in Public Transport (CASPT 2004)*, volume 600 of *Lecture Notes in Economics and Mathematical Systems*, pages 3–24. Springer-Verlag, 2008. Cited on page 4.

[22] Ralf Borndörfer and Marika Neumann. Abschlussbericht für das Projekt Potsdam Linienplan 2010. Cited on pages 174, 179.

[23] Ralf Borndörfer and Marika Neumann. Models for line planning with transfers. ZIB-Report 10-11, Zuse Institute Berlin, 2010. Cited on page 101.

[24] Ralf Borndörfer and Marika Neumann. Linienoptimierung – reif für die Praxis? In *HEUREKA'11*. FGSV Verlag, 2011. Cited on pages 5, 152, 174, 179.

[25] Ralf Borndörfer, Marika Neumann, and Marc E. Pfetsch. Angebotsplanung im öffentlichen Nahverkehr. In *HEUREKA'08*. FGSV Verlag, 2008. Cited on pages 5, 105.

[26] Ralf Borndörfer, Marika Neumann, and Marc E. Pfetsch. The line connectivity problem. In *Operations Research Proceedings 2008*, pages 557–562. Springer-Verlag, 2009. Cited on pages 11, 39.

[27] Albert Bouma and Christine Oltrogge. Linienplanung und Simulation für öffentliche Verkehrswege in Praxis und Theorie. *Eisenbahntechnische Rundschau*, 43(6):369–378, 1994. Cited on page 103.

[28] Michael Bussieck. Gams – lop.gms: Line optimization. http://www.gams.com/modlib/libhtml/lop.htm. Cited on pages 89, 152.

[29] Michael R. Bussieck. *Optimal lines in public rail transport*. PhD thesis, TU Braunschweig, 1997. Cited on pages 102, 104, 118, 139.

[30] Michael R. Bussieck, Peter Kreuzer, and Uwe T. Zimmermann. Optimal lines for railway systems. *European Journal of Operations Research*, 96(1):54–63, 1997. Cited on pages 103, 116, 118.

[31] Michael R. Bussieck, Thomas Lindner, and Marco E. Lübbecke. A fast algorithm for near optimal line plans. *Mathematical Methods Operations Research*, 59(2), 2004. Cited on page 104.

[32] Michael R. Bussieck, T. Winter, and Uwe T. Zimmermann. Discrete optimization in public rail transport. *Mathematical Programming*, 79(1–3):415–444, 1997. Cited on page 4.

[33] Alberto Caprara and Matteo Fischetti. $\{0, \frac{1}{2}\}$-Chvátal-Gomory Cuts. *Mathematical Programming*, 74(3):221–235, 1996. Cited on page 60.

[34] Avishai Ceder and Nigel H. M. Wilson. Bus network design. *Transportation Research*, 20B(4):331–344, 1986. Cited on page 102.

[35] Sunil Chopra and Mendu R. Rao. The Steiner tree problem I: Formulations, compositions and extension of facets. *Mathematical Programming*, 64(2):209–229, 1994. Cited on pages 1, 6, 43.

[36] Vašek Chvátal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4(3):233–235, 1979. Cited on page 23.

[37] M. T. Claessens, Nico M. van Dijk, and Peter J. Zwaneveld. Cost optimal allocation of rail passanger lines. *European Journal of Operations Research*, 110(3):474–489, 1998. Cited on pages 102, 103, 104.

[38] Gérard Cornuéjols. *Combinatorial Optimization: Packing and Covering*. CBMS-NSF regional conference series in applied mathematics; 74. SIAM, Berlin, 2001. Cited on pages 51, 64, 69.

[39] Alysson M. Costa, Jean-Francois Cordeau, and Bernard Gendron. Benders, metric and cutset inequalities for multicommodity capacitated network design. *Comput Optim Appl*, 42:371–392, 2009. Cited on page 145.

[40] Geir Dahl and Mechthild Stoer. A cutting plane algorithm for multicommodity survivable network design problems. *INFORMS Journal on Computing*, 10(1):1–11, 1998. Cited on page 142.

[41] Sanjeeb Dash, Oktay Günlük, and Andrea Lodi. MIR closures of polyhedral sets. *Mathematical Programming*, 121:33–60, June 2009. Cited on page 143.

[42] Jacques Desrosiers and Marco E. Lübbecke. A primer in column generation. In G.Desaulniers, J.Desrosiers, and M.M. Solomon, editors, *Column Generation*, pages 1–32. Springer, Berlin, 2005. Cited on page 111.

[43] DFG reasearch center Matheon. Digitaler Adventskalender. http://www.mathekalender.de/. Cited on page 191.

[44] Annegret Dix. Das statische Linienplanungsproblem. Diploma thesis, TU Berlin, 2007. Cited on pages 135, 138, 161.

[45] Cee W. Duin. *Steiner's problem in graphs*. PhD thesis, University of Amsterdam, 1993. Cited on page 45.

[46] Uriel Feige. A threshold of ln n for approximating set-cover. *Proceedings of the 28th ACM Symposium on Theory of Computing*, pages 314–318, 1996. Cited on pages 6, 22, 23.

[47] Jon Feldman and Matthias Ruhl. The directed steiner network problem is tractable for a constant number of terminals. In *IEEE Symposium on Foundations of Computer Science*, pages 299–308, 1999. Cited on page 21.

[48] Gerd Fischer. *Lineare Algebra*, volume 11. Viewef Verlag, 1997. Cited on page 150.

[49] Lester Randolph Ford Jr. and Delbert Ray Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8:399–404, 1956. Cited on pages 66, 67.

[50] András Frank. Edge-connection of graphs, digraphs, and hypergraphs. Technical Report TR-2001-11, Egerváry Research Group, Budapest, 2001. Cited on pages 65, 66.

[51] András Frank. *Connections in combinatorial optimization*. Oxford University Press, Oxford, 2011. Cited on page 18.

[52] Delbert Ray Fulkerson. Blocking and anti-blocking pairs of polyhedra. *Mathematical Programming*, 1:168–194, 1971. Cited on page 64.

[53] Giorgio Gallo, Giustino Longo, and Stefano Pallottino. Directed hypergraphs and applications. *Discrete Applied Mathematics*, 42:177–201, 1993. Cited on page 18.

[54] Michel X. Goemans and David P. Williamson. A general approximation technique for constrained forest problems. *SIAM Journal on Computing*, 24(2):296–317, 1995. Cited on pages 6, 27, 33.

[55] Jan-Willem Goossens, Stan van Hoesel, and Leo Kroon. A branch-and-cut approach for solving railway line-planning problems. *Transportation Science*, 28(3):379–393, 2004. Cited on page 104.

[56] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric Algorithms and Combinatorial Optimization*, volume 2 of *Algorithms and Combinatorics*. Springer, 1988. Cited on pages 8, 59, 65.

[57] Martin Grötschel and Clyde L. Monma. Integer polyhedra arising from certain network design problems with connectivity constraints. *SIAM Journal on Discrete Mathematics*, 3(4):502–523, 1990. Cited on pages 53, 54.

[58] Martin Grötschel, Clyde L. Monma, and Mechthild Stoer. Computational results with a cutting plane algorithm for designing communication networks with low-connectivity constraints. *Operations Research*, 40:309–330, 1992. Cited on pages 57, 82.

[59] Oktay Günlük. A branch-and-cut algorithm for capacitated network design problems. *Mathematical Programming*, 86:17–39, 1999. Cited on page 82.

[60] Alan J. Hoffman. A generalization of max flow-min cut. *Mathematical Programming*, 6(1):352–359, 1974. Cited on page 67.

[61] IBM. ILOG CPLEX. http://www-01.ibm.com/software/integration/optimization/cplex/. Cited on pages 92, 165.

[62] M. Iri. On an extension of the maximum-flow minimum-cut theorem to multicommodity flows. *Journal of the Operations Research Society of Japan*, 13(3):129–135, 1971. Cited on page 137.

[63] IVU Traffic Technologies AG; Produkte für den öffentlichen Nahverkehr. IVU website. http://www.ivu.de/produkte-und-loesungen/busse-und-bahnen.html. Cited on page 4.

[64] Takeshita Kazuhiko, Fujito Toshihiro, and Watanabe Toshimasa. On primal-dual approximation algorithms for several hypergraph problems. *Joho Shori Gakkai Kenkyu Hokoku*, 99(15):13–18, 1999. Cited on pages 18, 37.

[65] Tamás Király. *Edge-connectivity of undirected and directed hypergraphs*. Phd thesis, Eötvös Loránd University, Budapest, 2003. Cited on pages 65, 66.

[66] Thorsten Koch and Alexander Martin. Solving Steiner tree problems in graphs to optimality. *Networks*, 32:207–232, 1998. Cited on pages 6, 45, 89, 92.

[67] Christian Liebchen. Der Berliner U-Bahn Fahrplan 2005 - Realisierung eines mathematisch optimierten Angebotskonzepts. In M. Boltze, editor, *Tagungsbericht der Heureka'05 – Optimierung in Transport und Verkehr*, pages 483–500. FGSV Verlag, 2005. Cited on page 5.

[68] Christian Liebchen. *Periodic Timetable Optimization in Public Transport*. Phd thesis, TU Berlin, 2006. Cited on pages 4, 5.

[69] Christian Liebchen. Linien-, Fahrplan-, Umlauf- und Dienstplanoptimierung: Wie weit können diese bereits integriert werden? In *HEUREKA'08*. FGSV Verlag, 2008. Cited on page 5.

[70] Andreas Löbel. Vehicle scheduling in public transport and lagrangean pricing. *Management Science*, 44(12-1):1637–1649, 1998. Cited on page 4.

[71] Thomas L. Magnanti and Prakash Mirchandani. Shortest paths, single origin-destination network design and associated polyhedra. *Networks*, 33:103–121, 1993. Cited on page 135.

[72] Thomas L. Magnanti and Richard T. Wong. Network design and transportation planning: Models and algorithms. *Transportation Science*, 1(18):1–55, 1984. Cited on page 135.

[73] Karl Menger. Zur allgemeinen Kurventheorie. *Fundamenta Mathematicae*, 10:96–115, 1927. Cited on page 65.

[74] Karl Nachtigall and Karl Jerosch. Simultaneous network line planning and traffic assignment. In Matteo Fischetti and Peter Widmayer, editors, *ATMOS 2008 - 8th Workshop on Algorithmic Approaches for Transportation Modeling, Optimization, and Systems*, Dagstuhl, Germany, 2008. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, Germany. Cited on page 105.

[75] Sang Nguyen and Stefano Pallottino. Hyperpaths and shortest hyperpaths. *Lecture Notes in Mathematics*, 1403:258–271, 1989. Cited on page 18.

[76] K. Onaga and O. Kakusho. On feasibility conditions of multicommodity flows in networks. *Transactions on Circuit Theory*, 18(4):425–429, 1971. Cited on page 137.

[77] James G. Oxley. *Matroid Theory*. Oxford University Press, Oxford, 1992. Cited on page 23.

[78] Marc E. Pfetsch and Ralf Borndörfer. Routing in line planning for public transportation. In H.-D. Haasis, editor, *Operations Research Proceedings 2005*, pages 405–410. Springer-Verlag, 2006. Cited on page 110.

[79] Tobias Polzin. *Algorithms for the Steiner Problems in Networks*. PhD thesis, University of Saarland, Saarbrücken, 2003. Cited on pages 2, 6, 45.

[80] Tobias Polzin and Siavash Vahdati Daneshmand. A comparison of Steiner tree relaxations. *Discrete Aplied Mathematics*, 112:241–261, 2001. Cited on page 45.

[81] Tobias Polzin and Siavash Vahdati Daneshmand. On Steiner trees and minimum spanning trees in hypergraphs. Technical Report 10.1.1.16.1056, MPI Informatik, 2001. Cited on page 18.

[82] Hans Jürgen Prömel and Angelika Steger. *The Steiner Tree Problem*. Vieweg, Braunschweig/Wiesbaden, 2002. Cited on pages 2, 6, 19, 33.

[83] ptv AG. Visum website. http://vision-traffic.ptvgroup.com/de/produkte/ptv-visum/. Cited on pages 152, 174, 183, 185, 186.

[84] Christian Raack. *Capacitated Network Design – Multi-Commodity Flow Formulations, Cutting Planes, and Demand Uncertainty*. Phd thesis, TU Berlin, 2012. Cited on pages 135, 137, 144.

[85] Christian Raack, Arie M. C. A. Koster, Sebastian Orlowski, and Roland Wessäly. On cut-based inequalities for capacitated network design polyhedra. *Networks*, 57(2):141–156, March 2011. Cited on page 161.

[86] John. T. Robacker. *Min-Max theorems on shortest chains and and disjunct cuts of a network*. ResearchMemorandum RM-1660, The RAND Corporation, Santa Monica, Carlifornia, 1956. Cited on page 65.

[87] Gabriel Robins and Alexander Zelikovsky. Tighter bounds for graph Steiner tree approximation. *SIAM Journal on Discrete Mathematics*, 19(1):122–134, 2005. Cited on page 29.

[88] Daniel J. Rosenkrantz, Richard Edwin Stearns, and Philip M. Lewis II. An analysis of several heuristics for the traveling salesman problem. *SIAM Jounral on Computing*, 6(3):563–581, 1977. Cited on page 31.

[89] S-Bahn Berlin. Notfahrplan Juli 2009. s-bahn-berlin.de/aktuell/2009/pdf/spinne_090720.pdf. Cited on page 1.

[90] Thomas Schlechte. *Railway Track Allocation: Models and Algorithms*. Phd thesis, TU Berlin, 2011. Cited on page 4.

[91] Marie Schmidt. Line planning with equilibrium routing. Technical Report 2012-9, Institut für Numerische und Angewandte Mathematik, Georg-August Universität Göttingen, Göttingen, Germany, 2012. Cited on page 104.

[92] Anita Schöbel. Line planning in public transportation: models and methods. *OR Spectrum*, pages 1–20, 2011. Cited on page 102.

[93] Anita Schöbel and Susanne Scholl. Line planning with minimal traveling time. In Leo G. Kroon and Rolf H. Möhring, editors, *Proceedings of 5th Workshop on Algorithmic Methods and Models for Optimization of Railways*, 2006. Cited on pages 7, 101, 102, 104, 105, 112, 169.

[94] Susanne Scholl. *Customer-Oriented Line Planning*. PhD thesis, University of Göttingen, 2005. Cited on page 104.

[95] SCIP – Solving Constraint Integer Programs. http://scip.zib.de. Cited on pages 7, 81, 92, 165.

[96] Stadtwerke Potsdam – ViP Verkehrsbetrieb Potsdam GmbH. ViP website. http://www.swp-potsdam.de/swp/de/verkehr/home-vip.php. Cited on pages 5, 152.

[97] Mechthild Stoer and Geir Dahl. A polyhedral approach to multicommodity survivable network design. *Numerische Mathematik*, 68:149–167, 1994. Cited on pages 135, 136, 137, 138, 141, 142, 146, 148.

[98] Transportation network test problems. http://www.bgu.ac.il/~bargera/tntp/. Cited on pages 89, 151, 152.

[99] H. Takahashi and A. Matsuyama. An approximate solution for the Steiner problem in graphs. *Mathematica Japonica*, 24:573–577, 1980. Cited on pages 29, 88.

[100] The Mathworks. Matlab. http://www.mathworks.de. Cited on page 4.

[101] Thomas Christof. PORTA - a POlyhedron Representation Transformation Algorithm. http://www.iwr.uni-heidelberg.de/groups/comopt/software/PORTA/. Cited on pages 142, 143.

[102] Luis M. Torres, Ramiro Torres, Ralf Borndörfer, and Marc E. Pfetsch. Line planning on paths and tree networks with applications to the Quito Trolebús System. *International Transactions in Operational Research (ITOR)*, 18(455–472), 2011. Cited on page 103.

[103] David M. Warme. *Spanning Trees in Hypergraphs with Applications to Steiner Trees*. PhD thesis, University of Virginia, 1998. Cited on page 18.

[104] Roland Wessäly. *Dimensioning Survivable Capacitated Networks*. PhD thesis, TU Berlin, 2000. Cited on page 136.

[105] Laurence A. Wolsey. An analysis of the greedy algorithm for the submodular set covering problem. *Combinatorica*, 2(4):385–393, 1982. Cited on pages 6, 23.

[106] Laurence A. Wolsey. *Integer Programming*. John Wiley & Sons, 1998. Cited on page 143.

[107] Kati Wolter. Implementation of cutting plane separators for mixed integer programs. Diploma thesis, TU Berlin, 2006. Cited on page 160.