Teodor Gabriel Crainic
Michel Gendreau
Bernard Gendron *Editors*

Network Design with Applications to Transportation and Logistics

Springer

Network Design with Applications to Transportation and Logistics

Teodor Gabriel Crainic • Michel Gendreau
Bernard Gendron
Editors

Network Design with Applications to Transportation and Logistics

Springer

Editors
Teodor Gabriel Crainic
CIRRELT and AOTI
Université du Québec à Montréal
Montréal, QC, Canada

Michel Gendreau
CIRRELT and MAGI
Polytechnique Montréal
Montréal, QC, Canada

Bernard Gendron
CIRRELT and Département d'informatique
et de recherche opérationnelle
Université de Montréal
Montréal, QC, Canada

ISBN 978-3-030-64017-0　　ISBN 978-3-030-64018-7 (eBook)
https://doi.org/10.1007/978-3-030-64018-7

À nos épouses Diane, Johanne et Wissal et à nos enfants Lucie, Manon, Catherine, Laurent et Gabrielle.

Contents

Chapter 1
A Book About Network Design

Teodor Gabriel Crainic, Michel Gendreau, and Bernard Gendron

1 Introduction

Network design problems arise whenever optimal choices have to be made that can be represented conceptually as the selection of a subset of links in a graph. Typically, these optimal choices are the result of complex tradeoffs between various types of costs and constraints. In particular, most network design problems involve fixed costs associated with link selection and variable costs associated with flows (of people, goods, information,…). Because of their combinatorial nature and the complexity of their objective functions and constraints, network design problems are inherently difficult (most of them are $\mathscr{N}\mathscr{P}$-hard). For this reason, models and algorithms for network design problems involve approaches from several areas of combinatorial optimization and mathematical programming.

In this book, we study network design problems, as well as models and algorithms to solve them, through techniques derived from network optimization, linear programming (LP), mixed-integer linear programming (MILP), metaheuristics, and large-scale optimization. The book is intended to be useful not only to researchers

T. G. Crainic (✉)
CIRRELT and AOTI, Université du Québec à Montréal, Montréal, QC, Canada
e-mail: TeodorGabriel.Crainic@cirrelt.net

M. Gendreau
CIRRELT and MAGI, Polytechnique Montréal, Montréal, QC, Canada
e-mail: Michel.Gendreau@cirrelt.net

B. Gendron
CIRRELT and Département d'informatique et de recherche opérationnelle,
Université de Montréal, Montréal, QC, Canada
e-mail: Bernard.Gendron@cirrelt.net

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_1

specialized in these areas, but first and foremost, to graduate students in operations research, management science, analytics, and related fields. A special effort was made by all authors involved in the different chapters to make the exposition didactic. In particular, to facilitate reading, the main body of each chapter contains very few citations. Instead, a section on bibliographical notes is included at the end of each chapter, helping the reader to summarize the content of the chapter, to get an appreciation of the historical development of the particular topic addressed in the chapter, and to look for additional references. We assume that the reader is familiar with the basics of LP and MILP, including an exposition to linear network flow problems. More advanced topics, such as mathematical decomposition (including Lagrangian duality and Benders decomposition) and metaheuristics, are reviewed in Chaps. 3 and 4, although they are only covered in enough details to make the exposition self-contained. The reader who is less familiar with these topics would benefit from reading the articles and books that are referenced in these chapters.

Network design problems are prominent in transportation and logistics, but also in other areas, such as telecommunication and manufacturing. This book is about applications in transportation and logistics, although many problems, models and algorithms can be adapted to other areas. In fact, the first ten chapters of the book are dedicated to the network design methodology, common to most problem settings and applications, and many application-oriented chapters contain material that is highly relevant to other areas than transportation and logistics. Even though we purposely restricted ourselves to transportation and logistics, covering all applications of network design in that area is impossible and we had to make some difficult choices. In particular, the broad areas of facility location and supply chain management yield optimization problems that can be cast as the selection of a subset of nodes in a graph. By duplicating nodes and by introducing links between each node and its "copy," such problems can be seen as involving the selection of a subset of links, hence as network design problems. As such, they are covered in some chapters of the book, but not to the same extent as they are in books dedicated to facility location and supply chain management.

In Sect. 2, we summarize the contents of the book, including a short description of each chapter. Section 3 contains bibliographical notes on books related to the prerequisites mentioned above. We comment on the origins of the book and the research perspectives that we hope it will stimulate in Sect. 4. We conclude this introductory chapter with a few words of thanks.

2 Contents of the Book

The book is divided into three parts. Part I, entitled *Basic Problems and Models*, includes Chaps. 2 to 4 and focuses on models and algorithms for fixed-charge network design problems. Such problems display the typical objective function that involves fixed design costs and variable flow costs, which is common to most network design problems. Problems studied in Part I are otherwise assumed to be deterministic and their structure to be relatively "simple," typically including only flow conservation and capacity constraints. Part II, entitled *Advanced Problems and Models*, contains Chaps. 5 to 11. It is dedicated to network design problems and models that involve more complex objective functions and constraints. For instance, Part II studies problems that include non-linear objective functions, topological constraints, and uncertain data. Part III, entitled *Applications in Transportation and Logistics*, addresses network design problems encountered in the areas of transportation and logistics. Both people and freight transportation are considered, by all modes. Chapters 12 to 20 make up this part of the book. We now review in more details the content of each chapter in the three parts of the book.

2.1 Part I: Basic Problems and Models

Chapter 2, *Fixed-Charge Network Design Problems*, by Crainic, Gendreau, and Gendron, introduces problems and models that involve design decisions captured with arc-based binary variables. In particular, this chapter deals with problems and models that involve fixed design costs in the objective function. Problems and models are distinguished according to several characteristics, in particular: whether the demand can be represented as a single commodity or as multiple commodities; whether or not there are arc capacities; and whether or not there are variable transportation costs in the objective function, in addition to the fixed design costs. These characteristics yield different variants of network design problems and models, with various degrees of complexity. A few basic modeling approaches are studied, including the use of path flow variables instead of arc flow variables and the derivation of cut-set-based inequalities.

Chapter 3, *Exact Methods for Fixed-Charge Network Design*, by Crainic and Gendron, focuses on exact algorithms for single-commodity and multicommodity fixed-charge network design. These problems are notoriously difficult, since they are in general strongly $\mathscr{N}\mathscr{P}$-hard. Multicommodity capacitated fixed-charge problems are particularly challenging, since even their LP relaxations are difficult to solve, and the formulations display a large number of variables and constraints. In this context, decomposition methods that exploit subproblem structures are particularly useful. This chapter is divided into three parts. Part I presents relaxations that can improve the quality of the lower bounds, while exploiting the subproblem structures. Part II focuses on enumeration algorithms that use the modeling techniques covered in Part I. Part III is dedicated to the solution of large-scale instances by heuristic methods and parallel algorithms that exploit the techniques presented in the first two parts.

Chapter 4, *Heuristics and Metaheuristics for Fixed-Charge Network Design*, by Crainic and Gendreau, considers the heuristics and metaheuristics that are widely used to tackle difficult network design problems. The chapter begins with a presentation of fundamental concepts for the development of heuristic approaches, such as search spaces, neighborhoods, and populations of solutions. The main heuristic and metaheuristic solution methods are then introduced: constructive and local search heuristics, neighborhood-based metaheuristics, population-based methods, matheuristics, parallel meta- and matheuristics. The chapter focuses on the application of these methods to fixed-charge transportation and multicommodity capacitated fixed-charge network design problems. It also provides a historical perspective on the development of the field since the 1960's, as well as a number of challenging research avenues for meta- and matheuristics for network design problems.

2.2 Part II: Advanced Problems and Models

Chapter 5, *Multicommodity Multifacility Network Design*, by Atamtürk and Günlük, studies multicommodity network design models where capacity can be added to the arcs using multiples of facilities that might have different capacities. This class of models appears frequently in supply chain design, service network design, and telecommunication network capacity expansion problems. Valid inequalities used as cutting planes in enumeration algorithms have been instrumental in solving large-scale instances. This chapter reviews advances in polyhedral theory for this class of models by emphasizing three fundamental techniques: metric inequalities for projecting out continuous flow variables; mixed-integer rounding from appropriate base relaxations; and shrinking the network to a small k-node graph. The basic inequalities derived from arc-set, cut-set, and partition relaxations are also useful for solving robust and survivable network design models.

Chapter 6, *Piecewise Linear Cost Network Design*, by Frangioni and Gendron, considers one of the most important extensions to "basic" network design models required to accurately model real-world applications: the fact that the capacity on the arcs does not come in an "all-or-nothing" fashion, but with a more complex cost. Every reasonable cost-of-capacity function can be approximated as a piecewise linear one, where the extent of the approximation can be tightly controlled at the cost of the number of breakpoints. This chapter focuses on such models, considering both a general case and some more restricted ones that serve to illustrate the main concepts. It is shown that the best models in terms of tightness of the LP relaxation bound suffer from a significant increase in the number of variables. Techniques that efficiently solve very large formulations are presented, showing that they are instrumental for the practical solution of piecewise linear cost network design problems.

Chapter 7, *Topology-Constrained Network Design*, by Fortz, studies models and techniques for long-term planning of networks for which demands are not known in advance. In this case, the objective is to build a network at minimum cost, considering only the fixed cost associated with opening a link. Capacity and routing costs are ignored. Nevertheless, the network is subject to topological constraints to ensure its connectivity and survivability. This chapter covers the design of connected networks (in particular, the minimum spanning tree problem), followed by the design of networks requiring a higher level of survivability in terms of the number of available node-disjoint paths, to allow re-routing in case of failures. To avoid delays in the networks, models where the length of paths is bounded are also studied, by introducing hop constraints or covering of the links by cycles of bounded lengths.

Chapter 8, *Network Design with Routing Requirements*, by Balakrishnan, Magnanti, Mirchandani, and Wong, addresses network design problems in which constraints on flow routing decisions are imposed to ensure good end-to-end service performance. The chapter discusses modeling and methodological issues for effectively solving fixed-charge network design problems with routing requirements. This problem is $\mathscr{N}\mathscr{P}$-hard; the added routing restrictions increase computational difficulty even to find feasible solutions. The chapter first discusses recent results and a composite algorithm that combines problem reduction, valid inequalities, and heuristics with branch-and-bound to effectively solve problem instances with varying characteristics. Theoretical developments, modeling strategies, and algorithms for two well-studied special cases of the problem are presented next. This part focuses on constrained shortest path and hop-constrained network design models, presenting approximation algorithms, polyhedral results, extended model formulations, and specialized algorithms. The chapter concludes with decomposition solution methods, and a number of key observations and insights into addressing the routing-constrained network design problem.

Chapter 9, *Bilevel Network Design*, by Labbé and Marcotte, is dedicated to network design problems involving conflicting agents, referred to as the designer and the users, respectively. This paradigm is especially relevant when the designer of a network does not have a direct control of user flows, who are assigned according to their own logic. Such problems are best cast into the framework of bilevel programming, where the designer anticipates the reaction of rational users to its course of action, which fits many situations of interest. This chapter considers four applications of very different nature: the continuous network design problem; a competitive location-queuing model; the network pricing problem; and the bilevel network interdiction problem. Algorithmic issues are particularly emphasized.

Chapter 10, *Stochastic Network Design*, by Hewitt, Rei, and Wallace, study problems that explicitly account for various sources and levels of uncertainty. The chapter focuses on stochastic network design, addressing both the main modeling paradigms and the solution methods that can be applied. The paradigms are first illustrated on the stochastic fixed-charge capacitated multicommodity network design problem. This is followed up with a presentation of how scenario generation is applied to approximate the random distributions used to model the stochastic parameters in network design models. The general solution approaches, both exact and heuristic, that can be applied to solve stochastic network design models are then described, emphasizing how decomposition strategies may be used to produce more efficient solution processes for the considered models. The chapter concludes with perspectives regarding the future research in the field.

Chapter 11, *Robust Network Design*, by Koster and Schmidt, examines the robust network design problem, which is a network design problem under demand uncertainty, but in the framework of robust optimization. It is important to highlight that in this framework, the uncertainty of input parameters is approached by finding a solution that is feasible for all considered input vectors. This is particularly important when the design process involves long-term or strategic decisions, since the quality of demand forecasts determines the feasibility of the network design for its future purpose. After a brief introduction to robust optimization, its application to single- and multicommodity network design is presented. At appropriate times, extensions of the basic idea of robust optimization are also introduced.

2.3 Part III: Applications in Transportation and Logistics

Chapter 12, *Service Network Design*, by Crainic and Hewitt, opens the third part of the book, which focuses on network design as core methodology to address planning and assist decision making in various application areas. The chapter addresses service network design, the term designating issues, decisions, and network-design models targeted to planning the activities and resources of the supply side of a transportation or logistics system, with the general goal of satisfying demand efficiently, profitably, and within the quality standards agreed upon with the customers generating this demand. Service network design is particularly relevant in the context of consolidation-based transportation, an umbrella term for companies and systems which group and transport within the same vehicle several freight loads of different customers, aiming for a profitable balance between economy-of-scale-based costs and high service quality for customers. This chapter presents a comprehensive overview of the general service network design methodology, which is to be found in many application fields, many of which are addressed in the following chapters, including railroads, less-than-truckload motor carriers, land and water-based intermodal transport, and city logistics. The chapter starts with an overview of consolidation-based freight carriers and planning issues, situating service network design in this context. The static version of the problem is described next, followed by the time-dependent service network design problems where the schedules of the selected services are part of the design. The challenging issue of representing the management of resources into tactical planning is considered next, followed by how uncertainty may be addressed. The chapter concludes with bibliographical notes, which also address algorithmic issues, and with a research agenda for service network design.

Chapter 13, *Freight Railroad Service Network Design*, by Chouman and Crainic, examines the relation between railroad planning for freight transportation and network design. Rail transportation provides economically-priced, environmentally-friendly, and timely freight transportation services at all levels. To achieve this performance, railroads operate mostly according to a double consolidation policy, as cars are grouped into blocks, which are grouped into trains, and set up tactical operations plans specifying the train services and schedules to operate, the blocks to build at each terminal, the routing of blocks and cars, and the resource assignment to support these operations. Tactical planning is thus a very complex problem. Operations research provides the service network design methodology to build the railroad tactical plan making the most efficient use of the railroad's resources to achieve its performance objectives. The chapter focuses on service network design models for railroad tactical planning, targeting both particular activities, such as car blocking and train makeup, and integrated network-wide planning processes. Static and time-dependent problem contexts and models are presented. Particular attention is devoted to models and methods for integrated planning.

Chapter 14, *Motor Carrier Service Network Design*, by Bakir, Erera, and Savelsberg, discusses the service network design models and solution methodologies specifically focused on problems that arise in the planning of operations in the trucking, or motor freight, industry. Consolidation carriers such as less-than-truckload and package trucking companies face flow planning problems to decide how to route freight between transfer terminals, and load planning problems to decide how to consolidate shipments into trailer-loads and container-loads for dispatch. Integer programming models are introduced for these network design decision problems as well as exact and heuristic solution methods.

Chapter 15, *Liner Shipping Network Design*, by Christiansen, Hellsten, Pisinger, Sacramento, and Vilhelmsen, studies issues related to the construction of service networks for long-haul liner-shipping navigation. Liner shipping is the service of transporting large volumes of cargo using ocean-going vessels, sailing regular routes on fixed schedules. Designing a good network is a complex task, in which many aspects have to be taken into account. The chapter gives a brief introduction to containerised liner shipping, RoRo liner shipping, and network design, and introduces the LINER-LIB test instances for network design in containerised liner shipping. The most common network design models for containerised liner shipping are presented, including, integrated Mixed Integer Programming models, and two-stage algorithms where the service generation and the flowing of containers are separated into two steps. The chapter concludes with a discussion of future trends in liner shipping, indicating directions for future research.

Chapter 16, *City Logistics*, by Crainic, Perboli, and Ricciardi, examines City Logistics systems, which aim to reduce the nuisances associated with freight transportation within urban areas, while sustaining the social and economic development of the organizations and cities involved. City Logistics displays several core characteristics, e.g., cooperation among stakeholders, consolidation of cargo of different stakeholders within the same vehicles, synchronization of operations, resource sharing, multi and intermodal operations, which makes for complex planning problems. Network design is one the main methodologies used to address these issues. The particular settings and characteristics of City Logistics systems bring modeling challenges and lead to new formulations that account for several layers of facilities and operations, time-dependency of demand and activities, synchronization of fleets at terminals, integration of private and public transportation and logistic means, and that combine network design and vehicle routing. This chapter aims to capture these characteristics, present the network design methodology currently available, and identify promising research avenues for City Logistics and network design.

Chapter 17, *Public Transportation*, by Mauttone, Cancela, and Urquhart, focuses on network design methodologies applied to problems arising at the strategic and tactical planning of public transportation systems, including both urban and intercity services. The main problems discussed are the design of bus, rail and metro networks, which involve making decisions over links or groups of links from a given underlying network. In general terms, the resulting network should take into account the interests of different stakeholders, namely, the ones who perceive the cost of traveling across the network and those who perceive the cost of building the infrastructure and operating the services over it. The chapter presents several mathematical formulations, including aspects like passenger behavior, multiple objectives and multiple levels of decisions, as well as an overview of solution approaches covering both exact and heuristic methods and considering several sub-problems like route generation, route selection, and route set generation and improvement.

Chapter 18, *Hub Network Design*, by Contreras, examines a network design problem lying at the heart of network design planning in transportation and telecommunications systems. Hub-based networks provide connections between many origins and destinations via hub facilities that serve as transshipment, consolidation, or sorting points for commodities. Hub facilities help to reduce the number of required arcs to connect all nodes and enable economies of scale due to the consolidation of flows on relatively few arcs. Hub network design can be seen as a class of multicommodity network design problems in which node selection decisions are taken into account. This chapter overviews the key features of hub networks, the types of decisions that are usually considered when designing them, and how these decisions interact between them. It also describes commonly considered assumptions and properties and highlights how these impact the formulation and solution of various classes of hub network design problems.

Chapter 19, *Logistics Network Design*, by Cordeau, Klibi, and Nickel treats one of the most important areas of application for multi-commodity network design models, namely the design of logistics networks (or supply chains). Logistics networks connect suppliers, manufacturing plants, warehouses, distribution centers and customers to coordinate the acquisition of raw materials and components, their transformation into finished products, the movements of materials and components, and the delivery of the products to customers. The realism of logistics network design models has greatly improved over the last 40 years, and efficient solution methods have been developed to solve these models. There is now a vast literature on the topic with a very large number of models addressing the many problem variants encountered in practice. This chapter provides a general modeling framework that can be used to express many of these variants and gives a brief overview of the main solution methodologies. It also discusses two important and recent trends: the treatment of risk and uncertainty in the design of logistics networks and the incorporation of environmental, sustainability and reverse logistics aspects.

Chapter 20, *Collaboration in Transport and Logistics Networks*, by Hezarkhani, Slikker, and van Woensel, looks into issues that are becoming increasingly prominent in freight transportation and logistics, namely, stakeholder collaboration, to achieve economies of scale, as well as better resource utilization and decreased negative social impacts. The success of new network design concepts building on the domains of the Physical Internet, City Logistics, synchromodal networks, etc. is thus depending for to a large part upon the ability of stakeholders to successfully collaborate and agree on cost, benefit, risk, and resource sharing mechanisms. Yet, designing a fair cost-and-benefit sharing scheme is a major impediment for collaboration. The purpose of this chapter is to provide an overview of approaches in dealing with cost sharing problems in collaborative logistics and network design. The chapter discusses cost-sharing problems in some basic and stylized network design models as well as in the context of more operational problems in collaborative transport and logistics. Two alternative approaches to addressing cost-sharing problems are identified. The first defines a cooperative game associated with the situation and uses cooperative game theory to come up with allocations and/or cost shares. The second approach deals directly with the situation at hand and obtains cost shares using the information contained in the problem setting. In this approach, the solution often relies on the structure of the underlying optimization problem. The specific features of cooperative situations provide grounds for refining well-known solutions in cooperative game theory or develop new ones that are appropriate for special situations.

3 Bibliographical Notes

As mentioned in the Introduction, we assume the reader is familiar with the areas of LP, MILP and network optimization. Many excellent books on these topics can be recommended, among which we cite: Chvátal (1983); Schrijver (1986); Nemhauser and Wolsey (1988); Ahuja et al. (1993); Wolsey (1998); Conforti et al. (2014). The book does not cover in detail the related research on facility location and supply chain management, on which there are several books, including: Laporte et al. (2015); Shapiro (2007); Goetschalckx (2011). For references on the history of network design research, we refer the reader to the bibliographical notes at the end of Chap. 2.

4 Conclusions and Perspectives

We started the adventure of writing this book more than 10 years ago. Our initial intent was to publish "the definitive" work on network design in transportation and logistics. Soon, we realized that writing the book alone, without the help of our colleagues and friends, would be sheer utopia. In particular, covering all aspects of the topic is an immense task. Some 5 years ago, we asked the best experts on the different subjects to join our efforts to finally be able to produce the book. We warmly thank our colleagues who wrote the different chapters, as well as the reviewers who helped us to improve the final product.

Since the early days of our own research on network design in the 80’s and 90’s, a lot has been achieved in solving these problems and in expanding the applications. The evolution of the field of combinatorial optimization during the last 30 years is no stranger to this fact. The area of metaheuristics was still in its infancy in 1990. Significant progresses were achieved in the 1990’s when different paradigms were explored on many problems, most prominently derived from applications in transportation and logistics. Network design was one important class of such problems on which metaheuristics proved their utility and we modestly contributed to that evolution, soon realizing that hybrid metaheuristics that exploit mathematical programming were needed. In parallel with that evolution, in the early 2000’s, MILP research has integrated the principles of metaheuristics, expanding the capabilities of state-of-the-art solvers. This major change immediately followed the fruitful line of research on polyhedral theory that led to the development of general-purpose cutting-plane methods that changed the face of industrial MILP solvers. Network design problems were instrumental in achieving these progresses, as their basic fixed-charge flow structure appears in so many formulations. On the side of exact methods, decomposition algorithms, initially proposed in the early 60’s, were rediscovered in the last 40 years and led to significant advances in many areas, including transportation and logistics. Again, network design was at the heart of this evolution.

Significant progress has thus been made in the quest for solving large-scale network design problems. Simultaneously, one could observe a significant broadening of the scope of network design applications, and a continuous emulation between problem definition and modeling, on the one hand, and problem-solving methodology, on the other hand. It is this emulation and cross-fertilization that made the field grow and will continue to steadily and strongly do so. A lot remains to be done.

Each chapter of this book discusses perspectives for future avenues of research. It it not our intent here to summarize these specific conclusions, but rather to identify what we think will be general directions of research on network design in the coming years. On the problem-definition side, the contemplated problems will continue growing in complexity, explicitly accounting for, e.g., several layers of decisions, synchronization of decisions and decision makers, and uncertainty, to name but a few. With respect to solution methods, we believe, first, that the trend of research on matheuristics, the hybrid algorithms that combine metaheuristics and mathematical programming is far from over, and will be important in making significant progress in solving large-scale network design problems. Second, decomposition methods will expand their applicability, with the help of industrial MILP solvers, which gradually integrate these approaches. Instead of seeing MILP solvers as “competitors,” researchers will look at them as allies. As a result, more sophisticated decomposition schemes will be developed, exploiting the inherent structure of network design models to solve realistically-sized instances. Third, the area of parallel combinatorial optimization, to which we modestly contributed, will expand in the years to come. The goal will be to develop algorithmic paradigms that exploit the current knowledge, but escape from the sequential way of thinking that is too often implicitly assumed, in particular in mathematical programming. We are confident that these general perspectives, as well as those more specific identified in the different chapters of the book, will lead to fruitful research agendas that will help us to solve large-scale network design problems and to expand their applicability in many areas of transportation and logistics.

Acknowledgments First and foremost, we wish to address our warmest thanks to the authors of the different chapters who accepted so graciously our invitation to contribute to the book. Without their generous involvement, the book would not show such a rich variety of topics, allowing us to cover most of the models, methods and applications of network design in transportation and logistics. We also want to thank the reviewers, who dedicated time and effort to help the authors and ourselves to improve the final product.

We owe so much to our collaborators and students who worked with us over the years on developing models and methods for network design problems. Our research efforts also benefited from the environment offered by the CIRRELT, its staff and its professionals. We thank these individuals, as well as the FRQNT (Québec) and our universities, UQÀM, Polytechnique Montréal and Université de Montréal, for their financial contributions and the infrastructures and resources they offer through the CIRRELT. We also gratefully acknowledge the support of the Government of Canada, through its NSERC programs, of the Government of Québec, through its FRQNT programs, and of our partners in public and private institutions. The three funding sources have generously contributed to our research developments.

Teodor Gabriel Crainic holds the Chair on Intelligent Logistics and Transportation Systems Planning of the Université du Québec à Montréal. The financial contributions of ClearDestination and NSERC (Canada) that support the Chair are gratefully acknowledged. He is Adjunct Professor at the Département d'informatique et de recherche opérationnelle (DIRO), Université de Montréal, and Co-Director, Intelligent Transportation Systems Laboratory, Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT), Canada.

Michel Gendreau is Professor of Operations Research in the Department of Mathematics and Industrial Engineering of Polytechnique Montréal. The support of the department and Polytechnique to his research activities is gratefully acknowledged.

Bernard Gendron holds the Purolator Research Chair on Data Intelligence for Logistics at the Département d'informatique et de recherche opérationnelle (DIRO), Université de Montréal. The financial contributions of Purolator, NSERC (Canada) and Prompt (Québec) that support the Chair are gratefully acknowledged. He also holds the Research Chair on the Transformation of Transport at DIRO, financially supported by the MEI (Québec), whose support is also gratefully acknowledged.

References

Ahuja, R., Magnanti, T. L., & Orlin, J. B. (1993). *Network flows: theory, algorithms, and applications*. Upper Saddle River: Prentice Hall.
Chvátal, V. (1983). *Linear programming*. New York: W.H. Freeman.
Conforti, M., Cornuéjols, G., & Zambelli, G. (2014). *Integer programming*. Berlin: Springer.
Goetschalckx, M. (2011). *Supply chain engineering*. Berlin: Springer.
Laporte, G., Nickel, S., & Saldanha da Gama, F. (Eds.) (2015). *Location science*. Berlin: Springer.
Nemhauser, G. L., & Wolsey, L. A. (1988). *Integer and combinatorial optimization*. Hoboken: John Wiley and Sons.
Schrijver, A. (1986). *Theory of linear and integer programming*. Hoboken: John Wiley and Sons.
Shapiro, J. F. (2007). *Modeling the supply chain*. Pacific Grove: Thomson Brooks/Cole.
Wolsey, L. A. (1998). *Integer programming*. Hoboken: John Wiley and Sons.

Part I
Basic Design Problems

Chapter 2
Fixed-Charge Network Design Problems

Teodor Gabriel Crainic, Michel Gendreau, and Bernard Gendron

1 Introduction

This chapter sets the stage for the remaining chapters of the book. The main goal of in this chapter is to introduce problems and models that involve design decisions captured with arc-based binary variables. These variables represent building an infrastructure (e.g., roadways or railbeds) or establishing a transportation service (such as a bus or railway line, along with its schedule). Associated with these decisions are fixed costs that appear in the objective function or in constraints, typically then to represent budget limitations. While this chapter focuses on problems and models for fixed-charge network design, the next two chapters deal with exact and heuristic algorithms for solving such problems.

A fundamental distinction in network design problems is whether the demand can be represented as one commodity (possibly with multiple origins and multiple destinations) or as multiple commodities that need to be differentiated. The underlying problems that represent the transportation decisions, once the design decisions are taken, are significantly different in terms of their complexity and size, even though

T. G. Crainic
CIRRELT and AOTI, Université du Québec à Montréal, Montréal, QC, Canada
e-mail: TeodorGabriel.Crainic@cirrelt.net

M. Gendreau
CIRRELT and MAGI, Polytechnique Montréal, Montréal, QC, Canada
e-mail: Michel.Gendreau@cirrelt.net

B. Gendron (✉)
CIRRELT and Département d'informatique et de recherche opérationnelle,
Université de Montréal, Montréal, QC, Canada
e-mail: Bernard.Gendron@cirrelt.net

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_2

both can be cast as network flow problems. This chapter follows this fundamental distinction, with Sect. 2 dedicated to single-commodity formulations and Sect. 3 to multicommodity formulations.

Another important characteristic of network design problems is whether or not there are capacities. Indeed, the presence of capacities greatly complicates the task of finding feasible solutions. This is true especially in the multicommodity case, when capacities bind commodities together. We also consider the cost structure as another significant feature of network design problems, since the presence of fixed design costs and variable transportation costs in the objective function introduces complex trade-offs that further complicate the identification of optimal solutions.

In addition to the problems' features, we identify in this chapter a few basic modeling approaches for network design. First, while the transportation decisions are typically represented with arc flow variables, it is also possible to use path flow variables, which introduces models with an exponential number of variables that can be handled with column generation algorithms (see Chap. 3). Such path-based formulations can be of interest for computational reasons, but also for modeling purposes, since they permit representing path-dependent costs that are not additive by arc. Second, when transportation costs are ignored, flow variables can be projected out. In the single-commodity case, we can then exploit the max-flow-min-cut theorem to derive inequalities that are easy to generate within cutting-plane algorithms. Although these inequalities do not suffice in the multicommodity case, they are still useful, as will be seen in Chap. 3.

2 Single-Commodity Formulations

Let $\mathscr{G} = (\mathscr{N}, \mathscr{A})$ be a directed graph, where $\mathscr{N}$ is the set of nodes and $\mathscr{A} \subseteq \mathscr{N} \times \mathscr{N}$ is the set of potential arcs (in some situations, parallel arcs might be allowed and might even simplify the models, see, e.g., Chap. 6). A limited flow capacity $u_{ij} > 0$ is associated with each arc $(i, j) \in \mathscr{A}$. The network design problem consists of selecting a subset of arcs from $\mathscr{A}$ to satisfy a given demand at minimum total cost.

The demand to satisfy is defined at the nodes of the graph, which are partitioned into three subsets: $\mathscr{N}^o$, the set of *origin* (*source*) nodes, $\mathscr{N}^d$, the set of *destination* (*sink*) nodes, and $\mathscr{N}^t$, the set of *transshipment* (*intermediate*) nodes. Each origin $i \in \mathscr{N}^o$ has a *supply* (*availability*) $w_i > 0$ of the given commodity, each destination $i \in \mathscr{N}^d$ has a *demand* (*request*) $w_i < 0$ of the same commodity, while each transshipment node $i \in \mathscr{N}^t$ has neither availability nor request, i.e., $w_i = 0$. The *net supply* across any set $\mathscr{S} \subseteq \mathscr{N}$ is defined as $W(\mathscr{S}) \equiv \sum_{i \in \mathscr{S}} w_i$. We assume that demand is balanced, i.e., $W(\mathscr{N}) = 0$. Standard network flow transformations (adding a dummy origin or a dummy destination) might be applied when this is not the case. For instance, if $W(\mathscr{N}) > 0$, we add a dummy destination j, with demand -$W(\mathscr{N})$, that is connected to every origin i by an arc (i, j) of capacity $W(\mathscr{N})$ and arbitrarily large fixed cost.

The total cost of satisfying the demand consists of the sum of the costs to select the arcs to be included in the final design and the transportation costs to move the flow. For each potential arc $(i, j) \in \mathscr{A}$, let $f_{ij} \geq 0$ be the fixed *design cost* charged whenever the arc is selected for inclusion in the optimal design, and $c_{ij} \geq 0$ the unit *transportation cost*.

We introduce binary design variables y_{ij}, $(i, j) \in \mathscr{A}$, indicating if arc (i, j) is included (*open*, $y_{ij} = 1$) or not (*closed*, $y_{ij} = 0$) in the final design, and continuous flow variables $x_{ij} \geq 0$, $(i, j) \in \mathscr{A}$, equal to the amount of flow on each arc. The mathematical formulation of the *single-commodity capacitated fixed-charge network design* problem (*SCFND*) can then be written as

$$\text{Minimize} \quad \sum_{(i,j)\in\mathscr{A}} \left(f_{ij} y_{ij} + c_{ij} x_{ij}\right) \tag{2.1}$$

$$\text{Subject to} \quad \textstyle\sum_{j\in\mathscr{N}_i^+} x_{ij} - \sum_{j\in\mathscr{N}_i^-} x_{ji} = w_i, \quad \forall\, i \in \mathscr{N}, \tag{2.2}$$

$$x_{ij} \leq u_{ij} y_{ij}, \qquad \forall\, (i, j) \in \mathscr{A}, \tag{2.3}$$

$$x_{ij} \geq 0, \qquad \forall\, (i, j) \in \mathscr{A}, \tag{2.4}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall\, (i, j) \in \mathscr{A}, \tag{2.5}$$

where, for each $i \in \mathscr{N}$, we define

$$\mathscr{N}_i^+ = \{j \in \mathscr{N} : (i, j) \in \mathscr{A}\}, \quad \mathscr{N}_i^- = \{j \in \mathscr{N} : (j, i) \in \mathscr{A}\}.$$

The objective function (2.1) minimizes the total cost computed as the sum of the total fixed cost for arcs included in the optimal design and the total transportation cost for the commodity. Constraints (2.2) enforce flow conservation at nodes, while constraints (2.3) are the so-called *linking constraints* guaranteeing that flows use open arcs only and are less than the corresponding arc capacities.

Some variants of the problem include a budget constraint on any one of the main cost terms, design or transportation, the most common being a design budget constraint

$$\sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} \leq B, \tag{2.6}$$

where $B > 0$ is the global design budget. When such a constraint is added, the corresponding fixed cost term in the objective function is typically removed, which yields the *single-commodity capacitated budget network design* problem (*SCBND*).

Model (2.1)–(2.5) is also known as the *arc-based formulation*. An equivalent *path-based formulation* can be derived by using the fact that the arc flows can be decomposed into a finite set of path flows, each path connecting an origin to a destination. Note that there always exists an optimal solution with no flows on directed cycles, since all costs are nonnegative. For this reason, unless otherwise stated, we assume that all paths are elementary (without any directed cycle) and simply use the term "path" to designate an elementary path. Let $\mathscr{P}$ be the set of paths, each path $p \in \mathscr{P}$ connecting an origin to a destination, and $h_p \geq 0$ the amount of flow on path $p \in \mathscr{P}$. The path-based formulation of the *SCFND* can then be written as

$$\text{Minimize} \quad \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + \sum_{p\in\mathscr{P}} e_p h_p \tag{2.7}$$

$$\text{Subject to} \quad \textstyle\sum_{j\in\mathscr{N}_i^+} \sum_{p\in\mathscr{P}} \delta_{ij}^p h_p = |w_i|, \quad \forall\, i \in \mathscr{N}^o, \tag{2.8}$$

$$\textstyle\sum_{j\in\mathscr{N}_i^-} \sum_{p\in\mathscr{P}} \delta_{ji}^p h_p = |w_i|, \quad \forall\, i \in \mathscr{N}^d, \tag{2.9}$$

$$\textstyle\sum_{p\in\mathscr{P}} \delta_{ij}^p h_p \leq u_{ij} y_{ij}, \qquad \forall\, (i,j) \in \mathscr{A}, \tag{2.10}$$

$$h_p \geq 0, \qquad \forall\, p \in \mathscr{P}, \tag{2.11}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall\, (i,j) \in \mathscr{A}, \tag{2.12}$$

where δ_{ij}^p is the constant that indicates whether (i.e., $\delta_{ij}^p = 1$) or not (i.e., $\delta_{ij}^p = 0$) arc $(i, j) \in \mathscr{A}$ belongs to path $p \in \mathscr{P}$ and $e_p = \sum_{(i,j)\in\mathscr{A}} \delta_{ij}^p c_{ij}$, $\forall\, p \in \mathscr{P}$. Note that $x_{ij} = \sum_{p\in\mathscr{P}} \delta_{ij}^p h_p$, $\forall\, (i, j) \in \mathscr{A}$. In addition, it is worth to mention that the number of paths is exponential in the size of the graph, but that any solution expressed in terms of the arc flow variables x_{ij} can be decomposed into a small number of paths (at most $|\mathscr{N}| + |\mathscr{A}|$) in polynomial time.

It is interesting to note that the LP relaxation of both the arc-based and the path-based models reduces to a *minimum cost network flow* problem with transportation costs equal to $c_{ij} + f_{ij}/u_{ij}$ on each arc $(i, j) \in \mathscr{A}$. Indeed, when replacing the integrality constraints (2.5) by $y_{ij} \in [0, 1]$, $\forall\, (i, j) \in \mathscr{A}$, the y_{ij} variables can be projected out, since $f_{ij} \geq 0$ implies that there exists an optimal solution such that the linking constraints (2.3) are satisfied at equality, i.e., $y_{ij} = x_{ij}/u_{ij}$, $\forall\, (i, j) \in \mathscr{A}$. Using these equations to project out the y_{ij} variables, we then obtain the following arc-based minimum cost network flow model:

$$\text{Minimize} \quad \sum_{(i,j)\in\mathscr{A}} \left(c_{ij} + f_{ij}/u_{ij}\right) x_{ij} \tag{2.13}$$

$$\text{Subject to} \quad \textstyle\sum_{j\in\mathscr{N}_i^+} x_{ij} - \sum_{j\in\mathscr{N}_i^-} x_{ji} = w_i, \quad \forall\, i \in \mathscr{N}, \tag{2.14}$$

$$0 \leq x_{ij} \leq u_{ij}, \qquad \forall\, (i,j) \in \mathscr{A}. \tag{2.15}$$

In the remainder of Sect. 2, we discuss three special cases of the *SCFND*: the case where there are no transportation costs, for which we present an equivalent cut-set-based formulation in Sect. 2.1; the case where there are no capacities, the *single-commodity uncapacitated fixed-charge network design* problem (*SUFND*), presented in Sect. 2.2; and the case of a bipartite graph, the *fixed-charge transportation problem* (*FCTP*), described in Sect. 2.3.

2.1 Cut-Set-Based Formulation

A *cut* is a partition of $\mathscr{N}$ into two subsets $\mathscr{S}$ and $\overline{\mathscr{S}} \equiv \mathscr{N} \setminus \mathscr{S}$ such that the net supply across $\mathscr{S}$ is positive, i.e., $W(\mathscr{S}) > 0$ (note that this condition implies that at least one destination is not in $\mathscr{S}$, i.e., $\mathscr{N}^d \cap \overline{\mathscr{S}} \neq \emptyset$). A *cut-set* $(\mathscr{S}, \overline{\mathscr{S}})$ is the subset of arcs induced by the cut, i.e., $(\mathscr{S}, \overline{\mathscr{S}}) = \{(i, j) \in \mathscr{A} : i \in \mathscr{S}, j \in \overline{\mathscr{S}}\}$. The *max-flow-min-cut theorem* guarantees the existence of a feasible solution to the *SCFND if and only if* the (exponentially many) *cut-set-based inequalities* (2.16) are satisfied,

$$\sum_{(i,j)\in(\mathscr{S},\overline{\mathscr{S}})} u_{ij} y_{ij} \geq W(\mathscr{S}),\ \forall \mathscr{S} \subset \mathscr{N},\ W(\mathscr{S}) > 0. \tag{2.16}$$

When we consider the special case of the *SCFND* where $c_{ij} = 0$ for any arc $(i, j) \in \mathscr{A}$, we can project out the x_{ij} variables and obtain an equivalent *cut-set-based formulation*

$$\text{Minimize} \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} \tag{2.17}$$

subject to (2.16) and (2.5).

Obviously, when there are strictly positive transportation costs on some of the arcs, we cannot project out the x_{ij} variables by using only the cut-set-based inequalities. In that case, we would have to exploit LP duality to derive additional valid inequalities affecting the global transportation cost $v = \sum_{(i,j)\in\mathscr{A}} c_{ij} x_{ij}$, as in the celebrated *Benders decomposition* method (see the details of such an approach in Chap. 3).

2.2 The Uncapacitated Variant of the Problem

A particular case of the *SCFND* is obtained when there are no arc capacities on the flows. Since the flow on any arc is bounded by $W(\mathscr{N}^o)$, we can then formulate the problem in the same way as the arc-based formulation of the *SCFND* by replacing u_{ij} by $W(\mathscr{N}^o)$ on every arc $(i, j) \in \mathscr{A}$. The cut-set-based inequalities (2.16) then reduce to the following *connectivity inequalities*:

$$\sum_{(i,j)\in(\mathscr{S},\overline{\mathscr{S}})} y_{ij} \geq 1,\ \forall \mathscr{S} \subset \mathscr{N},\ W(\mathscr{S}) > 0. \tag{2.18}$$

The structure of any feasible solution to the *SUFND* corresponds to a *directed forest* that contains directed trees rooted at each origin. In particular, when there is only one origin and no transportation costs, the problem reduces to the $\mathcal{NP}$-hard *directed Steiner tree* problem, where the terminals correspond to the destinations.

2.3 Fixed-Charge Transportation Problem

The fixed-charge transportation problem (*FCTP*) is the special case of the *SUFND* where the graph is bipartite, i.e., $\mathcal{N} = \mathcal{N}^o \cup \mathcal{N}^d$ and $\mathcal{A} \subseteq \mathcal{N}^o \times \mathcal{N}^d$. Because of the particular structure of the graph, the amount of flow on any arc $(i, j) \in \mathcal{A}$ is bounded by $u_{ij} = \min\{|w_i|, |w_j|\}$. The arc-based formulation of the *FCTP* can then be written as

$$\text{Minimize} \quad \sum_{(i,j)\in\mathcal{A}} \left(f_{ij}y_{ij} + c_{ij}x_{ij}\right) \tag{2.19}$$

$$\text{Subject to} \quad \textstyle\sum_{j\in\mathcal{N}_i^+} x_{ij} = |w_i|, \quad \forall i \in \mathcal{N}^o, \tag{2.20}$$

$$\textstyle\sum_{j\in\mathcal{N}_i^-} x_{ji} = |w_i|, \quad \forall i \in \mathcal{N}^d, \tag{2.21}$$

$$x_{ij} \leq u_{ij}y_{ij}, \qquad \forall (i, j) \in \mathcal{A}, \tag{2.22}$$

$$x_{ij} \geq 0, \qquad \forall (i, j) \in \mathcal{A}, \tag{2.23}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall (i, j) \in \mathcal{A}. \tag{2.24}$$

Since any path $p \in \mathcal{P}$ between an origin $i \in \mathcal{N}^o$ and a destination $j \in \mathcal{N}^d$ corresponds to an arc $(i, j) \in \mathcal{A}$, the arc-based and path-based formulations for the *FCTP* are exactly the same. Although the *FCTP* might appear to be a relatively restrictive special case of the *SCFND*, it is possible, through network flow transformations, to reformulate any *SCFND* on graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$ as an *FCTP*, by associating each arc $(i, j) \in \mathcal{A}$ with an origin of supply u_{ij} and each node $i \in \mathcal{N}$ with a destination of demand $w_i - \sum_{l\in\mathcal{N}_i^+} u_{il}$. In the resulting bipartite graph, we introduce two outgoing arcs associated with each origin $(i, j) \in \mathcal{A}$: one arc incident to destination $i \in \mathcal{N}$ with all costs equal to 0 and one arc incident to destination $j \in \mathcal{N}$ with design cost equal to f_{ij} and transportation cost equal to c_{ij}. Hence, any algorithm to solve the *FCTP* could be used to solve the *SCFND*, although at the expense of a significant increase in the instance size, since the number of nodes and the number of arcs in an *FCTP* reformulation of the *SCFND* are, respectively, $|\mathcal{N}| + |\mathcal{A}|$ and $2 \times |\mathcal{A}|$.

3 Multicommodity Formulations

In this section, we consider network design problems for which several commodities, represented by set $\mathscr{K}$, share the same directed graph $\mathscr{G} = (\mathscr{N}, \mathscr{A})$. The demand to satisfy for any commodity $k \in \mathscr{K}$ is defined at each node $i \in \mathscr{N}$ and is denoted as w_i^k. We discuss below how to represent the demand for each commodity. For now, we simply assume that the demand is balanced for each commodity $k \in \mathscr{K}$, i.e., $\sum_{i\in\mathscr{N}} w_i^k = 0$. The total flow of all commodities on any arc $(i, j) \in \mathscr{A}$ is limited by the capacity $u_{ij} > 0$. We wish to minimize the sum of the costs to select the arcs to be included in the design and the transportation costs to move the flow of all commodities, where for each arc $(i, j) \in \mathscr{A}$, $f_{ij} \geq 0$ is the fixed design cost charged whenever the arc is included in the optimal design, and $c_{ij}^k \geq 0$ is the unit transportation cost for commodity $k \in \mathscr{K}$.

Using binary design variables y_{ij}, $(i, j) \in \mathscr{A}$, as in the single-commodity case, and continuous multicommodity flow variables $x_{ij}^k \geq 0$, $(i, j) \in \mathscr{A}, k \in \mathscr{K}$, the arc-based model for the *multicommodity capacitated fixed-charge network design* problem (*MCFND*) can then be written as

$$\text{Minimize} \quad \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + \sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}} c_{ij}^k x_{ij}^k \tag{2.25}$$

$$\text{Subject to} \quad \textstyle\sum_{j\in\mathscr{N}_i^+} x_{ij}^k - \sum_{j\in\mathscr{N}_i^-} x_{ji}^k = w_i^k, \quad \forall\, i \in \mathscr{N}, \forall\, k \in \mathscr{K}, \tag{2.26}$$

$$\textstyle\sum_{k\in\mathscr{K}} x_{ij}^k \leq u_{ij} y_{ij}, \qquad \forall\,(i, j) \in \mathscr{A}, \tag{2.27}$$

$$x_{ij}^k \geq 0, \qquad \forall\,(i, j) \in \mathscr{A}, \ \forall\, k \in \mathscr{K}, \tag{2.28}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall\,(i, j) \in \mathscr{A}. \tag{2.29}$$

The objective function (2.25) minimizes the total cost computed as the sum of the total fixed cost for arcs included in the optimal design and the total transportation cost for commodities. Constraints (2.26) correspond to flow conservation equations for each node and each commodity. Relations (2.27) represent capacity constraints for each arc. These are also linking constraints, linking together flow and design variables by forbidding any flow to pass through an arc that is not chosen as part of the design. Note that it is easy to include commodity-dependent capacities u_{ij}^k, $(i, j) \in \mathscr{A}, k \in \mathscr{K}$, although we do not consider this case here. One could then simply add the constraints

$$x_{ij}^k \leq u_{ij}^k, \ \forall\,(i, j) \in \mathscr{A}, \ \forall\, k \in \mathscr{K}, \tag{2.30}$$

or, even better,

$$x_{ij}^k \leq u_{ij}^k y_{ij}, \ \forall\,(i, j) \in \mathscr{A}, \ \forall\, k \in \mathscr{K}. \tag{2.31}$$

Below, we comment further on these two different ways of modeling commodity-dependent capacities.

Although this appears to be a special case, we define a commodity as an *origin-destination (OD) pair* $(O(k), D(k))$ with a demand $d^k > 0$ to satisfy between $O(k) \in \mathscr{N}$ and $D(k) \in \mathscr{N}$, in which case we have

$$w_i^k = \begin{cases} d^k, \text{ if } i = O(k), \\ -d^k, \text{ if } i = D(k), \\ 0, \text{ otherwise.} \end{cases}$$

Note that we can always represent a commodity as an OD pair, even when there are several products (physical goods, information or people), each with many origins and many destinations. First, for each product, we introduce a super-origin connected to each origin for the product. Each super-origin has a supply equal to the total supply for the corresponding product. Each arc (i, j) from the super-origin has no costs and a capacity equal to the supply at j. Thus, we obtain a new, larger, graph where each commodity has one origin, but many destinations. At this stage, there are two possible ways to obtain a network in which every commodity corresponds to an OD pair. The first approach is to introduce super-destinations, in the same way as the super-origins, which further increases the size of the graph. The second approach can be used when there are no destination-dependent data (for instance, when there are no commodity-dependent capacities and when transportation costs do not depend on destinations). This approach keeps the size of the graph constant, but increases the number of commodities by simply introducing a commodity for each pair between a super-origin and a destination, with a demand equal to the demand at that destination. Any of these techniques has an implication on the size of the corresponding models and, hence, on the solution methods. We further discuss these issues in several chapters, but for the remainder of the book, unless otherwise stated, we assume that a commodity is defined as an OD pair.

The path-based formulation for the *MCFND* uses this equivalence between a commodity and an OD pair. To derive this model, we introduce $\mathscr{P}^k$, the set of paths between $O(k)$ and $D(k)$ for each $k \in \mathscr{K}$ and $h_p^k \geq 0$, the amount of flow on path $p \in \mathscr{P}^k$, $k \in \mathscr{K}$. The path-based model for the *MCFND* is then written as

$$\text{Minimize} \quad \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + \sum_{k\in\mathscr{K}} \sum_{p\in\mathscr{P}^k} e_p^k h_p^k \tag{2.32}$$

$$\text{Subject to} \quad \textstyle\sum_{p\in\mathscr{P}^k} h_p^k = d^k, \qquad \forall k \in \mathscr{K}, \tag{2.33}$$

$$\textstyle\sum_{k\in\mathscr{K}} \sum_{p\in\mathscr{P}^k} \delta_{ij}^p h_p^k \leq u_{ij} y_{ij}, \quad \forall (i, j) \in \mathscr{A}, \tag{2.34}$$

$$h_p^k \geq 0, \qquad \forall k \in \mathscr{K}, \forall p \in \mathscr{P}^k, \tag{2.35}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall (i, j) \in \mathscr{A}, \tag{2.36}$$

where δ_{ij}^{p} indicates whether (i.e., $\delta_{ij}^{p} = 1$) or not (i.e., $\delta_{ij}^{p} = 0$) arc $(i, j) \in \mathscr{A}$ belongs to path $p \in \cup_{k\in\mathscr{K}} \mathscr{P}^k$ and $e_p^k = \sum_{(i,j)\in\mathscr{A}} \delta_{ij}^{p} c_{ij}^{k}$, $\forall k \in \mathscr{K}$, $\forall p \in \mathscr{P}^k$. Thus, $x_{ij}^{k} = \sum_{p\in\mathscr{P}^k} \delta_{ij}^{p} h_p^k$, $\forall (i, j) \in \mathscr{A}$, $\forall k \in \mathscr{K}$.

An important variant of the *MCFND* is the case with *unsplittable* (or non-bifurcated) flows, where a single path between $O(k)$ and $D(k)$ must be used for each commodity $k \in \mathscr{K}$. We can easily modify both the path-based and the arc-based formulations to model this additional requirement. For the path-based formulation, we define a binary variable H_p^k that assumes value 1, if path $p \in \mathscr{P}^k$ is used for commodity $k \in \mathscr{K}$, and value 0, otherwise. We then have the equations $h_p^k = d^k H_p^k$, $\forall k \in \mathscr{K}, \forall p \in \mathscr{P}^k$, allowing us to project out variables h_p^k, leaving a model that has only binary variables. For the arc-based formulation, we introduce a binary variable X_{ij}^k that takes value 1, if arc $(i, j) \in \mathscr{A}$ belongs to the path between $O(k)$ and $D(k)$ for commodity $k \in \mathscr{K}$, and value 0, otherwise. Similarly, we use the equations $x_{ij}^k = d^k X_{ij}^k$, $\forall (i, j) \in \mathscr{A}, \forall k \in \mathscr{K}$, to project our variables x_{ij}^k and end up with a model that has only binary variables.

For both the arc-based and the path-based formulations, the LP relaxation reduces to a *minimum cost multicommodity network flow* problem with transportation costs per commodity $k \in \mathscr{K}$ equal to $c_{ij}^k + f_{ij}/u_{ij}$ on each arc $(i, j) \in \mathscr{A}$. Indeed, using the same argument as in the single-commodity case, the continuous y_{ij} variables can be projected out, since there is always an optimal solution that satisfies the linking constraints (2.27) at equality, i.e., $y_{ij} = \sum_{k\in\mathscr{K}} x_{ij}^k/u_{ij}$, $\forall (i, j) \in \mathscr{A}$, which yields the following arc-based minimum cost multicommodity network flow model:

$$\text{Minimize} \sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}} \left(c_{ij}^k + f_{ij}/u_{ij}\right) x_{ij}^k \tag{2.37}$$

$$\text{Subject to} \quad \textstyle\sum_{j\in\mathscr{N}_i^+} x_{ij}^k - \sum_{j\in\mathscr{N}_i^-} x_{ji}^k = w_i^k, \quad \forall i \in \mathscr{N}, \forall k \in \mathscr{K}, \tag{2.38}$$

$$\textstyle\sum_{k\in\mathscr{K}} x_{ij}^k \le u_{ij}, \qquad \forall (i, j) \in \mathscr{A}, \tag{2.39}$$

$$x_{ij}^k \ge 0, \qquad \forall (i, j) \in \mathscr{A}, \forall k \in \mathscr{K}. \tag{2.40}$$

The LP relaxation lower bound is generally far from the optimal value, hence the name *weak relaxation* to qualify it. A tighter lower bound is obtained by adding the following valid inequalities to the arc-based model

$$x_{ij}^k \le b_{ij}^k y_{ij}, \ \forall (i, j) \in \mathscr{A}, \ \forall k \in \mathscr{K}, \tag{2.41}$$

where b_{ij}^k is an upper bound on the amount of flow of commodity $k \in \mathscr{K}$ on arc $(i, j) \in \mathscr{A}$, for instance $b_{ij}^k = \min\{u_{ij}, d^k\}$ (or $b_{ij}^k = u_{ij}^k$, if there is a commodity-dependent capacity $u_{ij}^k < \min\{u_{ij}, d^k\}$). These inequalities are called *strong linking constraints* and the corresponding LP relaxation the *strong relaxation*.

It is easy to see that there exists an optimal solution to this LP relaxation such that $y_{ij} = \max\{\max_{k\in\mathscr{K}}\{x_{ij}^k/b_{ij}^k\}, \sum_{k\in\mathscr{K}} x_{ij}^k/u_{ij}\}, \forall\,(i,j) \in \mathscr{A}$. Obviously, projecting out the continuous y_{ij} variables by using these equations would yield a non-linear programming problem. Instead, most solution methods for the strong relaxation leave the y_{ij} variables in the formulation. Because the number of strong linking constraints is typically large for reasonably sized instances, solving the strong relaxation can be quite challenging. Chapter 3 reviews several methods to solve the strong relaxation.

In Sect. 3.1, we study the special case where there are no capacities, the *multicommodity uncapacitated fixed-charge network design* problem (*MUFND*). Then, Sect. 3.2 presents the generalization of the cut-set-based inequalities for multicommodity fixed-charge network design problems.

3.1 The Uncapacitated Variant of the Problem

When there are no arc capacities on the multicommodity flows, we can formulate the resulting *MUFND* by replacing u_{ij} with $\sum_{k\in\mathscr{K}} d^k$ on every arc $(i,j) \in \mathscr{A}$. In addition, the strong linking constraints (2.41) can then be simplified by replacing b_{ij}^k with d^k, for any arc $(i,j) \in \mathscr{A}$ and any commodity $k \in \mathscr{K}$. It is then trivial to observe that the linking constraints (2.27) are implied by the strong linking constraints (2.41), since (2.27) are obtained by aggregating (2.41) over $\mathscr{K}$.

This discussion illustrates the choice between two formulations: the *aggregated model*, which does not include the strong linking constraints (2.41), and the *disaggregated model*, where the linking constraints (2.27) are replaced by the strong ones. Our discussion above on the quality of the LP relaxation bounds for the *MCFND* also applies to the *MUFND*: the aggregated model yields a weak LP relaxation bound, while the LP relaxation of the disaggregated formulation is significantly stronger.

To summarize, the disaggregated model for the *MUFND* can be stated as

$$\text{Minimize} \quad \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + \sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}} c_{ij}^k x_{ij}^k \tag{2.42}$$

$$\text{Subject to} \quad \sum_{j\in\mathscr{N}_i^+} x_{ij}^k - \sum_{j\in\mathscr{N}_i^-} x_{ji}^k = w_i^k, \quad \forall\, i \in \mathscr{N}, \forall\, k \in \mathscr{K}, \tag{2.43}$$

$$x_{ij}^k \le d^k y_{ij}, \qquad \forall\,(i,j) \in \mathscr{A}, \forall\, k \in \mathscr{K}, \tag{2.44}$$

$$x_{ij}^k \ge 0, \qquad \forall\,(i,j) \in \mathscr{A}, \forall\, k \in \mathscr{K}, \tag{2.45}$$

$$y_{ij} \in \{0,1\}, \qquad \forall\,(i,j) \in \mathscr{A}. \tag{2.46}$$

It is worth noting that, once the y_{ij} variables are fixed, the resulting subproblem decomposes into $|\mathscr{K}|$ shortest path subproblems, i.e., it consists in finding the shortest path between $O(k)$ and $D(k)$ for each commodity $k \in \mathscr{K}$, which is easy since the transportation costs are nonnegative. As a consequence, there exists an optimal solution to the *MUFND* where a single path is used between $O(k)$ and $D(k)$ for each commodity $k \in \mathcal{K}$, i.e., the flows are unsplittable. By contrast, for the *MCFND*, the subproblem obtained after fixing the y_{ij} variables is a minimum cost multicommodity network flow problem, which is significantly more difficult, even though it is a linear program. In particular, there are *MCNFD* instances for which all optimal solutions incur splittable flows, i.e., several paths are used to satisfy the demand d^k for some commodity $k \in \mathcal{K}$.

3.2 Cut-Set-Based Inequalities

It is easy to generalize cut-set-based inequalities for the *MCFND*. Using the same definitions and notations as in Sect. 2.1, we derive the following valid inequalities from Eqs. (2.26) and capacity constraints (2.27):

$$\sum_{(i,j)\in(\mathcal{S},\overline{\mathcal{S}})} u_{ij} y_{ij} \geq \sum_{k\in\mathcal{K}(\mathcal{S},\overline{\mathcal{S}})} d^k, \ \forall \mathcal{S} \subset \mathcal{N},\ \mathcal{S} \neq \emptyset, \tag{2.47}$$

where $\mathcal{K}(\mathcal{S},\overline{\mathcal{S}}) = \{k \in \mathcal{K} : O(k) \in \mathcal{S},\ D(k) \in \overline{\mathcal{S}}\}$. By combining flow conservation equations (2.26) with strong linking constraints (2.41), we can derive the following commodity-dependent cut-set-based inequalities

$$\sum_{(i,j)\in(\mathcal{S},\overline{\mathcal{S}})} b^k_{ij} y_{ij} \geq d^k,\ \forall \mathcal{S} \subset \mathcal{N},\ \forall k \in \mathcal{K},\ O(k) \in \mathcal{S},\ D(k) \in \overline{\mathcal{S}}. \tag{2.48}$$

Contrary to the *SCFND* for which (2.16) are not only necessary (valid), but also sufficient to characterize any feasible solution, cut-set-based inequalities (2.47) and (2.48) do not characterize feasible solutions to the *MCFND*. Consequently, even if there are no transportation costs, we cannot obtain a reformulation in the space of the y_{ij} variables by using only the cut-set-based inequalities (2.47) and (2.48). To derive such a reformulation, we would have to use Benders feasibility cuts, a topic covered in Chap. 3.

For the *MUFND*, inequalities (2.48) reduce to the following *connectivity inequalities*, which imply cut-set-based inequalities (2.47):

$$\sum_{(i,j)\in(\mathcal{S},\overline{\mathcal{S}})} y_{ij} \geq 1,\ \forall \mathcal{S} \subset \mathcal{N},\ \forall k \in \mathcal{K},\ O(k) \in \mathcal{S},\ D(k) \in \overline{\mathcal{S}}. \tag{2.49}$$

For the *MUFND* with no transportation costs, i.e., $c^k_{ij} = 0$ for any arc $(i, j) \in \mathcal{A}$ and any commodity $k \in \mathcal{K}$, these connectivity inequalities define a reformulation of the problem obtained by projecting out the x^k_{ij} variables, the *cut-set-based formulation*:

$$\text{Minimize} \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} \tag{2.50}$$

subject to (2.46) and (2.49). This observation illustrates another significant difference between the *MCFND* and the *MUFND*: while cut-set-based inequalities (2.49) characterize feasible solutions to the *MUFND*, feasible solutions to the *MCFND* cannot be completely characterized by cut-set-based inequalities (2.47) and (2.48).

4 Bibliographical Notes

The study of fixed-charge problems (with an underlying general LP structure) originated from the work of Hirsch and Dantzig (1968), first published as a technical report in 1954. To the best of our knowledge, the first article that presents a special case of single-commodity fixed-charge network design problem is Balinski (1961), where the *FCTP* is introduced. The arc-based formulation of the problem is described and it is shown that the LP relaxation reduces to a linear transportation problem with costs $c_{ij} + f_{ij}/u_{ij}$ on each arc $(i, j) \in \mathscr{A}$. A similar result applies to the *SCFND*, as we have seen in Sect. 2.

In 1961, also appeared one of the first studies on multicommodity network design, due to Gomory and Hu (1961). In the problem considered in that paper, there are several commodities, each commodity $k \in \mathscr{K}$ having a demand d^k to be routed between the origin $O(k)$ and the destination $D(k)$, and we wish to decide how many units of capacity y_{ij} to install on each edge of an undirected network, so as to minimize the cost of installing the capacities, where each unit of capacity installed on edge (i, j) incurs a cost f_{ij}. This problem can be modeled with general integer variables y_{ij}, rather than binary variables, and cut-set-based inequalities can be used to represent the set of feasible solutions. Generalizations of this problem are studied in Chaps. 5 and 6.

Following the seminal papers of Balinski (1961) and Gomory and Hu (1961), the research on network design has been fruitful during the next 25 years. The early efforts of the research community are synthesized in the survey papers of Magnanti and Wong (1984) and Minoux (1989). The first paper formalizes the *MUFND*, showing that it generalizes well-known problems, such as the shortest path problem, the traveling salesman problem and the Steiner tree problem. The second paper emphasizes the importance of piecewise linear costs in network design applications (see Chap. 6). Both papers include significant reviews on algorithms for the *multicommodity uncapacitated budget network design problem (MUBND)*.

Most papers on single-commodity network design have focused on the *FCTP*, which is not surprising, given that the *SCFND* can be reformulated as an *FCTP* (Malek-Zavarei and Frisch 1972). We review the papers on the *FCTP* in Chaps. 3 and 4. Notable exceptions are the works on the directed Steiner tree problem and its generalization, the *SUFND*. On the first problem, we mention the seminal work of Wong (1984), which proposes both a multicommodity flow model (each destination is identified as a commodity) and a cut-set-based formulation, showing that the two LP relaxations are equivalent. Similar results are obtained for other network design problems defined over trees (Magnanti and Wolsey 1995). On the *SUFND* with a single origin, it is worth mentioning the work of Rardin and Wolsey (1993), showing that a multicommodity reformulation of the problem, where each destination corresponds to a commodity, has the same LP relaxation as the one obtained from the single-commodity model by adding so-called dicut collection inequalities, which involve both the design variables and the single-commodity flow variables.

Following the early research on the *MUBND*, subsequent works on the *MUFND* can be found in Balakrishan (1987) on the path-based model and in Balakrishnan et al. (1989) on the arc-based model. On the *MCFND* and its generalization where each commodity has several origins and several destinations, early research (Rardin and Choe 1979; Gendron and Crainic 1994) has focused on disaggregated formulations and the impact of strong linking inequalities on the quality of the LP relaxation. Balakrishnan et al. (1997) presents an annotated bibliography that contains most references on network design that appeared since 1961. In Chaps. 3 and 4, we review many other references on the *MUFND* and on the *MCFND*.

5 Conclusions and Perspectives

This introductory chapter has presented basic fixed-charge network design problems and formulations. It has allowed us to identify fundamental distinctions between problems and models, which we further explore in the next chapters. In particular, we have seen significant differences between single-commodity and multicommodity formulations. Indeed, when the design variables are fixed or when their integrality is relaxed, the *SCFND* has an underlying single-commodity minimum cost network flow problem, for which many efficient specialized algorithms exist, while the *MCFND* displays a multicommodity minimum cost network flow problem, which is significantly more difficult, although it is a linear program. Also, when there are no transportation costs, the *SCFND* can be formulated with cut-set-based inequalities, by virtue of the max-flow-min-cut theorem, while cut-set-based inequalities are necessary, but not sufficient, to model the *MCFND* with no transportation costs. We have also emphasized important differences between uncapacitated and capacitated problems. For both the *SUFND* and the *MUFND*, connectivity inequalities are necessary and sufficient to characterize feasible solutions. Clearly, these inequalities are proper subsets of the cut-set-based inequalities for the *SCFND* and the *MCFND*, and therefore necessary, but not sufficient, to describe feasible solutions.

While the nature of the demand, single-commodity versus multicommodity, is a fundamental distinction between network design models, it is worth noting that single-commodity and multicommodity formulations can often be used for the same problem. For instance, we could reformulate the *SCFND* with a single origin and multiple destinations as an *MCFND*, where each commodity is associated with a destination. The advantage of this reformulation comes when we introduce strong linking inequalities, which then improve the LP relaxation, often significantly so. The disadvantage, of course, lies in the large number of additional flow variables and valid inequalities. Two approaches can be used to mitigate this effect. One is to develop decomposition methods, such as column-and-row generation algorithms, which we study in Chap. 3. Another is to derive inequalities equivalent to the strong linking constraints, but in the space of single-commodity flow (and design) variables. Some inequalities of this type have been derived for the *SUFND* with a single origin, but to the best of our knowledge, not for its capacitated counterpart. This is an avenue for future research.

References

Balakrishnan, A. (1987). LP extreme points and cuts for the fixed charge network design. *Mathematical Programming, 39*, 263–284.

Balakrishnan, A., Magnanti, T. L., & Mirchandani, P. (1997). Network design. In M. Dell'Amico, F. Maffioli, & S. Martello (Eds.), *Annotated bibliographies in combinatorial optimization* (pp. 311–334). New York: Wiley.

Balakrishnan, A., Magnanti, T. L., & Wong, R. (1989). A dual-ascent procedure for large-scale uncapacitated network design. *Operations Research, 37*(5), 716–740.

Balinski, M. L. (1961). Fixed-cost transportation problems. *Naval Research Logistics, 8*(1), 41–54.

Gendron, B., & Crainic, T. G. (1994). *Relaxations for Multicommodity Capacitated Network Design Problems*. Publication CRT-965, Centre for Research on Transportation, University of Montreal.

Gomory, R. E., & Hu, T. C. (1961). Multiterminal network flows. *SIAM Journal of Applied Mathematics, 9*, 551–570.

Hirsch, W. M., & Dantzig, G. B. (1968). The fixed charge problem. *Naval Research Logistics, 15*(3), 413–424.

Magnanti, T. L., & Wolsey, L. A. (1995). Optimal trees. In M. Ball, T. L. Magnanti, C. L. Monma, & G. L. Nemhauser (Eds.), *Network models*. Handbooks in Operations Research and Management Science (vol. 7, pp. 503–615). Amsterdam: North-Holland.

Magnanti, T. L., & Wong, R. T. (1984). Network design and transportation planning: Models and algorithms. *Transportation Science, 18*(1), 1–55.

Malek-Zavarei, M., & Frisch, I. T. (1972). On the fixed cost flow problem. *International Journal of Control, 16*(5), 897–902.

Minoux, M. (1989). Network synthesis and optimum network design problems: models, solution methods and applications. *Networks, 19*, 313–360.

Rardin, R. L., & Choe, U. (1979). *Tighter Relaxations of Fixed Charge Network Flow Problems*. Report J-79-18, Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta.

Rardin, R. L., & Wolsey, L. A. (1993). Valid Inequalities and Projecting the multicommodity extended formulation for uncapacitated fixed charge network flow problems. *European Journal of Operational Research, 71*, 95–109.

Wong, R. T. (1984). A dual ascent approach for steiner tree problems in a directed graph. *Mathematical Programming, 28*, 217–287.

Chapter 3
Exact Methods for Fixed-Charge Network Design

Teodor Gabriel Crainic and Bernard Gendron

1 Introduction

This chapter is dedicated to exact algorithms for the fixed-charge network design problems introduced in Chap. 2. These problems are notoriously difficult to solve and many factors contribute to their complexity. First and foremost, most network design problems presented in Chap. 2 are strongly $\mathscr{NP}$-hard. This is the case for both the *SCFND* and the *MCFND*, i.e., the single-commodity and multicommodity capacitated fixed-charge network design problems. In practice, this result implies that no polynomial (even, pseudo-polynomial) algorithms are known to solve these problems (and it is unlikely that any polynomial algorithm can be found to solve them, unless $\mathscr{P} = \mathscr{NP}$). Another factor that makes these problems so difficult to solve is the objective function that comprises fixed design costs and variable transportation costs. This results in complex tradeoffs between fixed and variable costs, which complicates the task of finding optimal solutions. In addition, simple linear approximations of such functions are typically weak, in particular when fixed costs are significant compared to variable costs.

Among fixed-charge network design problems, the developments in Chap. 2 clearly identify multicommodity capacitated problems as the most challenging. These problems are difficult for the reasons stated above, but also because their underlying multicommodity capacitated network flow subproblems are compu-

T. G. Crainic
CIRRELT and AOTI, Université du Québec à Montréal, Montréal, QC, Canada
e-mail: TeodorGabriel.Crainic@cirrelt.net

B. Gendron (✉)
CIRRELT and Département d'informatique et de recherche opérationnelle,
Université de Montréal, Montréal, QC, Canada
e-mail: Bernard.Gendron@cirrelt.net

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_3

tationally elusive, even if they can be solved as linear programs. In particular, these models are often highly degenerate. Models for the *MCFND* also contain a large number of variables and constraints. Indeed, the arc-based model for the *MCFND* has $O(|\mathscr{A}|)$ binary design variables and $O(|\mathscr{A}| \times |\mathscr{K}|)$ continuous flow variables. Chapter 2 has emphasized the importance of adding strong linking inequalities between these two types of variables to compute better LP relaxations. The drawback is to significantly increase the number of constraints from $O((|\mathscr{N}| \times |\mathscr{K}|) + |\mathscr{A}|)$ to $O(|\mathscr{A}| \times |\mathscr{K}|)$. To handle such large-scale models, decomposition methods that exploit subproblem structures are often useful.

The chapter is divided into three parts. Part I presents relaxations that can improve the quality of the lower bounds, while exploiting the subproblem structures. This part contains three sections: Sect. 2 on Lagrangian relaxations and Dantzig–Wolfe reformulations; Sect. 3 on relaxations by projection and Benders reformulations; Sect. 4 on valid inequalities. Part II focuses on enumeration algorithms that use the modeling techniques covered in Part I. This part includes four sections: Sect. 5 on branch-and-bound algorithms; Sect. 6 on branch-and-cut algorithms; Sect. 7 on Benders decomposition; Sect. 8 on branch-and-price algorithms. Part III is dedicated to the solution of large-scale instances by combining the advanced enumeration algorithms of Part II with heuristic methods, a topic covered in Sect. 9, and by exploiting parallel computing, as explained in Sect. 10.

Part I: Relaxations

2 Lagrangian Relaxations and Dantzig–Wolfe Reformulations

We study two classical Lagrangian relaxations for network design models, one that involves relaxing the linking constraints, the other being obtained by relaxing the flow conservation equations. Alternative Lagrangian relaxations are also studied. We focus mostly on the general single-commodity and multicommodity capacitated fixed-charge models, i.e., the *SCFND* and the *MCFND*. First, we give a primer on Lagrangian relaxation, introducing the necessary notions to understand the remainder of this section.

2.1 A Primer on Lagrangian Relaxation

We are given a MILP model of the form $z = \min\{cx \mid Ax \geq b, x \in X\}$, where X is a bounded feasible set defined by linear constraints and integrality requirements on some (or all) of the variables. The Lagrangian relaxation of constraints $Ax \geq b$ consists in relaxing them and in penalizing their violations in the objective with Lagrange multipliers $\gamma \geq 0$. We then obtain the Lagrangian subproblem $v(\gamma) = \min\{c - \gamma A)x \mid x \in X\} = \min\{c - \gamma A)x \mid x \in conv(X)\}$, where $conv(X)$ is the convex hull of X. The Lagrangian subproblem is said to have the *integrality property* if $conv(X) = \overline{X}$, where $\overline{X}$ is obtained from X by relaxing the integrality requirements (in other words, an integer optimal solution to the Lagrangian subproblem can be obtained even if the integrality constraints are relaxed). It is easy to see that $v(\gamma) + \gamma b \leq z$, with optimal Lagrange multipliers obtained by solving the Lagrangian dual: $v = \max_{\gamma \geq 0}\{v(\gamma) + \gamma b\}$. The *primal interpretation of Lagrangian duality* states that $v = \min\{cx \mid Ax \geq b, x \in conv(X)\}$. Since $conv(X) \subseteq \overline{X}$, this implies that $v \geq \overline{z}$, the LP relaxation lower bound, with $v = \overline{z}$ if $conv(X) = \overline{X}$.

We now assume that X is decomposable into $m = |\mathscr{L}|$ bounded MILP feasible sets: $x = (x_1, \ldots, x_m) \in X_1 \times \cdots \times X_m = X$. Since any point $x_l \in conv(X_l)$, $l \in \mathscr{L}$, can be written as a convex combination of the extreme points $(\xi_l^q)_{q \in \mathscr{Q}_l}$ of $conv(X_l)$, we can reformulate the Lagrangian dual as an LP with the so-called *Dantzig–Wolfe reformulation*, where the variable λ_l^q, $l \in \mathscr{L}$, $q \in \mathscr{Q}_l$, represents the weight given to extreme point ξ_l^q:

$$v = \text{Minimize} \sum_{l \in \mathscr{L}} \sum_{q \in \mathscr{Q}_l} c\lambda_l^q \xi_l^q \tag{3.1}$$

$$\text{Subject to} \quad \textstyle\sum_{l \in \mathscr{L}} \sum_{q \in \mathscr{Q}_l} A\lambda_l^q \xi_l^q \geq b, \tag{3.2}$$

$$\textstyle\sum_{q \in \mathscr{Q}_l} \lambda_l^q = 1, \qquad \forall\, l \in \mathscr{L}, \tag{3.3}$$

$$\lambda_l^q \geq 0, \qquad \forall\, l \in \mathscr{L}, \forall\, q \in \mathscr{Q}_l. \tag{3.4}$$

2.2 Relaxing Linking Constraints

First, we consider the Lagrangian relaxation of the linking constraints for the *SCFND*. After adding the redundant bounding constraints on the flow variables

$$x_{ij} \leq u_{ij}, \ \forall\, (i, j) \in \mathscr{A}, \tag{3.5}$$

we relax the linking constraints and penalize their violations in the objective function with Lagrange multipliers $\alpha_{ij} \geq 0$, $(i, j) \in \mathscr{A}$. We thus obtain the following Lagrangian subproblem:

$$\text{Minimize} \sum_{(i,j) \in \mathscr{A}} \left\{(f_{ij} - \alpha_{ij} u_{ij}) y_{ij} + (c_{ij} + \alpha_{ij}) x_{ij}\right\} \tag{3.6}$$

$$\text{Subject to} \quad \textstyle\sum_{j \in \mathscr{N}_i^+} x_{ij} - \sum_{j \in \mathscr{N}_i^-} x_{ji} = w_i, \ \forall\, i \in \mathscr{N}, \tag{3.7}$$

$$0 \leq x_{ij} \leq u_{ij}, \qquad \forall\, (i, j) \in \mathscr{A}, \tag{3.8}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall\, (i, j) \in \mathscr{A}. \tag{3.9}$$

This Lagrangian subproblem decomposes into two parts: a subproblem in flow variables that reduces to a single-commodity minimum cost network flow problem and a subproblem in design variables that is solvable by inspection, i.e., $y_{ij} = 1$ if $f_{ij} - \alpha_{ij} u_{ij} < 0$, and 0, otherwise, $(i, j) \in \mathscr{A}$. It is clear that the Lagrangian subproblem would be solved in the same way if the design variables were continuous, i.e., $y_{ij} \in [0, 1]$, $(i, j) \in \mathscr{A}$. The Lagrangian subproblem thus has the integrality property, which implies that the Lagrangian dual provides the same lower bound as the LP relaxation. It follows that optimal Lagrange multipliers are given by $\alpha_{ij} = f_{ij}/u_{ij}$, $(i, j) \in \mathscr{A}$, since the corresponding Lagrangian subproblem reduces to the LP relaxation (see Chap. 2, Sect. 2).

Now, we turn our attention to the Lagrangian relaxation of the linking constraints for the *MCFND*. To obtain a Lagrangian bound that is at least as good as the strong relaxation (see Chap. 2, Sect. 3), we consider the Lagrangian relaxation of the capacity constraints and the strong linking constraints:

$$\textstyle\sum_{k \in \mathscr{K}} x_{ij}^k \leq u_{ij} y_{ij}, \ \forall\, (i, j) \in \mathscr{A}, \tag{3.10}$$

$$x_{ij}^k \leq b_{ij}^k y_{ij}, \qquad \forall\, (i, j) \in \mathscr{A},\ \forall\, k \in \mathscr{K}. \tag{3.11}$$

If we denote by $\alpha_{ij} \geq 0$, $(i, j) \in \mathscr{A}$, and $\beta_{ij}^k \geq 0$, $(i, j) \in A$, $k \in \mathscr{K}$, the Lagrange multipliers associated with constraints (3.10) and (3.11), respectively, we obtain the following Lagrangian subproblem:

$$\text{Minimize} \quad \sum_{(i,j) \in \mathscr{A}} \left\{ \left(f_{ij} - \alpha_{ij} u_{ij} - \sum_{k \in \mathscr{K}} \beta_{ij}^k b_{ij}^k \right) y_{ij} + \sum_{k \in \mathscr{K}} (c_{ij}^k + \alpha_{ij} + \beta_{ij}^k) x_{ij}^k \right\} \tag{3.12}$$

$$\text{Subject to} \quad \textstyle\sum_{j \in \mathscr{N}_i^+} x_{ij}^k - \sum_{j \in \mathscr{N}_i^-} x_{ji}^k = w_i^k, \ \ \forall\, i \in \mathscr{N}, \forall\, k \in \mathscr{K}, \tag{3.13}$$

$$x_{ij}^k \geq 0, \qquad \forall\, (i, j) \in \mathscr{A},\ \forall\, k \in \mathscr{K}, \tag{3.14}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall\, (i, j) \in \mathscr{A}. \tag{3.15}$$

This Lagrangian subproblem decomposes into two parts: a subproblem in flow variables and a subproblem in design variables that is solvable by inspection, i.e., $y_{ij} = 1$ if $f_{ij} - \alpha_{ij} u_{ij} - \sum_{k \in \mathscr{K}} \beta_{ij}^k b_{ij}^k < 0$, and 0, otherwise, $(i, j) \in \mathscr{A}$. In exactly the same way as for the *SCFND*, the Lagrangian subproblem has the integrality property, which implies that the Lagrangian dual gives the same lower bound as the strong relaxation. Note that the subproblem in flow variables reduces to the computation of a shortest path (with respect to arc lengths $c_{ij}^k + \alpha_{ij} + \beta_{ij}^k$) between

$O(k)$ and $D(k)$ for each commodity $k \in \mathscr{K}$. Hence, this Lagrangian relaxation is called the *shortest path relaxation*.

Using the particular structure of the Lagrangian subproblem that is decomposable into $|\mathscr{K}|$ shortest path problems and $|\mathscr{A}|$ problems solvable by inspection, one can model the Lagrangian dual with the following Dantzig–Wolfe reformulation (the notation is introduced in Chap. 2, Sect. 3):

$$\text{Minimize} \quad \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + \sum_{(i,j)\in\mathscr{A}} \sum_{k\in\mathscr{K}} \sum_{p\in\mathscr{P}^k} d^k c_{ij}^k \delta_{ij}^p h_p^k \tag{3.16}$$

$$\text{Subject to} \quad \textstyle\sum_{p\in\mathscr{P}^k} h_p^k = 1, \quad \forall\, k \in \mathscr{K}, \tag{3.17}$$

$$\textstyle\sum_{k\in\mathscr{K}} \sum_{p\in\mathscr{P}^k} d^k \delta_{ij}^p h_p^k \le u_{ij} y_{ij}, \quad \forall\, (i,j) \in \mathscr{A}, \tag{3.18}$$

$$\textstyle\sum_{p\in\mathscr{P}^k} d^k \delta_{ij}^p h_p^k \le b_{ij}^k y_{ij}, \quad \forall\, (i,j) \in \mathscr{A}, \forall\, k \in \mathscr{K}, \tag{3.19}$$

$$h_p^k \ge 0, \quad \forall\, k \in \mathscr{K}, \forall\, p \in \mathscr{P}^k, \tag{3.20}$$

$$y_{ij} \in [0,1], \quad \forall\, (i,j) \in \mathscr{A}. \tag{3.21}$$

By imposing integrality on the design variables, we obtain a reformulation of the *MCFND*. This reformulation corresponds to the path-based model introduced in Chap. 2, Sect. 3, with two modifications: (1) for each commodity $k \in \mathscr{K}$, the path-based flow variable h_p^k now provides the fraction of the demand d^k that is satisfied through path $p \in \mathscr{P}^k$ instead of the actual flow; (2) the model is reinforced by adding the strong linking constraints (3.19). The LP relaxation of this so-called *path-based reformulation* of the *MCFND* solves the Lagrangian dual, providing the same lower bound as the strong relaxation.

2.3 Relaxing Flow Conservation Constraints

We study the subproblems that arise when the flow conservation constraints are relaxed in a Lagrangian way. We consider first the case of the *SCFND*.

After relaxing the flow conservation equations, penalizing their violations in the objective function with Lagrange multipliers π_i, $i \in \mathscr{N}$, we obtain the following Lagrangian subproblem:

$$\text{Minimize} \quad \sum_{(i,j)\in\mathscr{A}} \left\{ f_{ij} y_{ij} + (c_{ij} + \pi_i - \pi_j) x_{ij} \right\} - \sum_{i\in\mathscr{N}} \pi_i w_i \tag{3.22}$$

$$\text{Subject to} \quad 0 \le x_{ij} \le u_{ij} y_{ij}, \quad \forall\, (i,j) \in \mathscr{A}, \tag{3.23}$$

$$y_{ij} \in \{0,1\}, \quad \forall\, (i,j) \in \mathscr{A}. \tag{3.24}$$

This subproblem decomposes by arc: for each $(i, j) \in \mathscr{A}$, if $f_{ij} + (c_{ij} + \pi_i - \pi_j)u_{ij} < 0$, then the optimal solution is $y_{ij} = 1$ and $x_{ij} = u_{ij}$; otherwise, an optimal solution is $y_{ij} = x_{ij} = 0$. It is clear that this Lagrangian subproblem has the integrality property, which implies that the Lagrangian dual provides the same lower bound as the LP relaxation. It follows that optimal Lagrange multipliers π_i, $i \in \mathscr{N}$, can be obtained by solving the LP relaxation as a minimum cost network flow problem with transportation costs equal to $c_{ij} + f_{ij}/u_{ij}$ on each arc $(i, i) \in \mathscr{A}$ (see Chap. 2, Sect. 2), where π_i corresponds to the dual variable associated with the flow conservation equation for node $i \in \mathscr{N}$.

For the *MCFND*, we consider the formulation obtained after adding the strong linking inequalities (3.11) in order for the Lagrangian dual to provide a lower bound that is at least as good as that of the strong relaxation. After relaxing the flow conservation equations in a Lagrangian way, using Lagrange multipliers π_i^k, $i \in \mathscr{N}$, $k \in \mathscr{K}$, we obtain the following subproblem:

$$\text{Minimize} \quad \sum_{(i,j)\in\mathscr{A}} f_{ij}y_{ij} + \sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}} (c_{ij}^k + \pi_i^k - \pi_j^k)x_{ij}^k + \sum_{k\in\mathscr{K}} d^k(\pi_{D(k)}^k - \pi_{O(k)}^k) \tag{3.25}$$

$$\text{Subject to} \quad \textstyle\sum_{k\in\mathscr{K}} x_{ij}^k \le u_{ij}y_{ij}, \ \forall (i, j) \in \mathscr{A}, \tag{3.26}$$

$$x_{ij}^k \le b_{ij}^k y_{ij}, \qquad \forall (i, j) \in \mathscr{A}, \ \forall k \in \mathscr{K}, \tag{3.27}$$

$$x_{ij}^k \ge 0, \qquad \forall (i, j) \in \mathscr{A}, \ \forall k \in \mathscr{K}, \tag{3.28}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall (i, j) \in \mathscr{A}. \tag{3.29}$$

This Lagrangian subproblem decomposes by arc. For each arc (i, j), a continuous knapsack problem (with costs $c_{ij}^k + \pi_i^k - \pi_j^k$, $k \in \mathscr{K}$) is obtained for each value $y_{ij} \in \{0, 1\}$. If $y_{ij} = 0$, all variables assume value 0. If $y_{ij} = 1$, the solution $(\widetilde{x}_{ij}^k)_{k\in\mathscr{K}}$ to the continuous knapsack problem is compared to the solution where all variables take value 0: if $f_{ij} + \sum_{k\in\mathscr{K}}(c_{ij}^k + \pi_i^k - \pi_j^k)\widetilde{x}_{ij}^k < 0$, then $y_{ij} = 1$ and $x_{ij}^k = \widetilde{x}_{ij}^k$, $k \in \mathscr{K}$, is the optimal solution; otherwise, the solution with all variables equal to 0 is optimal. Because of its particular structure, this Lagrangian relaxation is called the *knapsack relaxation*.

We now show that this Lagrangian subproblem has the integrality property. Consider the LP relaxation of the Lagrangian subproblem, which also decomposes by arc. For each $(i, j) \in \mathscr{A}$, given $\widehat{y}_{ij} \in [0, 1]$, an optimal solution in x variables is obtained by solving the following continuous knapsack problem: $w_{ij}(\widehat{y}_{ij}) = \min\{\sum_{k\in\mathscr{K}}(c_{ij}^k + \pi_i^k - \pi_j^k)x_{ij}^k \mid \sum_{k\in\mathscr{K}} x_{ij}^k \le u_{ij}\widehat{y}_{ij};\ 0 \le x_{ij}^k \le b_{ij}^k\widehat{y}_{ij}, k \in \mathscr{K}\}$. It is easy to see that an optimal solution to this continuous knapsack problem is either 0, if $f_{ij}\widehat{y}_{ij} + w_{ij}(\widehat{y}_{ij}) \ge 0$, or $\widehat{y}_{ij}\widetilde{x}_{ij}^k$, $k \in \mathscr{K}$, otherwise, where $(\widetilde{x}_{ij}^k)_{k\in\mathscr{K}}$ is an optimal solution to the continuous knapsack problem when $\widehat{y}_{ij} = 1$. The minimum value of the LP relaxation of the Lagrangian subproblem for arc $(i, j) \in \mathscr{A}$ is therefore obtained either for $\widehat{y}_{ij} = 0$, or for $\widehat{y}_{ij} = 1$, since $f_{ij} + w_{ij}(1) \le f_{ij}\widehat{y}_{ij} + w_{ij}(\widehat{y}_{ij})$, if

$f_{ij}\widehat{y}_{ij}+w_{ij}(\widehat{y}_{ij}) < 0$. Thus, the Lagrangian subproblem has the integrality property, which implies that the Lagrangian dual gives the same lower bound as the strong relaxation.

To derive a Dantzig–Wolfe reformulation of the Lagrangian dual for the knapsack relaxation, we define for each arc $(i, j) \in \mathscr{A}$, the set $\{(\xi_{ij}^{kq})_{k\in\mathscr{K},q\in\mathscr{Q}_{ij}}\}$ of non-trivial extreme points (i.e., excluding 0) of the knapsack polyhedron $\{(x_{ij}^k)_{k\in\mathscr{K}} \mid \sum_{k\in\mathscr{K}} x_{ij}^k \leq u_{ij}; 0 \leq x_{ij}^k \leq b_{ij}^k, k \in \mathscr{K}\}$. Since the Lagrangian subproblem is decomposable into one subproblem for each $(i, j) \in \mathscr{A}$ and each value $y_{ij} \in \{0, 1\}$, the Lagrangian dual for the knapsack relaxation can be formulated as follows:

$$\text{Minimize} \sum_{(i,j)\in\mathscr{A}} f_{ij}y_{ij} + \sum_{(i,j)\in\mathscr{A}}\sum_{q\in\mathscr{Q}_{ij}}\sum_{k\in\mathscr{K}} c_{ij}^k\xi_{ij}^{kq}\lambda_{ij}^q \tag{3.30}$$

Subject to

$$\sum_{j\in\mathscr{N}_i^+}\sum_{q\in\mathscr{Q}_{ij}} \xi_{ij}^{kq}\lambda_{ij}^q - \sum_{j\in\mathscr{N}_i^-}\sum_{q\in\mathscr{Q}_{ji}} \xi_{ij}^{kq}\lambda_{ji}^q = w_i^k, \ \forall i \in \mathscr{N}, \forall k \in \mathscr{K}, \tag{3.31}$$

$$\textstyle\sum_{q\in\mathscr{Q}_{ij}} \lambda_{ij}^q = y_{ij}, \quad \forall (i, j) \in \mathscr{A}, \tag{3.32}$$

$$\lambda_{ij}^q \geq 0, \qquad \forall (i, j) \in \mathscr{A}, \forall q \in \mathscr{Q}_{ij}, \tag{3.33}$$

$$y_{ij} \in [0, 1], \qquad \forall (i, j) \in \mathscr{A}, \tag{3.34}$$

where λ_{ij}^q, $(i, j) \in \mathscr{A}, q \in \mathscr{Q}_{ij}$, is the convex combination weight assigned to the extreme point $(\xi_{ij}^{kq})_{k\in\mathscr{K}}$ in an optimal solution to the Lagrangian dual. By imposing integrality on the design variables, we obtain a reformulation of the *MCFND*. The LP relaxation of this so-called *knapsack-based reformulation* of the *MCFND* provides the same lower bound as the strong relaxation, given that the Lagrangian subproblem has the integrality property.

2.4 Other Lagrangian Relaxations

As just seen above, the two alternatives, relaxing the linking constrains or the flow conservation equations, yield Lagrangian duals that provide the same lower bound as the LP relaxation, and this is the case for both the *SCFND* and the *MCFND*. In this section, we examine other Lagrangian relaxations that can give tighter lower bounds than the LP relaxation. We show two techniques to derive such relaxations: *variable splitting* (also called Lagrangian decomposition) and *constraint splitting*. We illustrate the first technique for a special case of the *SCFND*, the fixed-charge transportation problem, or *FCTP*, while the second technique is shown for the *MCFND*. Note that similar techniques can be used for other network design models, but are simpler to explain on these two problems.

Variable splitting first involves introducing copies of some of the variables, along with corresponding *copy equations*. Then, some constraints are rewritten by using the copy variables, in order to exhibit a decomposable structure in the Lagrangian subproblem obtained after relaxing the copy equations. We illustrate this technique for the *FCTP*, formulated as in Chap. 2, Sect. 2.3:

$$\text{Minimize} \sum_{(i,j)\in\mathscr{A}} \left(f_{ij}y_{ij} + c_{ij}x_{ij}\right) \tag{3.35}$$

$$\text{Subject to} \quad \textstyle\sum_{j\in\mathscr{N}_i^+} x_{ij} = |w_i|, \quad \forall\, i \in \mathscr{N}^o, \tag{3.36}$$

$$\textstyle\sum_{j\in\mathscr{N}_i^-} x_{ji} = |w_i|, \quad \forall\, i \in \mathscr{N}^d, \tag{3.37}$$

$$x_{ij} \le u_{ij}y_{ij}, \qquad \forall\,(i,j) \in \mathscr{A}, \tag{3.38}$$

$$x_{ij} \ge 0, \qquad \forall\,(i,j) \in \mathscr{A}, \tag{3.39}$$

$$y_{ij} \in \{0,1\}, \quad \forall\,(i,j) \in \mathscr{A}. \tag{3.40}$$

We create copies for all variables: s_{ij} is a copy of x_{ij} and t_{ij} is a copy of y_{ij} for all $(i, j) \in \mathscr{A}$. This is specified by the following copy equations:

$$x_{ij} = s_{ij}, \quad \forall\,(i,j) \in \mathscr{A}, \tag{3.41}$$

$$y_{ij} = t_{ij}, \quad \forall\,(i,j) \in \mathscr{A}. \tag{3.42}$$

We then rewrite constraints (3.37) using the copy variables and introduce redundant constraints equivalent to (3.38)–(3.40), but for the copy variables:

$$\textstyle\sum_{j\in\mathscr{N}_i^-} s_{ji} = |w_i|, \quad \forall\, i \in \mathscr{N}^d, \tag{3.43}$$

$$s_{ji} \le u_{ji}t_{ji}, \qquad \forall\,(j,i) \in \mathscr{A}, \tag{3.44}$$

$$s_{ji} \ge 0, \qquad \forall\,(j,i) \in \mathscr{A}, \tag{3.45}$$

$$t_{ji} \in \{0,1\}, \qquad \forall\,(j,i) \in \mathscr{A}. \tag{3.46}$$

This yields the following reformulation of the *FCTP*:

$$\text{Minimize} \sum_{(i,j)\in\mathscr{A}} \left(f_{ij}y_{ij} + c_{ij}x_{ij}\right) \tag{3.47}$$

Subject to (3.36), (3.38)–(3.46).

Relaxing the copy Eqs. (3.41) and (3.42) using Lagrange multipliers ξ_{ij} and ω_{ij}, $(i, j) \in \mathscr{A}$, respectively, we derive the following Lagrangian subproblem:

$$\text{Minimize} \sum_{(i,j)\in\mathscr{A}} \left\{(f_{ij} - \omega_{ij})y_{ij} + (c_{ij} - \xi_{ij})x_{ij}\right\} + \sum_{(i,j)\in\mathscr{A}} \left(\omega_{ij}t_{ij} + \xi_{ij}s_{ij}\right) \tag{3.48}$$

Subject to (3.36), (3.38)–(3.40), (3.43)–(3.46). This subproblem decomposes into two parts, one that depends on the original variables x and y, and the other on the copy variables s and t. Both parts further decompose by node, the first part for each node $i \in \mathscr{N}^o$ and the second part for each node $i \in \mathscr{N}^d$, yielding the Lagrangian bound $v(\xi, \omega) = \sum_{i \in \mathscr{N}^o} v_i^O(\xi, \omega) + \sum_{i \in \mathscr{N}^d} v_i^D(\xi, \omega)$, where

$$v_i^O(\xi, \omega) = \text{Minimize} \sum_{j \in \mathscr{N}_i^+} \left\{(f_{ij} - \omega_{ij})y_{ij} + (c_{ij} - \xi_{ij})x_{ij}\right\} \tag{3.49}$$

$$\text{Subject to} \quad \textstyle\sum_{j \in \mathscr{N}_i^+} x_{ij} = |w_i|, \tag{3.50}$$

$$x_{ij} \le u_{ij} y_{ij}, \qquad \forall j \in \mathscr{N}_i^+, \tag{3.51}$$

$$x_{ij} \ge 0, \qquad \forall j \in \mathscr{N}_i^+, \tag{3.52}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall j \in \mathscr{N}_i^+, \tag{3.53}$$

and

$$v_i^D(\xi, \omega) = \text{Minimize} \sum_{j \in \mathscr{N}_i^-} \left(\omega_{ji} t_{ji} + \xi_{ji} s_{ji}\right) \tag{3.54}$$

$$\text{Subject to} \quad \textstyle\sum_{j \in \mathscr{N}_i^-} s_{ji} = |w_i|, \tag{3.55}$$

$$s_{ji} \le u_{ji} t_{ji}, \qquad \forall j \in \mathscr{N}_i^-, \tag{3.56}$$

$$s_{ji} \ge 0, \qquad \forall j \in \mathscr{N}_i^-, \tag{3.57}$$

$$t_{ji} \in \{0, 1\}, \qquad \forall j \in \mathscr{N}_i^-. \tag{3.58}$$

Each of these node-based subproblems has the structure of a *single-node fixed-charge flow problem*, which we study further in Sect. 4. This Lagrangian subproblem does not have the integrality property, which implies that the Lagrangian dual bound can improve upon the LP relaxation bound.

We now illustrate the constraint splitting technique for the *MCFND*. The technique consists in dividing a set of constraints into several subsets that define a reformulation, and then to relax one of these subsets in a Lagrangian way to obtain a decomposable subproblem. For the *MCFND*, we divide the flow conservation equations into three subsets of constraints:

$$\textstyle\sum_{j \in \mathscr{N}_i^+} x_{ij}^k - \sum_{j \in \mathscr{N}_i^-} x_{ji}^k = 0, \quad \forall i \in \mathscr{N}, \forall k \in \mathscr{K}_i^T, \tag{3.59}$$

$$\textstyle\sum_{j \in \mathscr{N}_i^+} x_{ij}^k = d^k, \qquad \forall i \in \mathscr{N}, \forall k \in \mathscr{K}_i^O, \tag{3.60}$$

$$x_{ij}^k = 0, \qquad \forall i \in \mathscr{N}, \forall j \in \mathscr{N}_i^+, \forall k \in \mathscr{K}_j^O \cup \mathscr{K}_i^D, \tag{3.61}$$

where, for each node $i \in \mathcal{N}$, we define $\mathcal{K}_i^O = \{k \in K \mid i = O(k)\}$, $\mathcal{K}_i^D = \{k \in K \mid i = D(k)\}$ and $\mathcal{K}_i^T = \{k \in K \mid i \neq O(k), D(k)\}$. Constraints (3.59)–(3.61) exploit two basic properties: (1) for each $k \in \mathcal{K}$, the flow conservation equation at $i = D(k)$ is redundant; (2) because the costs are nonnegative, there is an optimal solution with no directed cycles, which implies that, for each arc $(i, j) \in A$, we have $x_{ij}^k = 0$ if $k \in \mathcal{K}_j^O$ or $k \in \mathcal{K}_i^D$.

We relax constraints (3.59) in a Lagrangian way, using Lagrange multipliers π_i^k, $i \in \mathcal{N}, k \in \mathcal{K}_i^T$. We also add the following valid inequalities to improve the relaxation:

$$\textstyle\sum_{j \in \mathcal{N}_i^+} x_{ij}^k \leq g_i^k, \quad \forall i \in \mathcal{N}, \forall k \in \mathcal{K}_i^T, \tag{3.62}$$

where $g_i^k = \min\{d^k, \sum_{j \in \mathcal{N}_i^-} u_{ji}\}$, $i \in \mathcal{N}, k \in \mathcal{K}_i^T$. The resulting Lagrangian subproblem decomposes by node, yielding the Lagrangian bound $v(\pi) = \sum_{i \in \mathcal{N}} v_i(\pi)$, where

$$v_i(\pi) = \text{Minimize} \sum_{j \in \mathcal{N}_i^+} \left(\sum_{k \in \mathcal{K}} c_{ij}^k(\pi) x_{ij}^k + f_{ij} y_{ij} \right) \tag{3.63}$$

$$\text{Subject to} \quad \textstyle\sum_{j \in \mathcal{N}_i^+} x_{ij}^k = d^k, \quad \forall k \in \mathcal{K}_i^O, \tag{3.64}$$

$$\textstyle\sum_{j \in \mathcal{N}_i^+} x_{ij}^k \leq g_i^k, \quad \forall k \in \mathcal{K}_i^T, \tag{3.65}$$

$$x_{ij}^k = 0, \quad \forall j \in \mathcal{N}_i^+, \forall k \in \mathcal{K}_j^O \cup \mathcal{K}_i^D, \tag{3.66}$$

$$\textstyle\sum_{k \in \mathcal{K}} x_{ij}^k \leq u_{ij} y_{ij}, \quad \forall j \in \mathcal{N}_i^+, \tag{3.67}$$

$$x_{ij}^k \leq b_{ij}^k y_{ij}, \quad \forall j \in \mathcal{N}_i^+, \forall k \in \mathcal{K}, \tag{3.68}$$

$$x_{ij}^k \geq 0, \quad \forall j \in \mathcal{N}_i^+, \forall k \in \mathcal{K}, \tag{3.69}$$

$$y_{ij} \in \{0, 1\}, \quad \forall j \in \mathcal{N}_i^+, \tag{3.70}$$

where

$$c_{ij}^k(\pi) = \begin{cases} c_{ij}^k + \pi_i^k - \pi_j^k, & \text{if } k \in \mathcal{K}_i^T \cap \mathcal{K}_j^T, \\ c_{ij}^k + \pi_i^k, & \text{if } k \in \mathcal{K}_i^T \setminus \mathcal{K}_j^T, \\ c_{ij}^k - \pi_j^k, & \text{if } k \in \mathcal{K}_j^T \setminus \mathcal{K}_i^T, \\ c_{ij}^k, & \text{if } k \in \mathcal{K}_i^O \cap \mathcal{K}_j^D, \end{cases} \quad \forall j \in \mathcal{N}_i^+, \forall k \in \mathcal{K}.$$

It is easy to see that this subproblem reduces to a *capacitated facility location problem*, where "depots" correspond to arcs (i, j), $j \in \mathcal{N}_i^+$, and "customers" are associated with commodities $k \in \mathscr{K}_i^O \cup \mathscr{K}_i^T$. This yields a Lagrangian subproblem that does not have the integrality property. Hence, the resulting Lagrangian dual bound can improve upon the LP relaxation bound.

This section has illustrated two useful techniques, variable splitting and constraint splitting, to derive Lagrangian relaxations that obtain potentially better lower bounds than the LP relaxation one. This comes at the cost, however, of having to solve more difficult Lagrangian subproblems. This is clearly the case for the Lagrangian relaxations we studied for the *MCFND*. Indeed, the shortest path and knapsack relaxations yield Lagrangian subproblems that are easy to solve, but their corresponding Lagrangian dual bounds are equal to the strong relaxation bound. By contrast, the relaxation studied in this section improves upon the strong relaxation bound, at the expense of having to solve capacitated facility location subproblems, which are notoriously difficult and for which some of the most efficient algorithms also use Lagrangian relaxation.

3 Relaxations by Projection and Benders Reformulations

In this section, we study the Benders reformulations that can be obtained for single-commodity and multicommodity network design models. The discussion is limited to modeling aspects, in particular the structure of the Benders subproblems. Additional considerations on Benders decomposition and its algorithmic features are presented in Sect. 7. We first present a primer on Benders decomposition, limiting ourselves to the modeling aspects that are relevant to the remainder of this section.

3.1 A Primer on Benders Decomposition

Given a model of the form $z = \min\{fy + cx \mid Ax + Dy \geq b,\ x \geq 0,\ y \in Y\}$, where Y is a bounded MILP feasible set, Benders decomposition first consists in reformulating the model as follows: $z = \min_{y \in Y}\{fy + \min_{x \geq 0}\{cx \mid Ax \geq b - Dy\}\}$. For any $y \in Y$, the Benders subproblem $v(y) = \min_{x \geq 0}\{cx \mid Ax \geq b - Dy\}$ is a linear program that can be reformulated with its dual $v(y) = \max_{\lambda \geq 0}\{\lambda(b - Dy) \mid \lambda A \leq c\}$, assuming $v(y)$ is unbounded whenever the Benders subproblem is infeasible for a given $y \in Y$. Thus, we obtain the following reformulation of the model: $z = \min_{y \in Y}\{fy + \max_{\lambda \geq 0}\{\lambda(b - Dy) \mid \lambda A \leq c\}\}$. A key advantage of this reformulation is that the dual polyhedron $\mathscr{D} = \{\lambda \geq 0 \mid \lambda A \leq c\}$ is independent of $y \in Y$, which implies that the solutions to the dual of the Benders subproblem can be either extreme points of $\mathscr{D}$, if the Benders subproblem is feasible, or extreme rays of $\mathscr{D}$, if the Benders subproblem is infeasible. We denote by $\mathscr{P}_{\mathscr{D}}$ and $\mathscr{R}_{\mathscr{D}}$, respectively, the two finite sets of extreme points and extreme rays of $\mathscr{D}$. Obviously, when the

Benders subproblem is infeasible for a given $y \in Y$, this cannot yield an optimal solution to the original model. Hence, such $y \in Y$ can be cut from the feasible domain using the property that $v(y)$ is bounded if and only if $\lambda(b - Dy) \leq 0,\ \forall\lambda \in \mathscr{R}_{\mathscr{D}}$. Each of these inequalities is called a *Benders feasibility cut*. Otherwise, for any $y \in Y$ that yields a feasible Benders subproblem, the corresponding optimal solution to its dual can be found at an extreme point of $\mathscr{D}$. Thus, the model can be reformulated as follows: $z = \min_{y \in Y}\{fy + \max_{\lambda \in \mathscr{P}_{\mathscr{D}}}\{\lambda(b - Dy) \mid \lambda(b - Dy) \leq 0,\ \forall\lambda \in \mathscr{R}_{\mathscr{D}}\}$. This last model can be simply linearized by introducing a variable v whose value we seek to minimize and by imposing the constraints $\lambda(b - Dy) \leq v,\ \forall\lambda \in \mathscr{P}_{\mathscr{D}}$. Each of these inequalities is called a *Benders optimality cut*. Thus, we arrive at the *Benders reformulation*:

$$z = \text{Minimize}\ \ fy + v \tag{3.71}$$

$$\text{Subject to}\ \ \lambda(b - Dy) \leq v,\ \ \forall\,\lambda \in \mathscr{P}_{\mathscr{D}}, \tag{3.72}$$

$$\lambda(b - Dy) \leq 0,\ \ \forall\,\lambda \in \mathscr{R}_{\mathscr{D}}, \tag{3.73}$$

$$y \in Y. \tag{3.74}$$

Since the projection of the polyhedron $\mathscr{F} = \{(x, y) \mid Ax + Dy \geq b,\ x \geq 0\}$ onto Y, $proj_Y(\mathscr{F}) = \{y \in Y \mid \exists x \geq 0,\ \ Ax + Dy \geq b\}$ corresponds to the polyhedron $\{y \in Y \mid \exists\lambda \geq 0,\ \lambda A \leq c,\ \lambda(b - Dy) \leq 0,\ \forall\lambda \in \mathscr{R}_{\mathscr{D}}\}$, the Benders reformulation is also called *reformulation by projection*. Relaxations can be obtained from that reformulation in two ways. First, by replacing the set Y with a larger set $\overline{Y} \supseteq Y$. In particular, if $\overline{Y}$ is obtained by relaxing the integrality constraints, the corresponding Benders reformulation is equivalent to the LP relaxation of the original model. Second, it is not realistic to assume that the sets $\mathscr{P}_{\mathscr{D}}$ and $\mathscr{R}_{\mathscr{D}}$ are known in advance. Hence, a natural relaxation is to generate only a subset of the corresponding Benders cuts. The process of gradually adding such cuts, thus tightening the relaxation, is called *Benders decomposition* and is described in more details in Sect. 7.

Now, we assume that the matrix (A, b) has a decomposable structure into $m = |\mathscr{L}|$ blocks, i.e., $\{x \geq 0 \mid Ax \geq b\} = \{x = (x_1, \ldots, x_m) \geq 0 \mid A_l x_l \geq b_l, l \in \mathscr{L}\}$. In general, such a structure cannot be exploited in the presence of y variables, i.e., the constraints $Ax + Dy \geq b$ are no more decomposable into m blocks. By projecting $\mathscr{F}$ onto Y, the Benders reformulation allows one to exploit the decomposition, since for any given $y \in Y$, we then have $\{x \geq 0 \mid Ax \geq b - Dy\} = \{x = (x_1, \ldots, x_m) \geq 0 \mid A_l x_l \geq (b - Dy)_l, l \in \mathscr{L}\}$, where $(b - Dy)_l$ corresponds to the lth component of (the constant) vector $b - Dy$. Thus, the Benders subproblem decomposes into m smaller subproblems and we can define the dual polyhedron accordingly $\mathscr{D} = \mathscr{D}_1 \times \cdots \times \mathscr{D}_m$, where $\mathscr{D}_l = \{\lambda_l \geq 0 \mid \lambda_l A_l \leq c_l\}$, $l \in \mathscr{L}$ (c_l is the lth component of c). Denoting $\mathscr{P}_{\mathscr{D}_l}$ and $\mathscr{R}_{\mathscr{D}_l}$, respectively, the sets of extreme points and extreme rays of $\mathscr{D}_l$, we can derive the *disaggregated Benders reformulation*

$$z = \text{Minimize}\ \ fy + \sum_{l \in \mathscr{L}} v_l \tag{3.75}$$

$$\text{Subject to}\quad \lambda_l(b-Dy)_l \le v_l,\ \ \forall\, l \in \mathscr{L}, \forall\, \lambda_l \in \mathscr{P}_{\mathscr{D}_l}, \tag{3.76}$$

$$\lambda_l(b-Dy)_l \le 0,\ \ \forall\, l \in \mathscr{L},\ \forall\, \lambda_l \in \mathscr{R}_{\mathscr{D}_l}, \tag{3.77}$$

$$y \in Y. \tag{3.78}$$

Of course, when the Benders subproblem has a decomposable structure, one can still use the standard, aggregated, Benders reformulation (3.71)–(3.74). In general, the disaggregated form (3.75)–(3.78) is preferable, for algorithmic reasons that we further discuss in Sect. 7.

We now focus on the *SCFND* and, then, on the *MCFND*, including some special cases of these problems. For all models, we define $Y = \{y_{ij} \in \{0,1\},\ (i,j) \in \mathscr{A}\}$.

3.2 Single-Commodity Formulations

For the *SCFND*, the Benders subproblem is a minimum cost network flow problem for any $y \in Y$:

$$\text{Minimize} \sum_{(i,j)\in\mathscr{A}} c_{ij}x_{ij} \tag{3.79}$$

$$\text{Subject to}\quad \textstyle\sum_{j\in\mathscr{N}_i^+} x_{ij} - \sum_{j\in\mathscr{N}_i^-} x_{ji} = w_i,\ \ \forall\, i \in \mathscr{N}, \tag{3.80}$$

$$0 \le x_{ij} \le u_{ij}y_{ij}, \qquad \forall\,(i,j) \in \mathscr{A}, \tag{3.81}$$

The dual of the Benders subproblem can then be written as follows, where π and α are the vectors of the dual variables associated with constraints (3.80) and (3.81), respectively:

$$\text{Maximize} \sum_{i\in\mathscr{N}} \pi_i w_i - \sum_{(i,j)\in\mathscr{A}} \alpha_{ij}u_{ij}y_{ij} \tag{3.82}$$

$$\text{Subject to}\quad \pi_i - \pi_j - \alpha_{ij} \le c_{ij},\ \ \forall\,(i,j) \in \mathscr{A}, \tag{3.83}$$

$$\alpha_{ij} \ge 0, \qquad \forall\,(i,j) \in \mathscr{A}, \tag{3.84}$$

The dual polyhedron is thus defined as follows: $\mathscr{D} = \{(\pi,\alpha)\,|\,\pi_i - \pi_j - \alpha_{ij} \le c_{ij},\ \alpha_{ij} \ge 0,\ \forall\,(i,j) \in \mathscr{A}\}$ with its sets of extreme points and extreme rays denoted $\mathscr{P}_{\mathscr{D}}$ and $\mathscr{R}_{\mathscr{D}}$, respectively. The Benders reformulation can then be written as follows:

$$\text{Minimize} \sum_{(i,j)\in\mathscr{A}} f_{ij}y_{ij} + v \tag{3.85}$$

$$\text{Subject to} \quad \textstyle\sum_{i\in\mathcal{N}} \pi_i w_i - \sum_{(i,j)\in\mathcal{A}} \alpha_{ij} u_{ij} y_{ij} \le v, \quad \forall (\pi,\alpha) \in \mathcal{P}_{\mathcal{D}}, \tag{3.86}$$

$$\textstyle\sum_{i\in\mathcal{N}} \pi_i w_i - \sum_{(i,j)\in\mathcal{A}} \alpha_{ij} u_{ij} y_{ij} \le 0, \quad \forall (\pi,\alpha) \in \mathcal{R}_{\mathcal{D}}, \tag{3.87}$$

$$y_{ij} \in \{0,1\}, \qquad \forall (i,j) \in \mathcal{A}. \tag{3.88}$$

Benders feasibility cuts (3.87) can be simplified by deriving necessary and sufficient conditions for a given $y \in Y$ to yield a feasible solution to the Benders subproblem. As discussed in Chap. 2, Sect. 2.1, the max-flow-min-cut theorem guarantees the existence of a feasible solution to the *SCFND* if and only if the following cut-set-based inequalities are satisfied:

$$\sum_{(i,j)\in(\mathcal{S},\overline{\mathcal{S}})} u_{ij} y_{ij} \ge W(\mathcal{S}), \ \forall \mathcal{S} \subset \mathcal{N}, \ W(\mathcal{S}) > 0, \tag{3.89}$$

where, as introduced in Chap. 2, $W(\mathcal{S}) = \sum_{i\in\mathcal{S}} w_i$ is the net supply across any cut $\mathcal{S} \subseteq \mathcal{N}$. These inequalities simply state that there should be enough capacity provided by the design solution $y \in Y$ to satisfy the demand across any cut.

Two special cases of the cut-set-based inequalities are worth mentioning. First, if we restrict ourselves to single-node cuts, i.e., $|\mathcal{S}| = 1$ or $|\overline{\mathcal{S}}| = 1$, then we obtain the following inequalities:

$$\textstyle\sum_{j\in\mathcal{N}_i^+} u_{ij} y_{ij} \ge |w_i|, \quad \forall i \in \mathcal{N}^o, \tag{3.90}$$

$$\textstyle\sum_{j\in\mathcal{N}_i^-} u_{ji} y_{ji} \ge |w_i|, \quad \forall i \in \mathcal{N}^d. \tag{3.91}$$

These inequalities guarantee that the design solution $y \in Y$ provides enough capacity to use the supply at each origin and meet the demand at each destination. A second special case of cut-set-based inequalities is obtained when we relax the capacity constraints by replacing u_{ij} with $W(\mathcal{N}^o)$ in each of them. We can divide the resulting cut-set inequalities by $W(\mathcal{N}^o)$ and round up the ratio $W(\mathcal{S})/W(\mathcal{N}^o) \le 1$, since the left-hand side $\sum_{(i,j)\in(\mathcal{S},\overline{\mathcal{S}})} y_{ij}$ is integer, to obtain the following connectivity inequalities:

$$\sum_{(i,j)\in(\mathcal{S},\overline{\mathcal{S}})} y_{ij} \ge 1, \ \forall \mathcal{S} \subset \mathcal{N}, \ W(\mathcal{S}) > 0. \tag{3.92}$$

As discussed in Chap. 2, Sect. 2.2, these inequalities are sufficient to characterize feasibility of $y \in Y$ for the *SUFND*. The particular structure of the Benders feasibility cuts, including these two special cases, can be exploited when solving the Benders subproblem for the *SCFND*, as discussed further in Sect. 7.

3.3 Multicommodity Formulations

For the *MCFND*, the Benders subproblem is a multicommodity minimum cost network flow problem for any $y \in Y$:

$$\text{Minimize} \sum_{k \in \mathscr{K}} \sum_{(i,j) \in \mathscr{A}} c_{ij}^k x_{ij}^k \tag{3.93}$$

$$\text{Subject to} \quad \textstyle\sum_{j \in \mathscr{N}_i^+} x_{ij}^k - \sum_{j \in \mathscr{N}_i^-} x_{ji}^k = w_i^k, \quad \forall\, i \in \mathscr{N}, \forall\, k \in \mathscr{K}, \tag{3.94}$$

$$\textstyle\sum_{k \in \mathscr{K}} x_{ij}^k \le u_{ij} y_{ij}, \qquad \forall\, (i,j) \in \mathscr{A}, \tag{3.95}$$

$$0 \le x_{ij}^k \le b_{ij}^k y_{ij}, \qquad \forall\, (i,j) \in \mathscr{A}, \forall\, k \in \mathscr{K}. \tag{3.96}$$

The dual of the Benders subproblem can then be written as follows, where π, α and β are the vectors of the dual variables associated with constraints (3.94), (3.95), and (3.96), respectively:

$$\text{Maximize} \sum_{i \in \mathscr{N}} \sum_{k \in \mathscr{K}} \pi_i^k w_i{}^k - \sum_{(i,j) \in \mathscr{A}} \left(\alpha_{ij} u_{ij} + \sum_{k \in \mathscr{K}} \beta_{ij}^k b_{ij}^k \right) y_{ij} \tag{3.97}$$

$$\text{Subject to} \quad \pi_i^k - \pi_j^k - \alpha_{ij} - \beta_{ij}^k \le c_{ij}^k, \quad \forall\, (i,j) \in \mathscr{A}, \forall\, k \in \mathscr{K}, \tag{3.98}$$

$$\alpha_{ij} \ge 0, \qquad \forall\, (i,j) \in \mathscr{A}, \tag{3.99}$$

$$\beta_{ij}^k \ge 0, \qquad \forall\, (i,j) \in \mathscr{A}, \forall\, k \in \mathscr{K}. \tag{3.100}$$

The dual polyhedron is thus defined as follows: $\mathscr{D} = \{(\pi, \alpha, \beta) \mid \pi_i^k - \pi_j^k - \alpha_{ij} - \beta_{ij}^k \le c_{ij}^k,\ \beta_{ij}^k \ge 0, \forall (i,j) \in \mathscr{A}, \forall k \in \mathscr{K},\ \alpha_{ij} \ge 0,\ \forall\, (i,j) \in \mathscr{A}\}$ with its sets of extreme points and extreme rays denoted $\mathscr{P}_{\mathscr{D}}$ and $\mathscr{R}_{\mathscr{D}}$, respectively. The Benders reformulation can then be written as follows:

$$\text{Minimize} \sum_{(i,j) \in \mathscr{A}} f_{ij} y_{ij} + v \tag{3.101}$$

Subject to

$$\sum_{i \in \mathscr{N}} \sum_{k \in \mathscr{K}} \pi_i^k w_i{}^k - \sum_{(i,j) \in \mathscr{A}} \left(\alpha_{ij} u_{ij} + \sum_{k \in \mathscr{K}} \beta_{ij}^k b_{ij}^k \right) y_{ij} \le v, \quad \forall\, (\pi, \alpha, \beta) \in \mathscr{P}_{\mathscr{D}}, \tag{3.102}$$

$$\sum_{i \in \mathscr{N}} \sum_{k \in \mathscr{K}} \pi_i^k w_i{}^k - \sum_{(i,j) \in \mathscr{A}} \left(\alpha_{ij} u_{ij} + \sum_{k \in \mathscr{K}} \beta_{ij}^k b_{ij}^k \right) y_{ij} \le 0, \quad \forall\, (\pi, \alpha, \beta) \in \mathscr{R}_{\mathscr{D}}, \tag{3.103}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall\, (i,j) \in \mathscr{A}. \tag{3.104}$$

In contrast with the single-commodity case, cut-set-based inequalities (see Chap. 2, Sect. 3.2) are necessary, but not sufficient to characterize feasible solutions to the *MCFND*. As a consequence, these inequalities form a proper subset of the Benders feasibility cuts (3.103).

When there are no arc capacities on the multicommodity flows, the Benders reformulation can be significantly simplified. Indeed, as discussed in Chap. 2, Sect. 3.2, the following connectivity inequalities are then both necessary and sufficient to characterize any feasible solution to the resulting *MUFND*:

$$\sum_{(i,j)\in(\mathcal{S},\overline{\mathcal{S}})} y_{ij} \geq 1, \ \forall \mathcal{S} \subset \mathcal{N}, \ \forall k \in \mathcal{K}, \ O(k) \in \mathcal{S}, \ D(k) \in \overline{\mathcal{S}}. \tag{3.105}$$

In addition, the Benders subproblem decomposes by commodity $k \in K$, yielding the following model for any $y \in Y$:

$$\text{Minimize} \quad \sum_{(i,j)\in\mathcal{A}} c_{ij}^k x_{ij}^k \tag{3.106}$$

$$\text{Subject to} \quad \textstyle\sum_{j\in\mathcal{N}_i^+} x_{ij}^k - \sum_{j\in\mathcal{N}_i^-} x_{ji}^k = w_i^k, \ \ \forall i \in \mathcal{N}, \tag{3.107}$$

$$0 \leq x_{ij}^k \leq d^k y_{ij}, \qquad \forall (i,j) \in \mathcal{A}. \tag{3.108}$$

Assuming this problem is feasible, i.e., there exists at least one path connecting $O(k)$ to $D(k)$, it reduces to the computation of a shortest path between $O(k)$ and $D(k)$ with nonnegative arc lengths, which can be solved efficiently by Dijkstra's algorithm.

The dual polyhedron associated with each $k \in \mathcal{K}$ is then defined as follows: $\mathcal{D}^k = \{(\pi^k, \beta^k) \,|\, \pi_i^k - \pi_j^k - \beta_{ij}^k \leq c_{ij}^k, \ \beta_{ij}^k \geq 0, \forall (i,j) \in \mathcal{A}, \}$ with its set of extreme points denoted $\mathcal{P}_{\mathcal{D}^k}$. Note that, after solving the Benders subproblem (3.106)–(3.108), we can easily derive π^k from the shortest path lengths and then β^k can be obtained by combining dual feasibility and complementary slackness conditions. The Benders reformulation for the *MUFND* can be written as follows, taking advantage of the fact that the Benders subproblem is decomposable by $k \in \mathcal{K}$:

$$\text{Minimize} \quad \sum_{(i,j)\in\mathcal{A}} f_{ij} y_{ij} + \sum_{k\in\mathcal{K}} v^k \tag{3.109}$$

Subject to (3.105) and

$$\sum_{i\in\mathcal{N}} \pi_i^k w_i{}^k - \sum_{(i,j)\in\mathcal{A}} \beta_{ij}^k d^k y_{ij} \leq v^k, \ \ \forall k \in \mathcal{K}, \forall (\pi^k, \beta^k) \in \mathcal{P}_{\mathcal{D}^k}, \tag{3.110}$$

$$y_{ij} \in \{0,1\}, \qquad \forall (i,j) \in \mathcal{A}. \tag{3.111}$$

The *MUFND* is thus particularly amenable to Benders decomposition, having three features that its capacitated counterpart, the *MCFND*, does not share: (1) Benders feasibility cuts can be easily generated as they reduce to connectivity inequalities; (2) the Benders subproblem can be efficiently solved by Disjktra's shortest path algorithm; (3) the Benders subproblem decomposes by $k \in \mathscr{K}$, thus allowing us to generate multiple Benders cuts each time the Benders subproblem is solved. Additional details on Benders decomposition for multicommodity formulations can be found in Sect. 7.

4 Valid Inequalities

In this section, we present valid inequalities for the *MCFND* that can be used in branch-and-cut algorithms (see Sect. 6). We focus on the *MCFND*, although the concepts can be easily adapted to single-commodity network design problems, as discussed below.

The inequalities that we study are based on cut-sets. In this context, for any proper subset $\mathscr{S}$ of $\mathscr{N}$, we recall the notations $(\mathscr{S}, \overline{\mathscr{S}}) = \{(i, j) \in \mathscr{A} : i \in \mathscr{S}, j \in \overline{\mathscr{S}}\}$ and $\mathscr{K}(\mathscr{S}, \overline{\mathscr{S}}) = \{k \in \mathscr{K} : O(k) \in \mathscr{S}, D(k) \in \overline{\mathscr{S}}\}$. In addition, given any nonempty subset of commodities $\mathscr{L} \subseteq \mathscr{K}$, we define $b_{ij}^{\mathscr{L}} = \min\{u_{ij}, \sum_{k\in\mathscr{L}} d^k\}$ for each arc $(i, j) \in \mathscr{A}$, which is an upper bound on the flow of the commodities in subset $\mathscr{L}$ on arc (i, j). Similarly, we introduce $w_i^{\mathscr{L}} = \sum_{k\in\mathscr{L}} w_i^k$ to represent the net supply of the commodities in $\mathscr{L}$ at any node $i \in \mathscr{N}$.

For given nonempty subsets $\mathscr{S} \subset \mathscr{N}$ and $\mathscr{L} \subseteq \mathscr{K}$, we obtain the following relaxation after summing flow conservation equations over these two sets:

$$\sum_{(i,j)\in(\mathscr{S},\overline{\mathscr{S}})} x_{ij}^{\mathscr{L}} - \sum_{(j,i)\in(\overline{\mathscr{S}},\mathscr{S})} x_{ji}^{\mathscr{L}} = \sum_{i\in\mathscr{S}} w_i^{\mathscr{L}}, \tag{3.112}$$

$$0 \le x_{ij}^{\mathscr{L}} \le b_{ij}^{\mathscr{L}} y_{ij}, \ \forall (i, j) \in \mathscr{A}, \tag{3.113}$$

$$y_{ij} \in \{0, 1\}, \ \forall (i, j) \in \mathscr{A}, \tag{3.114}$$

where $x_{ij}^{\mathscr{L}} = \sum_{k\in\mathscr{L}} x_{ij}^k$ for any arc $(i, j) \in \mathscr{A}$. It is clear that a similar relaxation can be derived for single-commodity network design by dropping the index $\mathscr{L}$ and by replacing $b_{ij}^{\mathscr{L}}$ by the capacity u_{ij} for each arc $(i, j) \in \mathscr{A}$. The inequalities that we now derive from this relaxation can thus be easily adapted to this case. Specifically, we study cover inequalities, as well as flow cover and flow pack inequalities. These inequalities are based on further relaxations that also appear in a large number of MILP models and are thus applicable not only to the *MCFND*, but also to many other problems.

4.1 Cover Inequalities

For the *MCFND*, we can express the right-hand-side of Eq. (3.112) as follows:

$$\sum_{i\in\mathscr{S}} w_i^{\mathscr{L}} = \sum_{k\in\mathscr{L}\cap\mathscr{K}(\mathscr{S},\overline{\mathscr{S}})} d^k - \sum_{k\in\mathscr{L}\cap\mathscr{K}(\overline{\mathscr{S}},\mathscr{S})} d^k.$$

Using (3.113) for each arc $(i, j) \in (\mathscr{S}, \overline{\mathscr{S}})$ and the inequality

$$\sum_{(j,i)\in(\overline{\mathscr{S}},\mathscr{S})} x_{ji}^{\mathscr{L}} \geq \sum_{k\in\mathscr{L}\cap\mathscr{K}(\overline{\mathscr{S}},\mathscr{S})} d^k,$$

we obtain the inequality

$$\sum_{(i,j)\in(\mathscr{S},\overline{\mathscr{S}})} b_{ij}^{\mathscr{L}} y_{ij} \geq \sum_{k\in\mathscr{L}\cap\mathscr{K}(\mathscr{S},\overline{\mathscr{S}})} d^k. \tag{3.115}$$

By restricting $\mathscr{L}$ to the extreme cases $\mathscr{L} = \mathscr{K}$ and $\mathscr{L} = \{k\}$ such that $O(k) \in \mathscr{S}$ and $D(k) \in \overline{\mathscr{S}}$, $k \in \mathscr{K}$, we obtain the cut-set-based inequalities for the *MCFND* introduced in Chap. 2, Sect. 3.2:

$$\sum_{(i,j)\in(\mathscr{S},\overline{\mathscr{S}})} u_{ij} y_{ij} \geq \sum_{k\in\mathscr{K}(\mathscr{S},\overline{\mathscr{S}})} d^k \tag{3.116}$$

and

$$\sum_{(i,j)\in(\mathscr{S},\overline{\mathscr{S}})} b_{ij}^k y_{ij} \geq d^k, \ \forall k \in \mathscr{K}, \ O(k) \in \mathscr{S}, \ D(k) \in \overline{\mathscr{S}}. \tag{3.117}$$

Using inequality (3.115), we can adapt to the *MCFND* concepts originally developed for the 0-1 knapsack problem: $\mathscr{C} \subseteq (\mathscr{S}, \overline{\mathscr{S}})$ is a *cover* if the total capacity of the arcs in $(\mathscr{S}, \overline{\mathscr{S}}) \setminus \mathscr{C}$ does not cover the demand, i.e., $\sum_{(i,j)\in(\mathscr{S},\overline{\mathscr{S}})\setminus\mathscr{C}} b_{ij}^{\mathscr{L}} < \sum_{k\in\mathscr{L}\cap\mathscr{K}(\mathscr{S},\overline{\mathscr{S}})} d^k$. In addition, a cover $\mathscr{C} \subseteq (\mathscr{S}, \overline{\mathscr{S}})$ is *minimal* if it is sufficient to open any arc in $\mathscr{C}$ to cover the demand, i.e., $\sum_{(i,j)\in(\mathscr{S},\overline{\mathscr{S}})\setminus\mathscr{C}} b_{ij}^{\mathscr{L}} + b_{pq}^{\mathscr{L}} \geq \sum_{k\in\mathscr{L}\cap\mathscr{K}(\mathscr{S},\overline{\mathscr{S}})} d^k$, $\forall (p, q) \in \mathscr{C}$. For every cover $\mathscr{C} \subseteq (\mathscr{S}, \overline{\mathscr{S}})$, we then obtain the *cover inequality* (CI)

$$\sum_{(i,j)\in\mathscr{C}} y_{ij} \geq 1. \tag{3.118}$$

This inequality states that at least one arc from the cover $\mathscr{C}$ must be selected in the final design solution in order to meet the demand. If $\mathscr{C}$ is a minimal cover, we can apply a lifting procedure to derive a facet of the convex hull of the set defined by (3.115) and $y_{ij} \in \{0, 1\}, \forall (i, j) \in (\mathscr{S}, \overline{\mathscr{S}})$. We discuss further the use of this inequality in the context of a branch-and-cut algorithm in Sect. 6.

4.2 *Flow Cover and Flow Pack Inequalities*

By relaxing Eq. (3.112) as follows:

$$\sum_{(i,j)\in(\mathscr{S},\overline{\mathscr{S}})} x_{ij}^{\mathscr{L}} - \sum_{(j,i)\in(\overline{\mathscr{S}},\mathscr{S})} x_{ji}^{\mathscr{L}} \le W_{\mathscr{S}}^{\mathscr{L}}, \tag{3.119}$$

where $W_{\mathscr{S}}^{\mathscr{L}} = \sum_{i\in\mathscr{S}} w_i^{\mathscr{L}}$, we obtain a single-node fixed-charge flow structure, defined over the variables $(x^{\mathscr{L}}, y)$ restricted to the arcs in $(\mathscr{S}, \overline{\mathscr{S}}) \cup (\overline{\mathscr{S}}, \mathscr{S})$. Two classes of inequalities have been derived for such structure, the flow cover and flow pack inequalities, which we now describe.

A flow cover $(\mathscr{C}_1, \mathscr{C}_2)$ is defined by two sets $\mathscr{C}_1 \subseteq (\mathscr{S}, \overline{\mathscr{S}})$ and $\mathscr{C}_2 \subseteq (\overline{\mathscr{S}}, \mathscr{S})$ such that $\mu = \sum_{(i,j)\in\mathscr{C}_1} b_{ij}^{\mathscr{L}} - \sum_{(j,i)\in\mathscr{C}_2} b_{ji}^{\mathscr{L}} - W_{\mathscr{S}}^{\mathscr{L}} > 0$. The *flow cover inequality* (FCI) is then defined as

$$\sum_{(i,j)\in\mathscr{C}_1} (x_{ij}^{\mathscr{L}} + (b_{ij}^{\mathscr{L}} - \mu)^+(1 - y_{ij})) \le \sum_{(j,i)\in\mathscr{D}_2} \min\{b_{ji}^{\mathscr{L}}, \mu\} y_{ji} + \sum_{(j,i)\in\mathscr{C}_2} b_{ji}^{\mathscr{L}} + W_{\mathscr{S}}^{\mathscr{L}} + \sum_{(j,i)\in(\overline{\mathscr{S}},\mathscr{S})\setminus\mathscr{C}_2\cup\mathscr{D}_2} x_{ji}^{\mathscr{L}}, \tag{3.120}$$

where $a^+ = \max\{0, a\}$ and $\mathscr{D}_2 \subset (\overline{\mathscr{S}}, \mathscr{S}) \setminus \mathscr{C}_2$.

Using the same notation as above, a flow pack $(\mathscr{C}_1, \mathscr{C}_2)$ is defined by two sets $\mathscr{C}_1 \subseteq (\mathscr{S}, \overline{\mathscr{S}})$ and $\mathscr{C}_2 \subseteq (\overline{\mathscr{S}}, \mathscr{S})$ such that $\mu = \sum_{(i,j)\in\mathscr{C}_1} b_{ij}^{\mathscr{L}} - \sum_{(j,i)\in\mathscr{C}_2} b_{ji}^{\mathscr{L}} - W_{\mathscr{S}}^{\mathscr{L}} < 0$. The *flow pack inequality* (FPI) is then defined as

$$\sum_{(i,j)\in\mathscr{C}_1} x_{ij}^{\mathscr{L}} + \sum_{(i,j)\in\mathscr{D}_1} (x_{ij}^{\mathscr{L}} - \min\{b_{ij}^{\mathscr{L}}, -\mu\} y_{ij})) \le \sum_{(j,i)\in(\overline{\mathscr{S}},\mathscr{S})\setminus\mathscr{C}_2} x_{ji}^{\mathscr{L}} + \sum_{(i,j)\in\mathscr{C}_1} b_{ij}^{\mathscr{L}} - \sum_{(j,i)\in\mathscr{C}_2} (b_{ji}^{\mathscr{L}} + \mu)^+(1 - y_{ji}), \tag{3.121}$$

where $\mathscr{D}_1 \subset (\mathscr{S}, \overline{\mathscr{S}}) \setminus \mathscr{C}_1$. The FPI can be viewed as a flow cover inequality for the relaxation defined by the inequality $\sum_{(j,i)\in(\overline{\mathscr{S}},\mathscr{S})} x_{ji}^{\mathscr{L}} - \sum_{(i,j)\in(\mathscr{S},\overline{\mathscr{S}})} x_{ij}^{\mathscr{L}} - t_{(\mathscr{S},\overline{\mathscr{S}})}^{\mathscr{L}} \le -W_{\mathscr{S}}^{\mathscr{L}}$, where $t_{(\mathscr{S},\overline{\mathscr{S}})}^{\mathscr{L}}$ is a slack variable.

Under mild conditions, both the FCI and the FPI can be lifted to obtain facet-defining inequalities for the convex hull of the set defined by (3.119) with the variables $(x^{\mathscr{L}}, y)$ restricted to the arcs in $(\mathscr{S}, \overline{\mathscr{S}}) \cup (\overline{\mathscr{S}}, \mathscr{S})$. We study the integration of these inequalities in a branch-and-cut algorithm in Sect. 6.

Part II: Enumeration Algorithms

5 Branch-and-Bound Algorithms

Branch-and-bound (B&B) algorithms are the most popular methods in MILP. They gradually generate feasible solutions (potentially all of them) by defining restrictions, also called subproblems. At each step, a subproblem (initially, the original problem) is partitioned into two or more subproblems defined in such a way that the union of their feasible domains corresponds to the set of all feasible solutions to the current subproblem. Such an operation, called *branching*, guarantees that no feasible solutions are "forgotten" along the process. For fixed-charge network design problems, branching is typically performed in a standard way by selecting a single binary design variable y_{ij} and by generating the two subproblems defined by adding the constraints $y_{ij} = 0$ and $y_{ij} = 1$, respectively. The difficulty lies in the choice of the variable to branch on, a topic that we cover in Sect. 5.2. This whole process corresponds to the construction of a tree, where each node is associated with a subproblem and there is an arc between two nodes, the parent and one of its children, when this child is generated from the parent by a branching operation.

To restrict the size of the tree, bounds on the optimal value of each subproblem need to be computed. More precisely, for any minimization problem, an upper bound can be obtained by computing feasible solutions to the subproblem, which are also feasible to the original problem, given the nature of the branching operations. In general, only the best feasible solution found so far, called the *incumbent*, is kept in memory, along with its objective value Z^*. This value is used to eliminate, or *fathom*, any node for which the feasible domain contains only solutions that cannot improve upon Z^*. To test this case, a lower bound $Z^l(Q)$ on the optimal value of every subproblem Q is computed by a suitably defined *relaxation*. Whenever $Z^l(Q) \geq Z^*$, we can fathom node Q. Note that this also includes the cases where the feasible domain of Q is empty, corresponding to $Z^l(Q) = +\infty$, and where Q is solved to optimality, i.e., a feasible solution (of value $Z^l(Q)$) for the original problem is obtained when solving the relaxation. In Sect. 5.1, we study some relaxations for network design problems, commonly used in B&B algorithms.

Note that we consider B&B algorithms that integrate advanced features, such as cut and column generation, in the subsequent sections of this chapter. In addition, heuristic methods for computing feasible solutions are mostly studied in Chap. 4. In Sect. 9, however, we consider heuristic methods that are tightly linked to the decomposition approaches studied in this chapter.

5.1 Relaxations

As seen in Chap. 2, Sect. 2, the LP relaxation for the *SCFND* reduces to a minimum cost network flow problem with transportation costs equal to $c_{ij} + f_{ij}/u_{ij}$ on each arc $(i, j) \in \mathscr{A}$. As a result, the LP relaxation can be solved efficiently at each node of the tree. In particular, when branching on a variable y_{ij}, the node associated with $y_{ij} = 0$ is evaluated by setting f_{ij} to an arbitrarily large value, while for the node associated with $y_{ij} = 1$, the fixed cost f_{ij} is added to the optimal value of the minimum cost network flow problem obtained by setting the transportation cost of arc (i, j) to c_{ij}. In general, the solution obtained by solving the LP relaxation at the parent node can be used to quickly reoptimize the LP relaxations of its children. Similar LP relaxations can be obtained for special cases of the *SCFND*, in particular the *FCTP*, for which the LP relaxation reduces to a classical transportation problem.

The situation becomes more complicated when considering the *MCFND*. The weak relaxation (introduced in Chap. 2, Sect. 3) is already difficult to solve, as it reduces to a minimum cost multicommodity network flow problem, which is a "structured" LP of typically large size that is often solved by decomposition techniques. To make things worse, as observed in Chap. 2, Sect. 3, the lower bound provided by the weak relaxation is generally far from the optimal value. Hence, the strong relaxation is used to derive tighter lower bounds. Adding all the strong linking constraints a priori and solving the model with a state-of-the-art LP solver is typically too slow. Adding these constraints in a dynamic way, within a cutting-plane method, is significantly more efficient. We explore this approach further in Sect. 6. Here, we consider an alternative approach based on the Lagrangian relaxation of the flow conservation constraints, which we studied in Sect. 2.3. The resulting Lagrangian subproblem can be solved efficiently, as it decomposes by arc, each of the resulting subproblems being reduced to a continuous knapsack problem. For this so-called knapsack relaxation, we noticed that the associated Lagrangian dual provides the same bound as the strong relaxation.

One approach to solve the Lagrangian dual is to apply column generation to the Dantzig–Wolfe reformulation presented in Sect. 2.3. Such an approach could be embedded in a branch-and-price algorithm to solve the *MCFND*, as explained in Sect. 8. Here, we outline a B&B algorithm based on a subgradient method to solve the Lagrangian dual.

Inspired by the classical gradient method for differentiable convex optimization, a subgradient method is an iterative algorithm for solving non-differentiable convex optimization problems, of which the Lagrangian dual is a special case. In this context, a subgradient method amounts to updating the values of the Lagrange multipliers by taking a step in the direction of a subgradient of the Lagrange function, which is easily computed by evaluating, at the optimal solution of the Lagrangian subproblem, the difference between the two sides of the constraints relaxed in a Lagrangian way.

For the knapsack relaxation, the subproblem for each arc $(i, j) \in \mathscr{A}$ can be written as follows, where π_i^k, $i \in \mathscr{N}$, $k \in \mathscr{K}$, are the Lagrange multipliers associated with the flow conservation equations:

$$\text{Minimize } v_{ij}(\pi, y_{ij}) = f_{ij}y_{ij} + \sum_{k\in\mathscr{K}} (c_{ij}^k + \pi_i^k - \pi_j^k)x_{ij}^k \tag{3.122}$$

$$\text{Subject to } \sum_{k\in\mathscr{K}} x_{ij}^k \leq u_{ij}y_{ij}, \tag{3.123}$$

$$0 \leq x_{ij}^k \leq b_{ij}^k y_{ij}, \quad \forall k \in \mathscr{K}, \tag{3.124}$$

$$y_{ij} \in \{0, 1\}. \tag{3.125}$$

By solving the continuous knapsack problem obtained when we fix $y_{ij} = 1$, we compute $v_{ij}(\pi, 1)$ and compare it with $v_{ij}(\pi, 0) = 0$ to derive the optimal value $v_{ij}(\pi) = \min\{v_{ij}(\pi, 0), v_{ij}(\pi, 1)\}$. If $v_{ij}(\pi) = v_{ij}(\pi, 1)$, the optimal solution is $\overline{y}_{ij} = 1$, with $\overline{x}_{ij}$ given by the solution of the continuous knapsack problem, and otherwise, the optimal solution is $\overline{y}_{ij} = \overline{x}_{ij} = 0$.

Given an optimal solution $(\overline{x}, \overline{y})$ to the Lagrangian subproblem (obtained by solving this subproblem for all arcs), a subgradient (of the Lagrange function at the point $\pi = (\pi_i^k)_{i\in\mathscr{N}}^{k\in\mathscr{K}}$) is a vector for which each component, corresponding to $i \in \mathscr{N}, k \in \mathscr{K}$, is computed as

$$\sum_{j\in\mathscr{N}_i^+} \overline{x}_{ij}^k - \sum_{j\in\mathscr{N}_i^-} \overline{x}_{ji}^k - w_i^k.$$

By taking a step in that direction, the Lagrange multipliers are updated and the next iteration starts by solving the Lagrangian subproblem with the new multipliers. The procedure stops when the current subgradient assumes value 0, meaning that none of the relaxed constraints is violated by the current solution to the Lagrangian subproblem. Unfortunately, that almost never happens for problem instances of even moderate size, even though we can show the theoretical convergence of the method under reasonable assumptions on the sequence of step sizes used to update the multipliers at each iteration. Hence, the subgradient method typically stops after a maximum number of iterations is achieved. In practice, it can be considered as a (relatively fast) heuristic for solving the Lagrangian dual.

At every iteration of the subgradient method, we obtain a lower bound on the optimal value of the *MCFND*, which is computed as:

$$Z^l(\pi) = \sum_{(i,j)\in\mathscr{A}} v_{ij}(\pi) - \sum_{i\in\mathscr{N}} \sum_{k\in\mathscr{K}} \pi_i^k w_i^k$$

At any given node of the tree, if this lower bound is larger than or equal to the incumbent value Z^*, the subgradient procedure is stopped and the node is fathomed. Since the subgradient method does not necessarily produce a non-decreasing sequence of lower bounds, we keep the best lower bound, along with the corresponding vector of Lagrange multipliers. When branching on a variable

y_{ij}, these multipliers can be used to initialize the children nodes. The child node corresponding to $y_{ij} = 0$ is then easily evaluated by setting all variables to 0 in the solution of the Lagrangian subproblem, with $v_{ij}(\pi) = 0$. The child node corresponding to $y_{ij} = 1$ is evaluated by solving the continuous knapsack problem (3.122)–(3.125) with the additional constraint $y_{ij} = 1$. The subgradient procedure can be resumed at each node, in order to solve (approximately) the Lagrangian dual of the restricted subproblem defining each B&B node.

In order for this B&B algorithm to converge, it is necessary to solve to optimality the multicommodity minimum cost network flow problem obtained when all design variables are fixed to 0 or 1. Otherwise, because the subgradient method is a heuristic for the Lagrangian dual, it is possible that it is terminated without reaching convergence, even when all design variables are fixed. In addition, to boost the performance of the B&B algorithm, one should integrate Lagrangian heuristics, at least at the root node of the B&B tree, a topic that we cover in Sect. 9.2. The other features that contribute to the performance of the method include the branching and filtering techniques, which we study next.

Another approach is based on the Lagrangian relaxation of the linking constraints (see Sect. 2.2), giving rise to Lagrangian subproblems that decompose by commodities and can be solved as shortest path problems. The resulting shortest path relaxation is used in a branch-and-price algorithm presented in Sect. 8.

5.2 Branching

Branching rules, both for general MILPs and for particular structures, have been the object of intense research over the last five decades. The state-of-the-art is based on the principle that, among several candidate variables to branch on, we should choose the one that contributes to the largest *weighted increase* in the lower bound, a term that we define precisely as follows. We are given a parent node with a computed lower bound Z^l and a set of candidate binary variables $\mathscr{C}$. For each variable y in $\mathscr{C}$, we compute estimates q_0 and q_1 of how much the lower bound would increase when adding $y = 0$ and $y = 1$, respectively. The weighted increase is then defined as $(1 - \mu)\min\{q_0, q_1\} + \mu \max\{q_0, q_1\}$, where μ is a parameter (typically set to a small value). Choosing the variable in $\mathscr{C}$ that achieves the largest weighted increase generally yield the best branching rules, which then differ in the way they compute the estimates q_0 and q_1.

In *pseudo-cost branching*, estimates are obtained by keeping track of the actual increases observed when branching on each variable and by computing average increases as estimates. The problem with this rule is that these estimates are totally unknown at the beginning of the tree exploration and quite uncertain for a while. It is only after accumulating a number of actual increases that these estimates become reliable, but poor branching decisions could have been made by then. In *strong branching*, estimates are computed by a (dual-based) heuristic. For example, if the lower bound at the parent node is computed by solving a linear program with the simplex method, the estimates are obtained by performing the dual simplex method for a limited number of iterations. The problem with this rule is that the procedure can be computationally heavy, although it generally produces smaller trees, unfortunately not always compensated by the large amount of time spent in branching. *Reliability branching* combines the advantages of the two former rules, pseudo-cost branching and strong branching. Unless "enough" actual increases have been computed for any candidate variable, the estimates for that variable are computed as in strong branching. Otherwise, the pseudo-cost estimates are considered reliable and used instead. Further refinements can contribute to the superior performance of the approach, in particular techniques to restrict the set of candidate variables for which we have to compute strong branching estimates.

A successful adaptation of these branching rules, typically applied to general MILPs, to fixed-charge network design problems clearly depends on the relaxations used at every node of the tree. To characterize a B&B node, we partition the set of arcs $\mathscr{A}$ into three categories: the closed arcs $\mathscr{A}_0$, with design variables fixed to 0; the open arcs $\mathscr{A}_1$, with design variables fixed to 1; and the free arcs $\mathscr{A}_{01}$, with design variables not yet fixed.

For the *SCFND*, a minimum cost network flow problem is solved, generating an optimal solution $\overline{x}$ from which we can derive the optimal solution in terms of the design variables: $\overline{y}_{ij} = \overline{x}_{ij}/u_{ij}$, for each $(i, j) \in \mathscr{A}_{01}$. As usual, the set of candidate variables is restricted to those that assume a fractional value, i.e., such that $0 < \overline{y}_{ij} < 1$, $(i, i) \in \mathscr{A}_{01}$, or equivalently, $0 < \overline{x}_{ij} < u_{ij}$, $(i, j) \in \mathscr{A}_{01}$. Assume that $\overline{x}$ is computed by the network simplex method, which implies that $\overline{x}$ is a primal basic feasible solution. Since adding any of the constraints $y_{ij} = 0$ or $y_{ij} = 1$ can be achieved by adjusting the transportation cost of arc (i, j), $\overline{x}$ is also a primal basic feasible solution for each of these two restricted subproblems. This means that we can compute strong branching estimates by reoptimizing efficiently with the network simplex method. Adapting reliability branching to this B&B algorithm for the *SCFND* is then straightforward.

We now turn our attention to the *MCFND*, for which we have proposed a Lagrangian-based B&B algorithm that makes use of subgradient optimization to compute the lower bounds at each node. In that case, all free variables are considered candidates for branching, since there is no fractional LP solution. The strong branching estimates can be obtained by performing subgradient optimization for a "small" number of iterations, using as input the best Lagrange multipliers from the current node. The procedure should be ran long enough to give it a chance to record an increase in the lower bound due to branching. As it can be costly to evaluate each candidate variable in this way, especially near the root node of the tree, when no variables are fixed, it is important to reduce the number of strong branching evaluations without compromising the overall performance of the branching scheme. In standard reliability branching, this is done by first sorting the candidate variables according to their pseudo-cost estimates, typically initialized with their cost and then gradually updated as actual increases are computed by strong branching evaluations. Here, we can use the Lagrangian subproblem instead to perform this task.

Looking back at the subproblem for each arc $(i, j) \in \mathscr{A}$, defined by (3.122)–(3.125), one can observe that, if $v_{ij}(\pi, 1) < 0$, then $\overline{y}_{ij} = 1$, otherwise, $\overline{y}_{ij} = 0$. In the first case, if we impose the constraint $y_{ij} = 0$, then the lower bound would increase by at least $-v_{ij}(\pi, 1)$, while in the second case, if we impose the constraint $y_{ij} = 1$, the lower bound would increase by at least $v_{ij}(\pi, 1)$. In other words, $|v_{ij}(\pi, 1)|$ can be seen as a weighted increase (with $\mu > 0$) and be used to choose only the "best" candidate variables for strong branching evaluations. Specifically, the free arcs can be sorted in non-increasing order of $|v_{ij}(\pi, 1)|$. Then, strong branching evaluations are performed in this order, but stopped as soon as the best candidate remains the same for λ consecutive strong branching evaluations. Typically, λ is a small value, say 4, so that strong branching evaluations are performed only for a fraction of the candidate variables. For example, let us assume that $\lambda = 4$ and that the second candidate in the initial order is better (according to strong branching evaluations) than the first, but also better than the next four candidates. Then, only five strong branching evaluations would be performed.

5.3 *Filtering*

Filtering methods are applied at every node of the tree. The general idea is to exclude solutions that cannot be optimal, given the current status of the design variables, i.e., the partition of the set of arcs into $\mathscr{A}_0$, $\mathscr{A}_1$ and $\mathscr{A}_{01}$. Special types of filtering are worth noting: *bound reduction* consists in decreasing (increasing) the upper (lower) bound on a single variable, while *variable fixing*, a special case of bound reduction, assigns a value to a single variable.

A common approach in filtering methods is to deduce from the addition of a constraint $\mathscr{C}$ the impossibility of finding an optimal solution that satisfies simultaneously $\mathscr{C}$ and the constraints that define the current B&B node. Hence, constraint $\neg\mathscr{C}$, the complement of $\mathscr{C}$, can be added to cut all solutions that satisfy $\mathscr{C}$. To infer that the addition of $\mathscr{C}$ cannot lead to an optimal solution, we generally compute a lower bound $Z^l(\mathscr{C})$ on the optimal value of the restricted problem derived from the addition of $\mathscr{C}$. If $Z^l(\mathscr{C}) \geq Z^*$, we can conclude that no optimal solution can be found when constraint $\mathscr{C}$ is added. A particular case of this test arises when we can deduce that no feasible solution can be obtained when $\mathscr{C}$ is added, since this case can be reduced to $Z^l(\mathscr{C}) = +\infty$.

For the *SCFND*, the reduced costs r_{ij} derived from the LP relaxation can be used to perform variable fixing. Indeed, for each non-basic variable x_{ij} at value $\overline{x}_{ij} \in \{0, u_{ij}\}$ and such that $(i, j) \in \mathscr{A}_{01}$, we have $r_{ij} \leq 0$ if $\overline{x}_{ij} = u_{ij}$, and $r_{ij} \geq 0$ if $\overline{x}_{ij} = 0$. If we add the constraint $y_{ij} = (1 - \overline{y}_{ij})$, where $\overline{y}_{ij} = \overline{x}_{ij}/u_{ij}$, then $Z^l + |r_{ij}|u_{ij}$ is a lower bound on the optimal value of the resulting problem, using standard LP duality theory. Therefore, if $Z^l + |r_{ij}|u_{ij} \geq Z^*$, then we can fix y_{ij} to value $\overline{y}_{ij}$. In addition, we can perform bound reduction and replace u_{ij} by $v_{ij} = (Z^* - Z^l)/r_{ij}$, whenever $\overline{x}_{ij} = 0$ and $r_{ij} > 0$.

For the *MCFND*, LP-based reduced costs are not available in our Lagrangian-based B&B algorithm. However, the values $v_{ij}(\pi, 1)$ can be interpreted as Lagrangian reduced costs associated with the variables y_{ij}. Indeed, for each $(i, j) \in \mathscr{A}_{01}$, as observed in Sect. 5.2, $Z^l + |v_{ij}(\pi, 1)|$ is a lower bound on the restricted problem obtained by adding the constraint $y_{ij} = 1 - \overline{y}_{ij}$. Consequently, if $Z^l + |v_{ij}(\pi, 1)| \geq Z^*$, then we can fix y_{ij} to value $\overline{y}_{ij}$.

We can also perform filtering based on feasibility. For instance, we can fix flow and design variables based on connectivity tests. Indeed, when, for some commodity k, an arc (i, j) does not belong to any path between $O(k)$ and $D(k)$, the upper bound b^k_{ij} associated with variable x^k_{ij} can be fixed to 0. Similarly, when, for some commodity k, an arc (i, j) belongs to all paths between $O(k)$ and $D(k)$, the lower bound associated with variable x^k_{ij} can be fixed to d^k. In addition, an arc (i, j) can be closed when it does not belong to any path between $O(k)$ and $D(k)$ for all commodities k. Conversely, an arc (i, j) can be opened when it belongs to all paths between $O(k)$ and $D(k)$ for at least one commodity k. This last test prevents the occurrence of infeasible subproblems due to a lack of connectivity. These tests can be easily performed using graph traversal algorithms. They can be triggered when some arcs have been closed since the last time the tests were performed. They can also be performed at the root node of the tree in order to simplify the problem instance.

6 Branch-and-Cut Algorithms

Branch-and-cut (B&C) algorithms are special cases of B&B that integrate a *cutting-plane procedure* in the computation of relaxation bounds. Given a set of valid inequalities, such an iterative procedure alternates between the solution of an LP relaxation and the addition of some of these valid inequalities to the LP relaxation. After solving the LP relaxation, if a fractional solution is obtained, then at least one valid inequality that cuts off (is violated by) that solution is added to the formulation and the next iteration is triggered. This process stops when an integer solution is obtained or when no more valid inequalities violated by the current fractional solution can be found. Such a cutting-plane procedure can be called at every node or at specific nodes of the tree, but is typically always invoked at the root node, since the cuts then added to the LP relaxation are propagated to every node of the tree.

The set of valid inequalities considered in a cutting-plane procedure is often partitioned into subsets, called *classes of valid inequalities*. For each class of valid inequalities, the *separation problem* is solved (either with a heuristic or an exact method): given a fractional solution, either at least one (ideally, the most) violated valid inequality is found or it is determined that there are no violated valid inequalities. Cuts identified through separation are often related to restrictions of the original problem obtained by fixing subsets of variables. In order to generate inequalities that are valid without any restrictions, a *lifting procedure* is applied. We now study separation and lifting procedures used in a cutting-plane method for the *MCFND*. This is a typical example of a difficult network design problem solved by a B&C algorithm that makes use of several classes of valid inequalities, namely strong linking, cover, flow cover and flow pack inequalities. In the remainder of this section, we use $(\overline{x}, \overline{y})$, with the appropriate indices, to denote the current fractional LP solution.

6.1 Separation and Lifting

Although there is a polynomial number, $O(|\mathscr{A}| \times |\mathscr{K}|)$, of strong linking inequalities, adding all of them to the LP relaxation yields large models that frequently exhibit degeneracy. Only a small number of these inequalities are typically added using a cutting-plane procedure. We note that the separation of strong linking inequalities is trivial, as it suffices to scan each arc and each commodity to identify all violated inequalities. For the cut-set-based inequalities introduced in Sect. 4, we assume a cut-set $(\mathscr{S}, \overline{\mathscr{S}})$ is given (see Sect. 6.2 for a description of cut-set generation procedures).

In our discussion on separation and lifting for cover inequalities, we use $\mathscr{L} = \mathscr{K}$ for the sake of simplicity, but we note that the generalization to any subset $\mathscr{L} \subseteq \mathscr{K}$ is easy. To simplify the notation, we then use $d_{\mathscr{S}} = \sum_{k \in \mathscr{K}(\mathscr{S}, \overline{\mathscr{S}})} d^k$. To generate a violated cover inequality (CI), we first determine, a priori, two subsets $\mathscr{C}_1$ (the open arcs) and $\mathscr{C}_0$ (the closed arcs) in $(\mathscr{S}, \overline{\mathscr{S}})$ that satisfy the condition

$$\sum_{(i,j) \in (\mathscr{S}, \overline{\mathscr{S}}) \setminus (\mathscr{C}_1 \cup \mathscr{C}_0)} u_{ij} \geq d_{\mathscr{S}} - \sum_{(i,j) \in \mathscr{C}_1} u_{ij} > 0.$$

The sets $\mathscr{C}_1$ and $\mathscr{C}_0$ are typically obtained by an iterative procedure that makes use of the current LP solution $\overline{y}$. At each iteration, such a procedure attempts to close an arc (i, j) with a small value $\overline{y}_{ij}$ such that the residual capacity after closing the arc still covers the residual demand, or to open an arc (i, j) with a large value $\overline{y}_{ij}$ such that there is still some residual demand to cover after opening the arc (i, j).

We then define the restricted cut-set inequality induced by $\mathscr{C}_1$ and $\mathscr{C}_0$ as

$$\sum_{(i,j) \in (\mathscr{S}, \overline{\mathscr{S}}) \setminus (\mathscr{C}_1 \cup \mathscr{C}_0)} u_{ij} y_{ij} \geq d_{\mathscr{S}} - \sum_{(i,j) \in \mathscr{C}_1} u_{ij}.$$

To define a cover $\mathscr{C}$ for this restricted cut-set inequality, a heuristic can be used that considers the arcs in non-decreasing order of $\overline{y}_{ij}$, thus attempting to exclude as much as possible from the set $\mathscr{C}$ the arcs with large $\overline{y}_{ij}$, in order to increase the chance of finding a violated inequality. Once a cover is obtained, it is easy to extract a minimal cover from it, by removing some of the arcs until the cover becomes minimal. Once the cover $\mathscr{C}$ is constructed, since the induced inequality is restricted to open arcs in $\mathscr{C}_1$ and closed arcs in $\mathscr{C}_0$, lifting (down for the variables in C_1 and up for the variables in C_0) is necessary to ensure its validity for the original problem. Note that, even if the identified CI is not violated, it is possible to find a violated one through the lifting procedure.

Lifting a CI consists in determining coefficients γ_{ij} for all $(i, j) \in (\mathscr{S}, \overline{\mathscr{S}}) \setminus \mathscr{C}$ such that

$$\sum_{(i,j)\in(\mathscr{S},\overline{\mathscr{S}})\setminus\mathscr{C}} \gamma_{ij} y_{ij} + \sum_{(i,j)\in\mathscr{C}} y_{ij} \geq 1 + \sum_{(i,j)\in(\mathscr{S},\overline{\mathscr{S}})\setminus(\mathscr{C}\cup\mathscr{C}_0)} \gamma_{ij}$$

is valid for the original problem. The lifting procedure is applied sequentially, meaning that the variables are lifted one after the other in some predetermined order. For each (i, j), the corresponding lifting coefficient γ_{ij} can be determined by solving a 0-1 knapsack problem. The quality of the resulting lifted inequality depends on the order in which the variables are lifted.

To generate flow cover and flow pack inequalities, we use two simpler valid inequalities. The first one is the *single-arc flow pack inequality* (SFPI), defined as:

$$\sum_{(i,j)\in\mathscr{C}_1'} x_{ij}^{\mathscr{L}} + x_{rt}^{\mathscr{L}} \leq \Big(\sum_{(j,i)\in\mathscr{C}_2'} b_{ji}^{\mathscr{L}} + W_{\mathscr{S}}^{\mathscr{L}}\Big) y_{rt} + \sum_{(j,i)\in(\overline{\mathscr{S}},\mathscr{S})\setminus\mathscr{C}_2'} x_{ji}^{\mathscr{L}} + (1 - y_{rt}) \sum_{(i,j)\in\mathscr{C}_1'} b_{ij}^{\mathscr{L}}, \tag{3.126}$$

where $(r, t) \in (\mathscr{S}, \overline{\mathscr{S}})$, $\mathscr{C}_1' \subseteq (\mathscr{S}, \overline{\mathscr{S}}) \setminus \{(r, t)\}$ and $\mathscr{C}_2' \subseteq (\overline{\mathscr{S}}, \mathscr{S})$. The second valid inequality is called the *single-arc flow cover inequality* (SFCI):

$$\sum_{(i,j)\in\mathscr{C}_1'} x_{ij}^{\mathscr{L}} + x_{rt}^{\mathscr{L}} \leq \Big(\sum_{(j,i)\in\mathscr{C}_2'} b_{ji}^{\mathscr{L}} + W_{\mathscr{S}}^{\mathscr{L}}\Big)(1 - y_{rt}) + \sum_{(j,i)\in(\overline{\mathscr{S}},\mathscr{S})\setminus\mathscr{C}_2'} x_{ji}^{\mathscr{L}} + y_{rt} \sum_{(i,j)\in\mathscr{C}_1'} b_{ij}^{\mathscr{L}}, \tag{3.127}$$

where $(r, t) \in (\overline{\mathscr{S}}, \mathscr{S})$, $\mathscr{C}_1' \subseteq (\mathscr{S}, \overline{\mathscr{S}})$ and $\mathscr{C}_2' \subseteq (\overline{\mathscr{S}}, \mathscr{S}) \setminus \{(r, t)\}$.

The interest of these single-arc inequalities is that their separation problems are simple, in contrast with the FCI and the FPI, which are hard to separate. Indeed, given an arc $(r, t) \in (\mathscr{S}, \overline{\mathscr{S}})$, separating the SFPI consists in setting

$$\mathscr{C}_1' = \{(i, j) \in (\mathscr{S}, \overline{\mathscr{S}}) \setminus \{(r, t)\} \mid \overline{x}_{ij}^{\mathscr{L}} > (1 - \overline{y}_{rt}) b_{ij}^{\mathscr{L}}\},$$

$$\mathscr{C}_2' = \{(j, i) \in (\overline{\mathscr{S}}, \mathscr{S}) \mid b_{ji}^{\mathscr{L}} \overline{y}_{rt} < \overline{x}_{ji}^{\mathscr{L}}\}.$$

For each subset $\mathscr{S}$, the separation procedure thus scans each arc in $(\mathscr{S}, \overline{\mathscr{S}})$, trying to find a violated SFPI associated with this arc. If $\mathscr{S}$ consists of a singleton containing the origin of commodity k, we set $\mathscr{L} = \{k\}$ and $\mathscr{C}_2' = \emptyset$, since in this case there is no flow of commodity k coming into r. Otherwise, we set $\mathscr{L} = \{k \in \mathscr{K} | \overline{x}_{rt}^k > 0\}$, in order to maximize the left-hand side of (3.126) and increase the chance of a violation. The separation procedure for the SFCI is derived in a similar way.

Once a violated SFPI is obtained, we lift the inequality to obtain an FPI. First, we set $\mathscr{C}_1 = \mathscr{C}_1'$, $\mathscr{C}_2 = \mathscr{C}_2'$ and $\mu = \mu'$. Then, we initialize $\mathscr{D}_1 = \{(r,t)\}$ and add to $\mathscr{D}_1$ each arc $(i, j) \in (\mathscr{S}, \overline{\mathscr{S}}) \setminus \mathscr{C}_1$ such that $\overline{x}_{ij}^{\mathscr{L}} - \min\{b_{ij}^{\mathscr{L}}, -\mu\}\overline{y}_{ij} > 0$. Finally, we lift the resulting FPI using a *sequence-independent lifting procedure*: we lift all variables in $\mathscr{C}_1$ and the variables in $(\overline{\mathscr{S}}, \mathscr{S}) \setminus \mathscr{C}_2$ such that $\overline{y}_{ij} = 0$. In addition, if $\mu' + b_{rt}^{\mathscr{L}} > 0$, we lift the violated SFPI to generate a violated FCI. We first add (r,t) to $\mathscr{C}_1'$ to obtain $\mathscr{C}_1$, set $\mathscr{C}_2 = \mathscr{C}_2'$ and compute $\mu = \mu' + b_{rt}^{\mathscr{L}}$. Then, for each arc $(i, j) \in \mathscr{C}_1$ such that $b_{ij}^{\mathscr{L}} > \mu$, we add to the left hand side of the inequality the term $(b_{ij}^{\mathscr{L}} - \mu)(1 - y_{ij})$. We then set $\mathscr{D}_2 = \{(j,i) \in (\overline{\mathscr{S}}, \mathscr{S}) \setminus \mathscr{C}_2 \mid \overline{x}_{ji}^{\mathscr{L}} > \min\{b_{ji}^{\mathscr{L}}, \mu\}\overline{y}_{ji}\}$. Finally, we lift the resulting FCI as follows: we lift all variables in $\mathscr{C}_2$ and the variables in $(\mathscr{S}, \overline{\mathscr{S}}) \setminus \mathscr{C}_1$ such that $\overline{y}_{ij} = 0$.

We proceed similarly when a violated SFCI is generated. First, we lift the inequality to derive a violated FCI. To this end, we set $\mathscr{C}_1 = \mathscr{C}_1'$, $\mathscr{C}_2 = \mathscr{C}_2'$ and $\mu = \mu'$, and then proceed as above to obtain a lifted FCI. If $\mu' - b_{rt}^{\mathscr{L}} < 0$, we also lift the violated SFCI to generate a violated FPI, by setting $\mathscr{C}_1 = \mathscr{C}_1'$, adding (r,t) to $\mathscr{C}_2'$ to obtain $\mathscr{C}_2$ and computing $\mu = \mu' - b_{rt}^{\mathscr{L}}$; then, we proceed as above to generate a lifted FPI.

6.2 Computational Issues

In any B&C algorithm, the computational effort required to generate cuts must translate into significant improvement in the relaxation bound, in particular at the root node of the tree. Typically, "many" cuts are generated at the root node of the tree, possibly by keeping them in memory through a so-called *cut pool*, while "few" additional cuts are generated at other nodes, especially the cuts that can be generated quickly and that provide significant improvement to the relaxation bound. In our case, the strong linking inequalities display such features: they are easy to generate and they improve, often significantly so, the lower bound. Hence, they should be generated at all nodes of the tree and before the cut-set-based inequalities. The latter are more expensive to generate and their generation should be performed only at the root node. In addition, these inequalities depend on the generation of cut-sets.

Single-node cut-sets that consists of one node i (origin or destination of at least one commodity) are easy to generate and should always be used to derive cut-set-based inequalities. Extending to subsets $\mathscr{S} \subset \mathscr{N}$ that contain more than one element is not trivial. Some key observations can be useful. First, if we consider constructing subsets of cardinality $l > 1$ from a subset of cardinality $l - 1$, the new subsets must add nodes that are connected by at least one arc to the nodes in the subset of cardinality $l - 1$ to avoid generating inequalities that are aggregations of previously generated ones. For example, given that we first consider single-node cut-sets, we can then generate subsets $\mathscr{S} \subset \mathscr{N}$ of cardinality 2 by merging two single-node subsets such that there is at least one arc between the two nodes. The same principle applies when building subsets of cardinality 3 or more. Second, it is possible to strengthen the cut-set-based inequality (3.115) with the following *metric inequality*:

$$\sum_{(i,j)\in(\mathscr{S},\overline{\mathscr{S}})} b_{ij}^{\mathscr{L}} y_{ij} \geq \sum_{k\in\mathscr{L}\cap\mathscr{K}(\mathscr{S},\overline{\mathscr{S}})} \pi_{\mathscr{S}}^{k} d^{k}, \tag{3.128}$$

where $\pi_{\mathscr{S}}^{k}$ is the smallest number of arcs in $(\mathscr{S}, \overline{\mathscr{S}})$ in any path between $O(k)$ and $D(k)$. For a given subset $\mathscr{S}$ such that $O(k) \in \mathscr{S}$, computing $\pi_{\mathscr{S}}^{k}$ is easy, as it reduces to the computation of a shortest path between $O(k)$ and $D(k)$ with arc lengths equal to 0, except for arcs in $(\mathscr{S}, \overline{\mathscr{S}})$, which are given arc lengths of 1.

Thus, when generating cut-set-based inequalities at the root node, a reasonable strategy is to generate cut-sets based on subsets of $\mathscr{N}$ of cardinality l, where l is relatively "small" (say, 2 or 3), while considering the two principles stated above (limiting ourselves to the extreme cases $\mathscr{L} = \mathscr{K}$ and $\mathscr{L} = \{k\}$, $k \in \mathscr{K}$). In this way, we can pre-generate several cut-sets, along with their corresponding metric inequalities, store them in memory, and then use them to generate violated cover, flow cover and flow pack inequalities. To generate cut-set-based inequalities for subsets of $\mathscr{N}$ of larger sizes, we can resort to approaches inspired by *metaheuristics* (studied in Chap. 4 in the context of deriving "good" feasible solutions to network design problems). Such approaches typically start with a *construction* procedure, which provides an initial partition of $\mathscr{N}$ into subsets of a given cardinality. They then perform a *local search* procedure that moves nodes among subsets around cycles, thus preserving the cardinality of the generated subsets of $\mathscr{N}$.

In addition to the generation of cuts at the different nodes of the tree, another fundamental issue in any B&C algorithm is its ability to reoptimize, i.e., to reuse information from the ancestor nodes to speed up the computation at the current node. This is typically achieved in two ways. First, by using a hybrid search strategy that combines the depth-first and best-first search approaches. After branching, the next node to evaluate is the child that gives the smallest estimated lower bound increase among the two generated children. The other child is stored in the *node pool*. When backtracking, we select the node that has the smallest lower bound estimate among all the nodes in the node pool. Second, when a node is handled immediately after its parent, the LP relaxation is simply reoptimized after taking into account the additions made by branching. When a node is obtained from backtracking, the LP relaxation is built by considering the LP solution from its parent and by adding cuts violated by this solution. This can be done efficiently, not only for the strong linking inequalities, but also for the cover, flow cover and flow pack inequalities, since the latter are generated only at the root and stored in the cut pool.

7 Benders Decomposition

In its simplest form, Benders decomposition is a cutting-plane algorithm that solves the Benders reformulation. At each iteration, a relaxation of the Benders reformulation, called the *master problem*, is solved, providing a lower bound and a solution y, which is then given as input to the Benders subproblem. If the Benders subproblem is infeasible, a Benders feasibility cut is generated and added to the master problem. Otherwise, we have identified a feasible solution x to the Benders subproblem, and hence a feasible solution (x, y) to the overall problem. The solution (x, y) is stored as the incumbent if its objective value improves upon that of the best known feasible solution. Then, either a Benders optimality cut is found and added to the master problem, or the relaxation bound and the incumbent objective value are equal, in which case the algorithm stops. This stopping test is necessarily satisfied at some point, since the master problem is a reformulation that contains a finite set of linear inequalities, i.e., the number of generated Benders cuts is finite.

This simple algorithmic scheme has several limitations that we now address in the case of network design problems. Specifically, we first study how to solve the LP relaxation of the Benders reformulation and how this simple feature can leverage the performance of Benders decomposition. Then, we see how the iterative cutting-plane algorithm outlined above can be modified to fit within a B&C framework. Finally, we discuss several computational issues that need to be addressed to speed up Benders decomposition.

7.1 Linear Programming Relaxation

When applying Benders decomposition, it is useful to start by solving the LP relaxation of the Benders reformulation obtained by replacing the constraint $y \in Y$ by $y \in \overline{Y}$. This LP relaxation can be solved by the cutting-plane algorithm presented above. Indeed, after solving the master problem (then, an LP model of, typically, small size), the solution y is given as input to the Benders subproblem. Then, either a Benders cut is generated and added to the LP master problem, or the method is stopped because the relaxation bound and the incumbent objective value are equal, where the incumbent is the best known feasible solution to the LP relaxation. So, when the stopping criterion is verified, the LP relaxation of the Benders reformulation, which is the same as the Benders reformulation of the LP relaxation of the original model, has been solved.

Applying Benders decomposition to solve the LP relaxation is interesting, as it allows to generate (potentially many) Benders cuts without having to solve the integer relaxed master problem to optimality. In fact, any $y \in \overline{Y}$ can be given as input to the Benders subproblem to generate Benders cuts. Since these cuts are expressed in terms of the extreme points and extreme rays of the dual polyhedron, which does not depend on y, the cuts generated when solving the LP relaxation might be useful and might then reduce the computational time needed to generate the cuts when solving the MILP model with the constraint $y \in Y$.

The LP relaxation of the Benders reformulation for the *SCFND* can be solved efficiently. The master problem can be initialized with single-node cut-set-based inequalities (3.90)–(3.91), which are valid for the LP relaxation of the original model. For any solution $y \in \overline{Y}$ to the master problem, we can exploit the particular structure of the Benders feasibility cuts, presented in Sect. 3.2, to solve efficiently the Benders subproblem. We first define the *support graph* $\mathscr{G}(y) = (\mathscr{N}, \mathscr{A}(y))$, where $\mathscr{A}(y) = \{(i, j) \in \mathscr{A} \mid y_{ij} > 0\}$. First, the violation of connectivity inequalities (3.92) is verified by a graph traversal algorithm that identifies all connected components of the support graph $\mathscr{G}(y)$. If at least one destination is disconnected from all origins, then a violated connectivity inequality is identified, where $\overline{\mathscr{S}}$ corresponds to the connected component associated with such destination. Second, if there are no violated connectivity inequalities, the violation of cut-set-based inequalities (3.89) can be verified by solving a maximum flow problem with capacities $u_{ij} y_{ij}$ on each arc (i, j) in the support graph augmented with a source s (or super-origin), connected to each origin $i \in \mathscr{N}^o$ by an arc (s, i) with capacity $|w_i|$, and a sink t (or super-destination), connected to each destination $i \in \mathscr{N}^d$ by an arc (i, t) with capacity $|w_i|$. If the capacity provided by $y \in \overline{Y}$ is insufficient, the solution to this maximum flow problem identifies a cut-set $(\mathscr{S}, \overline{\mathscr{S}})$ for which the corresponding inequality is violated. Third, if there are no violated cut-set-based inequalities, then the Benders subproblem is necessarily feasible, by the max-flow-min-cut theorem. In that case, the Benders subproblem is solved, typically by a specialized minimum cost network flow algorithm, to identify a violated Benders optimality cut (3.86).

A similar approach for solving the LP relaxation for the *MCFND* can be adopted, but key differences arise, showing the higher complexity of multicommodity network design. The master problem can be initialized with cut-set-based metric inequalities using subsets of $\mathscr{N}$ of "small" cardinality, in the same way as in the initialization of the B&C root node described in Sect. 6.2. Then, for any solution $y \in \overline{Y}$ to the master problem, the violation of connectivity inequalities (3.105) can be verified by a graph traversal algorithm that identifies all connected components of the support graph $\mathscr{G}(y)$. If any destination is disconnected from its origin, then a violated connectivity inequality is identified, where $\overline{\mathscr{S}}$ corresponds to the connected component associated with that destination. If there are no violated connectivity inequalities, we cannot solve a single maximum flow problem to verify the feasibility of the Benders subproblem for y, as we did for the *SCFND*. We can, however, solve one maximum flow problem per commodity to quickly identify Benders feasibility cuts of the form (3.117) without solving immediately the multicommodity Benders subproblem. To this end, for each commodity $k \in \mathscr{K}$, $O(k)$ and $D(k)$ are, respectively, the source and the sink in a maximum flow problem on the support graph with capacities $b^k_{ij} y_{ij}$ on each arc (i, j). The solution to this maximum flow problem can be used to identify a violated cut-set-based inequality of the form (3.117). If no violated cut-set-based inequalities can be found in this way, the Benders subproblem is not necessarily feasible. We have to solve the resulting multicommodity minimum cost network flow problem to identify a violated Benders cut, either an optimality one, (3.102), or a feasibility one, (3.103).

When solving the LP relaxation of the *MUFND*, several simplifications can be implemented, since the Benders feasibility cuts reduce to connectivity inequalities (3.105). A subset of these inequalities, corresponding to subsets of $\mathscr{N}$ of "small" cardinality, can be generated to initialize the master problem. Given a solution $y \in \overline{Y}$ to the master problem, the violation of connectivity inequalities can be verified in the same way as for the *MCFND*. If no such violated inequalities are found, we cannot conclude that the Benders subproblem is feasible, since there are capacities $d^k y_{ij}$ on each arc (i, j) for each commodity k, and $0 < d^k y_{ij} < d^k$ whenever y_{ij} has a fractional value. In that case, we can, however, solve one maximum flow problem per commodity k (with capacities $d^k y_{ij}$ on each arc (i, j)), in a similar way as we did for the *MCFND*, to identify further violated connectivity inequalities. If no such inequalities are found, we can conclude, contrary to the situation with the *MCFND*, that the Benders subproblem is feasible. The latter can then be solved as $|\mathscr{K}|$ minimum cost network flow problems with capacities $d^k y_{ij}$ on each arc (i, j) for each commodity k, which can be simplified to the computation of shortest paths by Dijkstra's algorithm in the case where $y \in Y$. Because the Benders subproblem decomposes by commodity, we can then generate up to $|\mathscr{K}|$ Benders optimality cuts of the form (3.110) at each iteration. This is in contrast with the *MCFND*, for which a single Benders cut is generated after solving the Benders subproblem.

7.2 Branch-and-Benders-Cut Algorithms

While the iterative Benders decomposition method is well-adapted to solve the LP relaxation, it has serious drawbacks when it is used to solve the MILP model. In particular, each iteration involves solving the relaxed master problem, which is an MILP model where only a subset of the Benders cuts have been generated. One can solve this MILP with a B&C algorithm, but that would imply calling this algorithm several times. While a fair amount of information from previous iterations could be used to initialize the B&C algorithm at a given iteration, the setup time would still be significant and the re-optimization capabilities limited. For this reason, Benders decomposition is mostly used within the framework of a single B&C algorithm. At the root node, the LP relaxation is solved with the cutting-plane procedure defined by the iterative Benders decomposition presented in Sect. 7.1. Using the terminology introduced when studying B&C algorithms, one can see the Benders subproblem as the separation problem for the class of valid inequalities defined as Benders cuts. At other nodes than the root, Benders cuts are typically generated only when an integer solution is found, since the corresponding node can be fathomed only if no violated Benders cuts can then be found. The resulting B&C algorithm is often called *Branch-and-Benders-Cut* (B&BC) to differentiate it from a "standard" B&C method.

We now illustrate this approach for the *MCFND* and contrast it with the B&C algorithm presented in Sect. 6 for the same problem. At the root node, we solve the LP relaxation of the Benders reformulation, as explained in Sect. 7.1. Note that this reformulation does not contain the flow variables as these have all been projected out and are handled in the Benders subproblem. The transportation costs are now captured in the single variable v, whose variation is controlled by the Benders optimality cuts. At nodes where an integral solution is found, Benders cuts can be generated in the same way as at the root node, i.e., first, by verifying connectivity, second, by solving single-commodity maximum flow problems, and third, by solving the multicommodity flow problem. Note that the solution to a maximum flow problem could still generate a violated inequality, even if the solution is integral, since the capacities b_{ij}^k on some arcs (i, j) can be smaller than d^k for some commodity k.

When comparing the B&BC and B&C algorithms, we note that two classes of valid inequalities used in the B&C method of Sect. 6 are easy to consider in the B&BC algorithm. First, the strong linking inequalities are added explicitly to the Benders subproblem, since when y is fixed, they become simple upper bounds on the flow variables. The information they contain is "captured" by the dual variables β that appear in the Benders cuts (3.102)–(3.103). Second, the cover inequalities involve only the design variables and can therefore be added to the master problem. They can be seen as strengthened cut-set-based inequalities that exploit the integrality of the design variables to cut fractional solutions. The flow cover and flow pack inequalities cannot be easily integrated in the B&BC algorithm, since their addition to the Benders subproblem would significantly complicate the latter, contrary to the strong linking inequalities.

In the special case of the *MUFND*, this B&BC algorithm has to be adapted to exploit the structure of the problem. At the root node, the LP relaxation of the Benders reformulation is solved as discussed in Sect. 7.1. Observe that, in this reformulation, the transportation costs per commodity k are captured in the variable v^k, reflecting the separability of the Benders cuts and subproblems. At nodes where an integer solution is found, the generation of Benders cuts can be significantly simplified. Indeed, verifying that there are no violated connectivity inequalities (3.105) is sufficient to conclude that the Benders subproblem is feasible. Also, any feasible Benders subproblem can be solved with $|\mathscr{K}|$ applications of Dijkstra's shortest path algorithm ($|\mathscr{N}|$ applications of that algorithm when transportation costs do not depend on the destinations), without resorting to maximum flow or minimum cost network flow algorithms, as is the case when y is a fractional solution at the root node.

7.3 Computational Issues

While the disaggregated Benders reformulation has the clear advantage of allowing the generation of multiple Benders cuts at each iteration, this feature might result in a very large number of added cuts and thus in an increase of the computational time needed to solve the master problem. A simple workaround is to solve only a subset of the separable Benders subproblems at each iteration. Consider the example of the *MUFND*, where the solution of the LP relaxation at the root node involves maximum flow and minimum cost network flow computations. Instead of solving all $|\mathscr{K}|$ subproblems, we could solve only a proportion of these, say approximately $|\mathscr{K}|/t$, where t is a positive integer, chosen in such a way that within t iterations of the method, all commodities are "covered."

Degeneracy is a major computational issue in Benders decomposition. In particular, a degenerate Benders subproblem implies that there are several optimal dual solutions. It is important to select among those the one that provides the "deepest" cut. This idea is formalized into the notion of *Pareto-optimal cut*, which is a cut that is not dominated by any other cut. To generate such a cut, one may solve an auxiliary subproblem, in addition to the dual Benders subproblem. The constraints of this auxiliary subproblem define the face of optimal dual solutions, while the objective maximizes the difference between the left-hand side of a Benders optimality cut evaluated at a so-called core point y^0, which, if appropriately chosen, guarantees to find a Pareto-optimal cut. While the details of this topic are beyond the scope of Chap. 3, we mention it here, because finding Pareto-optimal cuts efficiently can yield significant speedups when solving network design problems with Benders decomposition. In particular, multicommodity minimum cost network flow problems are often highly degenerate: ignoring this might result in poor performance of Benders decomposition.

Another important issue in Benders decomposition is the phenomenon of "oscillation" or "zig-zagging" in the sequence of solutions to the master problem: it is possible to move from a relatively good solution (as measured by the objective value) to a much worse one, which in turn has an effect on the quality of the Benders cuts. To avoid such phenomenon, several techniques have been proposed, which typically use a stabilizing point (in the interior of the convex hull of feasible solutions to the master problem) and try to identify cuts that do not deviate much from this point. The description of these so-called *stabilization* techniques is beyond the scope of Chap. 3, but they are extremely important to achieve good performance of Benders decomposition, especially to derive tight cuts at the root node of the B&BC tree. For problems where the Benders subproblem is not separable, such as the *MCFND*, a single cut is generated at each iteration of Benders decomposition at the root node. Hence, the generation of tight cuts is extremely slow in that case and stabilization techniques are essential to warm start the master problem as early as possible before starting the exploration of the tree.

8 Branch-and-Price Algorithms

In this section, we present *branch-and-price* (B&P) algorithms to solve the *MCFND* based on the Dantzig–Wolfe reformulations introduced in Sect. 2, namely the path-based reformulation, (3.16)–(3.21), and the knapsack-based reformulation, (3.30)–

(3.34). Since these two reformulations of the *MCFND* involve an exponential number of variables, we use a *column generation* procedure to solve the Dantzig–Wolfe reformulations, which are LP models.

Starting with a small number of variables, the procedure iterates between solving a restriction of the Dantzig–Wolfe reformulation and the generation of new variables through a *pricing subproblem*. The restriction of the Dantzig–Wolfe reformulation, called the *restricted master problem*, is simply the LP model for which the non generated variables are implicitly fixed to value 0. Once the restricted master problem is solved, new variables are sought by computing their reduced costs through the pricing subproblem. If variables with negative reduced costs are found, they are added to the restricted master problem. Since the restricted master problem is solved to optimality, all variables present in the LP master problem have nonnegative reduced costs, which imply that the columns with negative reduced costs identified by the pricing subproblem are necessarily new variables. The procedure stops when no columns with negative reduced costs are found when solving the pricing subproblem, which means that the Dantzig–Wolfe reformulation is solved.

Column generation solves the Dantzig–Wolfe reformulation (an LP model), but not necessarily the *MCFND*. In particular, we might end up the column generation procedure with a fractional solution, i.e., some design variables take fractional values. In this case, branching is needed to derive an optimal integer solution, which yields B&P algorithms that perform column generation at each node of the tree. Contrary to B&C algorithms, where most of the effort of generating cuts is spent at the root node, B&P algorithms perform column generation at every node of the tree. Otherwise, if column generation were applied only at the root node, then a restriction of the model (not a reformulation) would be solved by branching, because some variables with a positive value in an optimal integer solution could be forced to take value 0, as the corresponding columns would not have been generated at the root.

We first examine the structure of the pricing subproblems, then how branching and filtering are performed, for both the path-based and the knapsack-based reformulations. Finally, we discuss some key computational issues that arise when developing B&P algorithms for the *MCFND*.

8.1 Pricing Subproblems

For the path-based reformulation, (3.16)–(3.21), only the path variables h_p^k, $k \in \mathscr{K}$, $p \in \mathscr{P}^k$, need to be generated dynamically. The reduced cost $\bar{c}_p^k$ of variable h_p^k, $k \in \mathscr{K}$, $p \in \mathscr{P}^k$, can be computed as follows:

$$\bar{c}_p^k = \sum_{(i,j)\in\mathscr{A}} \delta_{ij}^p d^k \left(c_{ij}^k + \alpha_{ij} + \beta_{ij}^k\right) - \eta^k, \tag{3.129}$$

where η^k, $k \in \mathscr{K}$, α_{ij}, $(i, j) \in \mathscr{A}$ and β^k_{ij}, $(i, j) \in \mathscr{A}$, $k \in \mathscr{K}$ are the dual variables associated with constraints (3.17), (3.18), and (3.19), respectively. For the variables that already appear in the restricted master problem, we know that $\overline{c}^k_p \geq 0$, $k \in \mathscr{K}$, $p \in \mathscr{P}^k$. For the other path variables, not yet generated, it suffices to solve the Lagrangian subproblem (3.12)–(3.15), i.e., the shortest path relaxation. Indeed, after solving this relaxation, we obtain the shortest path $p \in \mathscr{P}^k$ for each commodity $k \in \mathscr{K}$ with respect to arc lengths $(c^k_{ij} + \alpha_{ij} + \beta^k_{ij})$ on each $(i, j) \in \mathscr{A}$, which is then "loaded" with the demand d^k to derive $\sum_{(i,j)\in\mathscr{A}} \delta^p_{ij} d^k (c^k_{ij} + \alpha_{ij} + \beta^k_{ij}) = \overline{c}^k_p + \eta^k$. Then, if $\overline{c}^k_p < 0$, the corresponding (new) path variable h^k_p is added to the master problem. Because the Lagrangian subproblem decomposes by commodity, this column generation procedure might add several columns at each iteration, at most one per commodity. Being able to generate several columns at each iteration is a desirable feature that typically improves the performance of column generation.

In order to generate columns, it is clearly not necessary to solve the part of the Lagrangian subproblem related to the design variables, i.e., the $|\mathscr{A}|$ problems solvable by inspection. However, the extra computations are negligible and they allow to generate a lower bound at every iteration. This is important, since the restricted master problem provides a lower bound only when the column generation has converged to an optimal solution of the Dantzig–Wolfe reformulation. Otherwise, because it is a restriction, the master problem at each iteration only gives an upper bound on the optimal value of the Dantzig–Wolfe reformulation. Having a lower bound at every iteration is important in the case where the column generation procedure experiments a "tailing off" effect, i.e., very slow convergence towards the end. In such a case, the column generation procedure might be stopped before the end, but a lower bound is still available for branching and filtering purposes.

Now, we turn our attention to the pricing subproblem for the knapsack-based reformulation, (3.30)–(3.34). In this case also, only the knapsack variables λ^q_{ij}, $(i, j) \in \mathscr{A}, q \in \mathscr{Q}_{ij}$, have to be generated in a dynamic way. The reduced cost $\overline{c}^q_{ij}$ of variable λ^q_{ij}, $(i, j) \in \mathscr{A}, q \in \mathscr{Q}_{ij}$, can be computed as follows:

$$\overline{c}^q_{ij} = \sum_{k\in\mathscr{K}} \left(c^k_{ij} + \pi^k_i - \pi^k_j\right) \xi^{kq}_{ij} + \theta_{ij}, \tag{3.130}$$

where $-\pi^k_i$, $i \in \mathscr{N}$, $k \in \mathscr{K}$ and θ_{ij}, $(i, j) \in \mathscr{A}$ are the dual variables associated with constraints (3.31) and (3.32), respectively. For the knapsack variables already in the master problem, we have $\overline{c}^q_{ij} \geq 0$, $(i, j) \in \mathscr{A}, q \in \mathscr{Q}_{ij}$. For the other knapsack variables, we need to solve the Lagrangian subproblem (3.25)–(3.29), i.e., the knapsack relaxation. Indeed, this relaxation decomposes by arc and provides a solution $(\widetilde{x}^k_{ij})_{k\in\mathscr{K}}$ to the continuous knapsack problem for arc (i, j) whenever $f_{ij} + \sum_{k\in\mathscr{K}} (c^k_{ij} + \pi^k_i - \pi^k_j)\widetilde{x}^k_{ij} < 0$. By construction $(\widetilde{x}^k_{ij})_{k\in\mathscr{K}}$ is an extreme point of the knapsack polyhedron $\{(x^k_{ij})_{k\in\mathscr{K}} \mid \sum_{k\in\mathscr{K}} x^k_{ij} \leq u_{ij}; 0 \leq x^k_{ij} \leq b^k_{ij}, k \in \mathscr{K}\}$, which we denote with the index $q \in \mathcal{Q}_{ij}$, i.e., $(\widetilde{x}_{ij}^k)_{k\in\mathcal{K}} \equiv (\xi_{ij}^{kq})_{k\in\mathcal{K}}$ In addition, it is a simple exercise (using dual feasibility and complementary slackness conditions) to show that there always exists an optimal solution to the dual of the Dantzig–Wolfe reformulation such that $\theta_{ij} = f_{ij}$ for each $(i, j) \in \mathcal{A}$. Hence, the condition $\overline{c}_{ij}^q < 0$ is equivalent to $f_{ij} + \sum_{k\in\mathcal{K}}(c_{ij}^k + \pi_i^k - \pi_j^k)\widetilde{x}_{ij}^k < 0$, which shows that the pricing subproblem corresponds to the Lagrangian subproblem. The resulting column generation procedure might add several columns at each iteration, at most one per arc. We note also that a lower bound is computed at every iteration, similarly to the path-based reformulation.

8.2 *Branching and Filtering*

Branching can be tricky in B&P algorithms, in particular when integer variables are represented with convex combinations of extreme points of underlying polyhedra. In that case, branching on the convex combination variables has the side effect of destroying the structure of the pricing subproblem. This is why specialized branching rules have been devised that typically branch on the original integer variables, which is usually easy to handle in the pricing subproblem. Both the path-based and the knapsack-based reformulations of the *MCFND* do not use convex combination variables to represent the binary design variables. This makes branching much easier, since the original design variables are present both in the master problem and in the pricing subproblem.

There are two obvious equivalent ways to perform branching for the two reformulations. Assuming we branch on variable y_{ij}, $(i, j) \in \mathcal{A}$, a first approach is simply to add the branching constraints $y_{ij} \leq 0$ and $y_{ij} \geq 1$ to the master problem, which introduces corresponding dual variables that are given as input to the pricing subproblem. Thus, the effect of the branching constraints immediately propagates to the pricing subproblem. A second approach consists in modifying the fixed cost f_{ij} instead of adding the branching constraints. Specifically, the effect of the branching constraint $y_{ij} \leq 0$ is captured by setting f_{ij} to an arbitrarily large value in both the master problem and the pricing subproblem. Likewise, the effect of the branching constraint $y_{ij} \geq 1$ is captured by setting $f_{ij} = 0$ and by adding the constant term f_{ij} to the objective value in both the master problem and the pricing subproblem. This last approach has some advantages, as it preserves the structures of both the master problem and the subproblem. In addition, if the master problem is solved with the primal simplex method, re-optimization is direct, as a primal basic feasible solution is readily available.

Concerning the selection of the variable y_{ij} to branch on, we first note that, contrary to the Lagrangian-based B&B algorithm for the *MCFND* presented in Sect. 5, fractional-valued variables are available from the optimal solution of the Dantzig–Wolfe reformulation. Thus, candidate variables for branching can be restricted to those with fractional values in the optimal solution of the Dantzig–Wolfe reformulation, in the same way as in standard B&B algorithms. Among these candidate variables, we can choose the one to branch on based on reliability branching, as discussed in Sect. 5.2. To compute the strong branching estimates, we can use the same approach, i.e., solve the knapsack subproblem to derive the estimates $|v_{ij}(\pi, 1)|$. For the knapsack-based reformulation, these have already been computed when solving the pricing subproblem. For the path-based reformulation, we can give as input to the knapsack relaxation the shortest path lengths π derived when solving the pricing subproblem.

In addition to their use for branching, the values $|v_{ij}(\pi, 1)|$ can be used to perform variable fixing, exactly in the same way as in the Lagrangian-based B&B for the *MCFND*, as explained in Sect. 5.3. In contrast to that algorithm, however, the B&P algorithms can also perform LP-based reduced cost fixing. Specifically, after solving the Dantzig–Wolfe reformulation, if we denote as r_{ij} the reduced cost of each non-basic free ($(i, j) \in \mathscr{A}_{01}$) variable y_{ij} at value $\overline{y}_{ij} \in \{0, 1\}$, if $Z^l + |r_{ij}| \geq Z^*$, then we can fix y_{ij} to value $\overline{y}_{ij}$, where Z^* is the incumbent value. Filtering based on connectivity tests can also be easily integrated into the B&P algorithms. For instance, using graph traversal algorithms, we can determine if an arc can be closed (if it does not belong to any path between $O(k)$ and $D(k)$ for all commodities k) or opened (if it belongs to all paths between $O(k)$ and $D(k)$ for at least one commodity k). Because these different filtering techniques involve only the design variables, they are as easy to integrate in the B&P algorithm as the branching constraints.

8.3 Computational Issues

The path-based reformulation, similar to the standard model for the *MCFND*, displays a large number of strong linking inequalities (3.19). It is impractical to generate all of them a priori, even for problems of moderate size, given that only a small fraction of them are active in an optimal solution of the Dantzig–Wolfe reformulation. Hence, in a similar way as in the B&C algorithm for the *MCFND* presented in Sect. 6, these inequalities should be generated in a dynamic way in a cutting-plane fashion. The resulting *column-and-row generation procedure* would alternate between column generation iterations (using the primal simplex method) and cutting-plane iterations (using the dual simplex method), until no more columns with negative reduced costs and no more violated inequalities can be found. The addition of cuts is easily managed in this case, since the separation problem for strong linking inequalities is trivial. Because of this last feature, it is preferable to generate strong linking inequalities at each node of the tree, not only at the root node. Note that positive values of the dual variables β are possible only for generated strong linking inequalities that are active at the current solution of the restricted master problem.

For the knapsack-based reformulation, the strong linking inequalities are part of the pricing subproblem, so they do not need to be generated through a cutting-plane procedure. It is possible, however, to generate other classes of valid inequalities, both for this reformulation and for the path-based one. The cover inequalities, introduced in Sect. 4.1, involve only the design variables and can be added to the Dantzig–Wolfe reformulation in a similar way as branching constraints: dual variables are associated with them and given as input to the pricing subproblem, thus preserving its structure. The flow cover and flow pack inequalities, described in Sect. 4.2, involve both the flow and the design variables. To integrate them in the Dantzig–Wolfe reformulation and to solve the corresponding separation problems, we need to "translate" the flow variables to the convex combination variables. For the path-based reformulation, this is simply achieved by the formula $x_{ij}^k = \sum_{p\in\mathscr{P}^k} d^k \delta_{ij}^p h_p^k$, $(i,j) \in \mathscr{A}$, $k \in \mathscr{K}$, while for the knapsack-based reformulation, we use $x_{ij}^k = \sum_{q\in\mathscr{Q}_{ij}} \xi_{ij}^{kq}\lambda_{ij}^q$, $(i,j) \in \mathscr{A}$, $k \in \mathscr{K}$. Because the separation problems for cover, flow cover and flow pack inequalities are difficult and time-consuming to solve (see Sect. 6.1), it is preferable to generate these inequalities only at the root node. This also has the advantage of "freezing" the set of Lagrange multipliers (associated with these inequalities) to give as input to the pricing subproblems, once the root node has been solved.

In a similar way as for Benders decomposition, degeneracy and "oscillations" (but, this time, in the dual space, i.e., the space of Lagrange multipliers) are major concerns to guarantee good performance of column generation. Subgradient optimization methods are often used to warm start the column generation procedure, providing at a relatively low computational cost both a set of "interesting" columns and good estimates of optimal Lagrange multipliers. Various stabilization techniques have also been proposed, typically based on defining a "good" dual point, also called *stability center*, with the goal of not deviating much from that point. To achieve this goal, one approach is to add a stabilizing term to the objective function that penalizes moves that are too far from the stability center. A full description of such stabilization techniques, in particular the well-known *bundle method*, is beyond the scope of Chap. 3, but it is important to mention that they have been successfully adapted to the *MCFND* and to other network design models (see Chap. 6).

Part III: Solution of Large-Scale Instances

9 Connections with Heuristic Methods

This section describes fundamental connections between the exact algorithms described in this chapter and heuristic methods that attempt to find "good" feasible solutions to network design problems, without necessarily reaching optimality. While heuristics, in particular metaheuristics and matheuristics, are covered extensively in Chap. 4, the heuristic methods presented here are tightly linked with the decomposition and enumeration approaches described in this chapter. In general, these heuristic methods provide not only feasible solutions, but also estimates of how good these solutions are, since they yield relaxation bounds. This is in contrast with several methods presented in Chap. 4.

We first present slope scaling methods, a class of basic heuristics that have been used successfully for solving several fixed-charge network design problems, both single-commodity and multicommodity, This class of heuristics turn out to play an important role in the design of Lagrangian heuristics, which we present next. The use of heuristics in Benders decomposition and in enumeration algorithms closes this section.

9.1 Slope Scaling Heuristics

We illustrate the *slope scaling heuristics* on the *MCFND*, but it should be clear from our developments how to adapt such heuristics to other fixed-charge network design problems. Slope scaling heuristics are based on the observation that feasible solutions to the *MCFND* can be obtained by solving multicommodity minimum cost network flow problems, an observation that is already exploited in Benders decomposition. While in that case, we give as input to the Benders subproblem the values of the design variables, in slope scaling heuristics, all arcs can be used, but modified transportation costs $\overline{c}_{ij}^k$, $(i, j) \in \mathscr{A}$, $k \in \mathscr{K}$, are given as input and updated along the iterations.

At iteration 0, initial modified transportation costs $\overline{c}(0)$ are provided as input, being typically derived from a relaxation. For example, using the simple linearization of the fixed costs derived from the weak relaxation (see Chap. 2, Sect. 3), one can use $\overline{c}_{ij}^k(0) = c_{ij}^k + f_{ij}/u_{ij}$, $(i, j) \in \mathscr{A}$, $k \in \mathscr{K}$. Other initial modified transportation costs, derived from Lagrangian relaxation, are discussed in Sect. 9.2.

At any iteration t, the following multicommodity minimum cost network flow problem is solved:

$$\text{Minimize} \sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}} \overline{c}_{ij}^k(t) x_{ij}^k \tag{3.131}$$

$$\text{Subject to} \quad \textstyle\sum_{j\in\mathscr{N}_i^+} x_{ij}^k - \sum_{j\in\mathscr{N}_i^-} x_{ji}^k = w_i^k, \quad \forall\, i \in \mathscr{N}, \forall\, k \in \mathscr{K}, \tag{3.132}$$

$$\textstyle\sum_{k\in\mathscr{K}} x_{ij}^k \le u_{ij}, \qquad \forall\,(i,j) \in \mathscr{A}, \tag{3.133}$$

$$x_{ij}^k \ge 0, \qquad \forall\,(i,j) \in \mathscr{A},\ \forall\, k \in \mathscr{K}. \tag{3.134}$$

Given a feasible solution $\overline{x}(t)$ to this problem, a feasible solution $(\overline{x}(t), \overline{y}(t))$ to the *MCFND* can be derived as follows: $\overline{y}_{ij}(t) = \lceil \sum_{k\in\mathscr{K}} \overline{x}_{ij}^k(t)/u_{ij} \rceil$, $(i, j) \in \mathscr{A}$, with objective value $Z(\overline{x}(t), \overline{y}(t)) = \sum_{(i,j)\in\mathscr{A}} \Big(f_{ij}\overline{y}_{ij}(t) + \sum_{k\in\mathscr{K}} c_{ij}^k \overline{x}_{ij}^k(t) \Big)$.

The modified transportation costs $\overline{c}^k_{ij}(t)$, $(i,j) \in \mathscr{A}$, $k \in \mathscr{K}$, are then updated as follows at any iteration $t > 0$, where we use the notation $\overline{\xi}_{ij}(t) = \sum_{k\in\mathscr{K}} \overline{x}^k_{ij}(t)$, $(i,j) \in \mathscr{A}$:

$$\overline{c}^k_{ij}(t) = \begin{cases} c^k_{ij} + \left(f_{ij}/\overline{\xi}_{ij}(t-1)\right), \text{ if } \overline{\xi}_{ij}(t-1) > 0, \\ \overline{c}^k_{ij}(t-1), \text{ otherwise.} \end{cases}$$

The procedure is stopped whenever two successive solutions provide the same objective value: $Z(\overline{x}(t), \overline{y}(t)) = Z(\overline{x}(t-1), \overline{y}(t-1))$. In that case, if $\overline{x}(t) = \overline{x}(t-1)$, the objective value of the multicommodity minimum cost network flow problem (3.131), corresponds to that of the *MCFND*:

$$\begin{aligned}\sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}} \overline{c}^k_{ij}(t)\overline{x}^k_{ij}(t) &= \sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}^+(t-1)} \left\{c^k_{ij} + \left(f_{ij}/\overline{\xi}_{ij}(t-1)\right)\right\} \overline{x}^k_{ij}(t) \\ &= \sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}} c^k_{ij}\overline{x}^k_{ij}(t) + \sum_{(i,j)\in\mathscr{A}^+(t)} \left\{\left(f_{ij}/\overline{\xi}_{ij}(t)\right) \sum_{k\in\mathscr{K}} \overline{x}^k_{ij}(t)\right\} \\ &= \sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}} c^k_{ij}\overline{x}^k_{ij}(t) + \sum_{(i,j)\in\mathscr{A}} f_{ij}\overline{y}_{ij}(t),\end{aligned}$$

where we use the notation $\mathscr{A}^+(t) = \{(i,j) \in \mathscr{A} \mid \overline{\xi}_{ij}(t) > 0\}$ for any $t \geq 0$.

Although the solution $(\overline{x}(t), \overline{y}(t))$ is designed to reflect the exact costs, it is computed by using the modified transportation costs, not the original ones. Hence, we can possibly (in fact, most of the times) improve upon its value by solving the Benders subproblem, (3.93)–(3.96), with $y = \overline{y}(t)$. The difference lies in the fact that the Benders subproblem, also a multicommodity minimum cost network flow problem, uses the original transportation costs. Note that this Benders subproblem is necessarily feasible, since $\overline{x}(t)$ is a feasible solution.

9.2 Lagrangian Heuristics

We illustrate the principle of Lagrangian heuristics on the *MCFND*, although it is straightforward to adapt it to other network design problems. We can use any of the Lagrangian relaxations for the *MCFND* introduced in Sect. 2.

The basic idea is to use the information derived from the Lagrangian subproblems to guide the derivation of feasible solutions of "good" quality. In its simplest form, a Lagrangian heuristic alternates between phases of Lagrangian dual optimization (using, for instance, a subgradient optimization method) and heuristic improvement. This last phase is given as input a partial solution (often not feasible) derived from the Lagrangian subproblems considered during the Lagrangian dual optimization phase. This partial solution is used to guide the construction of a feasible solution, which can then be further improved with specialized heuristics.

As an example, we use the knapsack relaxation for the *MCFND* and we assume that the Lagrangian dual is solved with a subgradient optimization method, as in Sect. 5. Suppose we interrupt the subgradient optimization method at regular intervals to extract the solution $(\overline{x}, \overline{y})$ to the last Lagrangian subproblem. Solving the Benders subproblem, (3.93)–(3.96), with $y = \overline{y}$, would then provide an upper bound on the optimal value of the *MCFND*, provided the Benders subproblem is feasible. Unfortunately, this is rarely the case and imposing constraints that ensure feasibility of the Benders subproblem is far from trivial, as discussed in Sect. 3.3. A slope scaling heuristic can be used to overcome this limitation. Indeed, at any iteration of a slope scaling heuristic, a feasible solution to the *MCFND* is obtained, since all arcs can be used. We can then simply penalize the arcs that are closed in the Lagrangian subproblem solution to guide the search for feasible solutions. This can be done by changing the initial modified transportation costs $\overline{c}_{ij}^k(0)$, $(i, j) \in \mathscr{A}$, $k \in \mathscr{K}$, given as input to the slope scaling heuristic: $\overline{c}_{ij}^k(0) = c_{ij}^k + (M(1 - \overline{y}_{ij}) + 1) f_{ij}/u_{ij}$, where M is a large positive number. This way, if $\overline{y}_{ij} = 0$, then the corresponding arc is heavily penalized, while if $\overline{y}_{ij} = 1$, we obtain the same initial modified transportation costs as in the standard slope scaling heuristic presented in Sect. 9.1. The slope scaling heuristic for the *MCFND* thus provides an elegant and effective approach to design Lagrangian heuristics: it gives a simple way to construct an initial feasible solution, guided by a Lagrangian subproblem solution, which is then improved through the slope scaling iterations.

In this example, we used a partial solution $\overline{y}$ derived from a single Lagrangian subproblem. However, it is in general more effective to consider a weighted average of the Lagrangian subproblem solutions generated since the start of the Lagrangian dual optimization method. For instance, we could give more weight to the subproblem solutions found more recently and less to those generated earlier. Such an approach would generate a partial solution $\overline{y}$ that is fractional, being a weighted average of several Lagrangian subproblem solutions. The slope scaling heuristic could still be performed as described above, with the interpretation that the initial modified transportation costs would penalize the arcs that are often closed, especially in recent Lagrangian subproblem solutions.

9.3 Benders Decomposition and Heuristics

A remarkable feature of Benders decomposition is its ability to generate Benders cuts when provided with any candidate design solution y. For the *MCFND*, we have used this feature to warm start the B&BC algorithm by solving the LP relaxation at the root node with Benders decomposition, as discussed in Sect. 7.1. Another way of exploiting this property is to perform heuristics before starting the B&BC to generate candidate design solutions y, which are then given as input to the Benders subproblem (3.93)–(3.96). After solving this subproblem, we can then generate Benders cuts that are used to initialize the B&BC algorithm, in addition to the cuts derived from solving the LP relaxation. An example of such an approach consists in applying the Lagrangian heuristic described in Sect. 9.2: at the end of each call to the slope scaling heuristic, the Benders subproblem is solved using as input the best solution found during the slope scaling iterations. It is then immediate to generate a Benders optimality cut from this solution and to use it to "feed" the initial relaxed master problem.

Another interesting feature of Benders decomposition is the ease with which it can integrate virtually any type of cut involving only the design variables. To illustrate this property, suppose we are given a partial assignment of values to the design variables, represented by the sets $\mathscr{A}_0$, $\mathscr{A}_1$, and $\mathscr{A}_{01}$ of closed, open and free arcs, respectively. If we can determine that no feasible solutions to the *MCFND* derived from this partial assignment could improve upon the best known feasible solution, then we could add the following *combinatorial Benders cut* to the Benders master problem:

$$\sum_{(i,j)\in\mathscr{A}_0} y_{ij} + \sum_{(i,j)\in\mathscr{A}_1} (1 - y_{ij}) \geq 1,$$

stating that at least one of the variables fixed to 0 or 1 must change its status in an improving feasible solution. We give an example of the use of such cut, particularly relevant in the context of using Benders decomposition to derive heuristics.

We assume that we are solving the LP relaxation by Benders decomposition, as in Sect. 7.1. At any iteration, we have a solution $\overline{y}$ such that the design variables can be partitioned into three subsets: $\mathscr{A}_0 = \{(i,j) \in \mathscr{A} \,|\, \overline{y}_{ij} \leq \delta_0\}$, $\mathscr{A}_1 = \{(i,j) \in \mathscr{A} \,|\, \overline{y}_{ij} \geq 1 - \delta_1\}$ and $\mathscr{A}_{01} = \{(i,j) \in \mathscr{A} \,|\, \delta_0 < \overline{y}_{ij} < 1 - \delta_1\}$, where $\delta_0 \geq 0$ and $\delta_1 \geq 0$ are parameters such that $\delta_0 + \delta_1 < 1$. The idea is to solve the restricted *MCFND* obtained by fixing y_{ij} to 0 for each $(i,j) \in \mathscr{A}_0$ and to 1 for each $(i,j) \in \mathscr{A}_1$, which we denote $M(\mathscr{A}_0, \mathscr{A}_1)$. If $M(\mathscr{A}_0, \mathscr{A}_1)$ is solved to optimality, we can then generate a combinatorial Benders cut, since we can then safely cut all feasible solutions to $M(\mathscr{A}_0, \mathscr{A}_1)$, given that we kept in memory its optimal solution. If $M(\mathscr{A}_0, \mathscr{A}_1)$ is too difficult to solve to optimality, we can still generate feasible solutions to $M(\mathscr{A}_0, \mathscr{A}_1)$, as well as corresponding Benders optimality cuts, for instance by solving $M(\mathscr{A}_0, \mathscr{A}_1)$ by B&BC with a limited computational time. There is one case where we can quickly generate a combinatorial Benders cut without trying to solve $M(\mathscr{A}_0, \mathscr{A}_1)$: when $M(\mathscr{A}_0, \mathscr{A}_1)$ is infeasible. This can happen only if the Benders subproblem for the candidate solution $\widetilde{y}$ is infeasible, where $\widetilde{y}_{ij} = 0$, if $(i,j) \in \mathscr{A}_0$, and $\widetilde{y}_{ij} = 1$, otherwise. In such a case, the combinatorial Benders cut can be strengthened to $\sum_{(i,j)\in\mathscr{A}_0} y_{ij} \geq 1$. The advantage of this inequality is that it cuts all solutions where the arcs in $\mathscr{A}_0$ are closed, as opposed to the standard Benders feasibility cut derived from $\widetilde{y}$, which cuts some of these solutions (including $\widetilde{y}$), but not necessarily all.

After trying to solve $M(\mathscr{A}_0, \mathscr{A}_1)$ to optimality, we can still generate the corresponding combinatorial Benders cut to eliminate all solutions where the arcs in $\mathscr{A}_0$ are closed and those in $\mathscr{A}_1$ are open. If $M(\mathscr{A}_0, \mathscr{A}_1)$ was not solved to optimality, this would turn the method into a heuristic. Otherwise, if $M(\mathscr{A}_0, \mathscr{A}_1)$ was solved to optimality, we would not lose any global optimal solution by adding the combinatorial Benders cut. We would, however, remove feasible solutions, which implies that the objective value of the master problem would not be anymore, in general, a lower bound on the optimal value of the unrestricted *MCFND*. This can be corrected by rewriting the combinatorial Benders cut with the following *L-shaped cut*, first proposed in the context of stochastic integer programming (see Chap. 9):

$$v \geq \left(Z^l - \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} \right) - \left(\sum_{(i,j)\in\mathscr{A}_0} y_{ij} + \sum_{(i,j)\in\mathscr{A}_1} (1 - y_{ij}) - 1 \right) (Z^* - Z^l),$$

where Z^l and Z^* are, respectively, lower and upper bounds on the optimal value of the unrestricted *MCFND*. In particular, when generating the cut, we could simply use the lower bound provided by the Benders master problem for Z^l and the value of the incumbent solution for Z^*. If Z^* is indeed the optimal value to $M(\mathscr{A}_0, \mathscr{A}_1)$ and it finally turns out to be the optimal value of the unrestricted *MCFND,* then we would have $\sum_{(i,j)\in\mathscr{A}_0} y_{ij} + \sum_{(i,j)\in\mathscr{A}_1}(1 - y_{ij}) = 0$ in an optimal solution, which would imply the valid inequality $\sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + v \geq Z^*$. Otherwise, the L-shaped cut would imply that $\sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + v \geq Z^l$, which is also valid.

9.4 Enumeration Algorithms and Heuristics

It is well-recognized that performing effective heuristics before starting the exploration of the tree is essential to achieve good performance in any enumeration algorithm. To achieve that goal for the *MCFND*, we can use the heuristics outlined in the previous sections, as well as the more sophisticated approaches described in Chap. 4.

In addition, we can also define restricted problems based on the solution of the relaxation and call the enumeration algorithm itself to solve that restricted problem, at least in a heuristic fashion, for instance by imposing a limited computational time. We have described such an approach in the context of Benders decomposition in Sect. 9.3. For the B&C and B&P algorithms described in Sects. 6 and 8, respectively, the principle is similar, since these algorithms involve solving an LP model to derive relaxation bounds. From a fractional solution $\overline{y}$, we partition the arcs into the three sets, $\mathscr{A}_0$, $\mathscr{A}_1$ and $\mathscr{A}_{01}$, as defined in Sect. 9.3. Then, we solve the restricted problem $M(\mathscr{A}_0, \mathscr{A}_1)$ defined by fixing to 0 the variables in $\mathscr{A}_0$ and to 1 the variables in $\mathscr{A}_1$. To solve $M(\mathscr{A}_0, \mathscr{A}_1)$, we call the enumeration algorithm itself, but by switching it to a heuristic mode: for instance, by "freezing" the model (no further addition of cuts or columns) and by searching the tree in a depth-first fashion to quickly find feasible solutions. For the Lagrangian-based B&B described in Sect. 5, a similar approach can be adopted, but instead of defining $\overline{y}$ based on the solution of an LP model, we then compute $\overline{y}$ as a weighted average of the Lagrangian subproblem solutions, as already discussed in Sect. 9.2.

Other than using the relaxation solutions to define restricted problems, it is also possible, for that purpose, to exploit the best feasible solutions already found by the enumeration algorithm. A popular approach, adopted in modern MILP solvers, is to define a population of elite solutions and to combine two (or more) of these elite solutions to define restricted problems. For example, two solutions $\overline{y}$ and $\widetilde{y}$ are chosen from the population and the arcs (i, j) for which the two solutions "agree" (i.e., $\overline{y}_{ij} = \widetilde{y}_{ij}$) are fixed, while the other variables are free.

10 Parallel Algorithms

Realistically-sized instances of network design problems are extremely difficult to solve to optimality. One reason is the large size of the models themselves, in terms of the number of variables and constraints, which justifies the development of the advanced decomposition methods presented in this chapter. Another reason is the combinatorial explosion that results from the huge number of network configurations, represented in the models by the binary design variables. For a large-size *MCFND* instance with 1000 arcs and 1000 commodities, there would be up to 2^{1000} network configurations, each involving the solution of a multicommodity network flow model having 1,000,000 variables. Parallel algorithms can contribute to curb this combinatorial explosion. In this section, we cover the development of parallel enumeration algorithms. As mentioned in Sect. 9, heuristics are essential components of enumeration algorithms, but their parallel implementations are beyond the scope of this chapter and are rather discussed in Chap. 4.

Three types of parallelism can be exploited in enumeration algorithms. Type 1, *node-based parallelism*, refers to the parallel computation of operations at each node of the tree, for example, solving relaxations and performing heuristics or branching. Type 2, *single-tree parallelism*, includes algorithms that perform the exploration of the tree concurrently on several processors. Type 3, *multiple-tree parallelism*, involves the concurrent exploration of several trees, where each tree has its own set of operations and parameter values. We now review each of these types of parallelism as they relate to the solution of network design problems. We use the *MCFND* as our representative problem in this discussion, although the concepts apply to other problems as well. We also discuss the potential for hybrid algorithms that combine several types of parallelism in a single method.

10.1 Node-Based Parallelism

In the context of solving the *MCFND*, decomposition arises naturally when defining relaxations and the resulting separable subproblems are directly amenable to parallel computations. When studying Lagrangian relaxations in Sect. 2, we have seen three approaches that yield separable Lagrangian subproblems: the shortest path relaxation, separable by commodity; the knapsack relaxation, separable by arc; and the facility location relaxation, separable by node. These three approaches are thus amenable to parallelization that could yield significant speedups when computing the lower bounds at each node of an enumeration tree, either in a Lagrangian-based B&B algorithm similar to the one described in Sect. 5, or in a B&P algorithm such as those presented in Sect. 8. There is a serious limitation though, which is the necessity to synchronize the computations at every iteration of a Lagrangian dual optimization method, either subgradient optimization or column generation, in order to generate a new set of Lagrange multiplier values. This is why the development of asynchronous algorithms that preserve convergence properties, while speeding up the computations by avoiding the necessity to synchronize at every iteration, is a topic for further research in the context of Lagrangian relaxation methods for network design.

Benders decomposition, even when the Benders subproblem is separable, apparently suffers from the same drawback. However, it has the remarkable property that several candidate solutions, generated for instance from several heuristics or relaxation methods, can be given to the Benders subproblem to generate multiple cuts in parallel. In that case, clever management of cuts must be performed to avoid repeating the same cuts several times or having too many inactive cuts. In any case, parallel computing appears especially promising in the context of building a strong initial Benders master problem before starting the B&BC algorithm. This could be seen as exploiting a special form of node-based parallelism to perform cut generation at the root node. Similar approaches could be used for B&C algorithms such as the one presented in Sect. 6. For instance, cuts for multiple reference points, not only for the current LP fractional solution, could be generated in parallel. This is similar in spirit to some stabilization approaches adopted in Benders or Dantzig–Wolfe decomposition.

10.2 Single-Tree Parallelism

Exploring an enumeration tree in parallel can be tricky, as the potential speedup obtained from this approach could be impaired by so-called *anomalies* that arise from performing much more work in parallel than in sequential. This happens whenever a branch is explored in parallel that would be fathomed in sequential. The reason for such extra work comes from the fact that incumbent values are generated in different branches of the tree. For example, consider a parallel algorithm on two processors that explores two subtrees, s_1 and s_2, one per processor. Subtree s_1 contains the optimal value, found after exploring 100 nodes, while subtree s_2 also explores 100 nodes before being communicated the optimal value found by subtree s_1. So, this parallel algorithm explores a total of 200 nodes. Assume, in addition, that if the optimal value was known when starting the exploration of

s_2, the subtree would be immediately fathomed, without exploring any node. If a sequential algorithm first explores subtree s_1, then it would generate only 100 nodes, compared to 200 nodes for the parallel algorithm. This would result in a speedup of 1, much less than the linear speedup of 2. The speedup could even be less than 1, considering the necessity to communicate between the two processors. To overcome this difficulty, we need to perform effective heuristics at the root node. If a near-optimal solution is found by these heuristics, anomalies then rarely happen.

Another tricky aspect in the exploration of an enumeration tree is the necessity to balance the workload among processors to avoid that too many processors fall out of work and remain idle for long periods. In general, the distribution of work follows three basic schemes, which induce different search control mechanisms: *master-slave*, *collegial* and *hybrid*. In a master-slave scheme, the generation of nodes (and thus, the control of the search) is centralized in a single processor, the master, that sends nodes to the other processors, called the slaves. The master performs branching operations, while the slaves compute the bounds at each node. This approach suffers from a scalability issue, as the number of processors increase, because of the necessity for the slaves to wait for the master to "feed" them. In a collegial, or distributed, approach, the nodes are distributed among processors, each of them managing its own tree. The control of the search is thus collectively assumed among the processors. It is then important to design efficient communication schemes to share information, in particular incumbent values, but also pseudo-costs, cuts or columns. In addition, workload balancing strategies are critical in such a scheme to ensure a successful implementation. In a hybrid scheme, part of the information is centralized in a single processor that receives them from each processor and broadcasts them to all. Each processor, except the centralized one, manages its own tree, but shares some nodes with the central processor that can distribute it to starving processors, even before waiting for them to become idle. As in a collegial scheme, the control of the search is collectively assumed among the processors, but the central processor plays a prominent role. In general, hybrid schemes show a good tradeoff between efficiency (by balancing the workload effectively as in a collegial approach) and robustness (by quickly broadcasting the relevant global information, in particular the incumbent, as in master-slave scheme), resulting in superior performance, compared to the two other approaches. The hybrid strategy can easily be adapted to both distributed or shared-memory parallel architectures.

The enumeration algorithms for the *MCFND* presented in Sects. 5–8 are all good candidates for such parallel implementation. Indeed, the computations at each node take a significant amount of time and the trees are typically large, even for moderate size instances, as soon as the fixed costs are important compared to the transportation costs. Of course, the parallel exploration of the tree becomes possible only when enough nodes are generated after the computations at the root node. In this case, to exploit parallelism as early as possible, we could use node-based parallelism at the root and at nodes near the root.

10.3 Multiple-Tree Parallelism

Any enumeration algorithm that solves a particular instance of a problem is prone to computational performances that vary significantly depending on the different components of the algorithm (relaxations, heuristics, branching rules, among others), as well as on the values of the parameters. In this context, it is almost impossible to identify a "best" sequential implementation of an enumeration algorithm that would solve every instance of a problem in the smallest computational time. It could then be advantageous to have several variants of enumeration algorithms performed in parallel, the idea being that, over a large set of instances, we could obtain a significant speedup when compared to a sequential implementation that uses particular algorithmic components and parameters.

Such strategy would be particularly useful when the heuristics at the root struggle to identify near-optimal solutions for most instances. In that case, optimal solutions would be identified in different branches of the tree and a parallel single-tree algorithm would be prone to anomalies. By using multiple-tree parallelism, with many different algorithmic strategies that showed good sequential performance, at least for some classes of instances, we protect ourselves against detrimental parallel performance. Clearly, multiple-tree parallelism can be combined with single-tree parallelism. For instance, assume that we have a distributed system of shared-memory multiprocessors, a very common architecture nowadays. Each multiprocessor in the distributed system could perform an enumeration algorithm, using different algorithmic components and parameter values than the other multiprocessors in the system, thus implementing multiple-tree parallelism. In addition, the enumeration algorithm performed by each multiprocessor could be parallelized among its own processors, resulting into single-tree parallelism.

Because the *MCFND* can be solved by the many enumeration algorithms presented in Sects. 5–8, it is clearly an interesting candidate for multiple-tree parallelism, given the fact that one can hardly identify a single "best" algorithm. A similar observation can be made concerning the many heuristics proposed to solve the *MCFND*, which also suggests that a similar approach (i.e., performing different heuristics in parallel) gives superior performance for the *MCFND*. Further discussion on this topic can be found in Chap. 4.

11 Bibliographical Notes

The challenges that arise when solving fixed-charge network design problems are already summarized in the survey papers that appeared in the 1980s, due to Magnanti and Wong (1984) and Minoux (1989). A summary of the exact algorithms based on decomposition for the *MCFND* can be found in Gendron et al. (1999) and Gendron (2011). More recent references on both single-commodity and multicommodity fixed-charge network design are provided in this section.

Lagrangian Relaxations and Dantzig–Wolfe Reformulations

Lagrangian relaxation for MILP models and its relationship with Dantizig-Wolfe reformulations is the main topic of Geoffrion (1974); Fisher (2004); Frangioni (2005), as well as being covered in classical textbooks on MILP (Nemhauser and Wolsey 1988; Wolsey 1998). The idea of decomposing large-scale structured LP models, which is the basis for column generation methods, is due to Dantzig and Wolfe (1960).

The relaxation of the linking constraints for the *MCFND* and its resulting shortest path subproblem appeared first in Gendron and Crainic (1994a). The associated Lagrangian dual is solved either with subgradient algorithms (Crainic et al. 2001; Frangioni et al. 2017) or with bundle methods (Crainic et al. 2001; Frangioni and Gorgone 2014). The structure of the Dantzig–Wolfe reformulation of the Lagrangian dual is exploited to develop a very efficient bundle method in Frangioni and Gorgone (2014). The dual-ascent method presented in Balakrishnan et al. (1989) for the *MUFND* can be seen as a specialized, fast, heuristic for solving the Lagrangian dual associated with the relaxation of the linking constraints.

The relaxation of the flow conservation constraints for the *MCFND* and its resulting knapsack subproblem appeared first in Gendron and Crainic (1994a). It is subsequently used in several Lagrangian-based B&B algorithms (Holmberg and Yuan 2000; Sellmann et al. 2002; Kliewer and Timajev 2005). The Lagrangian dual is solved either with subgradient algorithms (Holmberg and Yuan 2000; Crainic et al. 2001; Sellmann et al. 2002; Frangioni et al. 2017) or with bundle methods (Crainic et al. 2001; Kliewer and Timajev 2005). The Dantzig–Wolfe reformulation presented here is derived from a more general network design model in Frangioni et al. (2020). Gendron (2019) generalizes to a large class of network design models the results on the relative strength of the Lagrangian duals obtained by relaxing either the linking constraints or the flow conservation equations.

The application of variable splitting to the *FCTP* can be found in Zhao et al. (2018). Algorithms for the resulting single-node fixed charge flow problems are studied in Klose (2008); Görtz amd Klose (2009). The constraint splitting technique for the *MCFND* is due to Akhavan Kazemzadeh et al. (2021). The resulting Lagrangian subproblem reduces to capacitated facility location problems, often solved by Lagrangian relaxation (Klose and Görtz 2007; Görtz and Klose 2012). For the *MCFND*, stronger Lagrangian bounds based on variable splitting can also be found in Akhavan Kazemzadeh et al. (2021).

Relaxations by Projection and Benders Reformulations

The idea of projecting a MILP model onto the space of "complicating" integer variables to derive a more compact reformulation (at least in terms of the number of variables) is due to Benders (1962). The approach has since been applied to a large spectrum of problems, starting from the seminal work of Geoffrion and Graves (1974) on muticommodity facility location. The vast literature on Benders decomposition is reviewed in Rahmaniani et al. (2017), while the application of the approach to network design problems is the topic of Costa (2005).

To the best of our knowledge, Benders decomposition has not been used for solving the *SCFND* and its particular cases. We note, however, that the cut-set-based inequalities are often used in branch-and-cut algorithms, for instance for the *SUFND* (Ortega and Wolsey 2003) and for the *FCTP* (Agarwal and Aneja 2012). The structure of the cut-set-based inequalities for the *FCTP* is exploited in Göthe-Lundgren and Larsson (1994) to derive an algorithm for the pure *FCTP*, the variant of the *FCTP* without variable transportation costs. Single-source single-commodity network design problems are often reformulated as multicommodity network design problems (each destination is identified as a commodity), and Benders decomposition can then be used to address the resulting large-scale formulation (Ljubić et al. 2012).

The structure of Benders feasibility cuts for the *MCFND* is studied in Costa et al. (2009), while a Benders decomposition algorithm for the problem is developed in Costa et al. (2012). The application of the method to the *MUFND* is presented in Magnanti et al. (1986). Recently, Zetina et al. (2019) revisits Benders decomposition for the *MUFND*, taking advantage of the many refinements to the algorithm that have been proposed since the 1980s.

Valid Inequalities

Valid inequalities for MILP models are covered in classical textbooks (Nemhauser and Wolsey 1988; Wolsey 1998). A comprehensive survey can also be found in Wolsey (2003). Most of the material in Sect. 4 is adapted from Chouman et al. (2017). Valid inequalities based on cut-sets are part of the cutting-plane procedures of state-of-the-art MILP solvers, which are able to generate cover and flow cover inequalities (Gu et al. 1998, 1999b; Atamtürk 2005), but also to detect multicommodity network design structures to generate cut-set-based inequalities (Achterberg and Raack 2010).

Cover inequalities for the 0-1 knapsack problem were independently studied in Balas (1975); Hammer et al. (1975); Wolsey (1975). Computational issues related to their generation in the context of a general MILP solver are investigated in Gu et al. (1998, 1999a). The single-node fixed-charge flow problem and flow cover inequalities were first studied in Padberg et al. (1985). Subsequently, flow cover inequalities were integrated in general MILP solvers (Van Roy and Wolsey 1987; Gu et al. 1999b). Flow pack inequalities were proposed in Stallaert (1997); Atamtürk (2001). Other inequalities for the single-node fixed-charge flow problem are developed in Letchford and Souli (2019).

Apart from cut-sets, other concepts from network optimization can be exploited to derive valid inequalities for fixed-charge network design. In particular, it is possible to generalize the idea of partitioning the set of nodes into more than two subsets, giving rise to three- or four-partition inequalities (Atamtürk et al. 2016a; Agarwal and Aneja 2017). Paths of the network can also be used to derive inequalities (Van Roy and Wolsey 1987), in conjunction with flow cover and flow pack inequalities (Atamtürk et al. 2016b). Chapter 5 reviews other references relevant to the generation of valid inequalities for multicommodity network design models with general integer variables.

Branch-and-Bound Algorithms

"Branch-and-bound" is a term coined in Little et al. (1963), although the B&B algorithm for integer programs is attributed to Land and Doig (1960). It is true, however, that Manne and Markowitz (1957), for stating the principles behind the algorithm, and Eastman (1958), for essentially developing the algorithm for the traveling salesman problem, are precursors. The origins of B&B algorithms are discussed in Cook (2012).

The B&B algorithm that exploits the minimum cost network flow structure of the LP relaxation for the *SCFND* was first developed in Kennington and Unger (1976) for the *FCTP*. Subsequently, many researchers continued to focus on the *FCTP* by improving variable fixing and domain reduction techniques in the B&B algorithm (Barr et al. 1981; Cabot and Erenguc 1984, 1986; Palekar et al. 1990; Lamar and Wallace 1997; Bell et al. 1999).

The main ideas of the Lagrangian-based B&B algorithm for the *MCFND* that we describe in Sect. 5.1 were first presented in Holmberg and Yuan (2000), following work on a similar approach applied to the *MUFND* (Holmberg and Hellstrand 1998). Sellmann et al. (2002); Kliewer and Timajev (2005) use the same relaxation in their Lagrangian-based B&B algorithms for the *MCFND* and focus on improving variable fixing, heuristics and Lagrangian dual optimization.

Branching is an active topic of research in computational MILP. Benichou et al. (1971) proposed pseudo-cost branching, while strong branching was introduced in Applegate et al. (1995) as a key element in the Concorde code for solving very large-scale traveling salesman problems. The successful combination of these two techniques into the so-called reliability branching rule is due to Achterberg et al. (2005). All these rules are integrated into state-of-the-art MILP solvers. Reliability branching has been used in the B&C algorithm for the *MCFND* proposed in Chouman et al. (2018).

The term "filtering" is used in the area of constraint programming (Rossi et al. 2006) to describe domain reduction techniques that are called recursively at each node of the search tree. The description of filtering techniques for the *MCFND* in Sect. 5.3 follows the developments in Chouman et al. (2018). In the MILP community, filtering techniques, when applied at the root, are sometimes called "preprocessing," a term that captures filtering techniques like probing, but that also includes elimination of constraints and variables, and reduction of coefficients (Savelsbergh 1994). Filtering through the reduced costs (also called "penalties") derived from Lagrangian relaxation plays a significant role in the B&B algorithms for the *FCTP*, where it was first introduced in Cabot and Erenguc (1984) and subsequently improved in the references on B&B algorithms for the *FCTP* mentioned above.

Branch-and-Cut Algorithms

Cutting-plane methods originated from the seminal papers of Dantzig et al. (1954), on the traveling salesman problem, and of Gomory (1958), on MILP. The term "branch-and-cut" was first introduced in Padberg and Rinaldi (1987). The developments in Sect. 6 summarize the cutting-plane procedure presented in Chouman et al.

(2017) and the B&C algorithm for the *MCFND* described in Chouman et al. (2018). For single-commodity fixed-charge problems, B&C algorithms were proposed in Ortega and Wolsey (2003) for the *SUFND* and in Agarwal and Aneja (2012) for the *FCTP*.

Sequential lifting for cover inequalities was already proposed in Balas (1975); Wolsey (1975) to derive facet-defining inequalities for the convex hull of solutions to a 0-1 knapsack set. The approach was generalized and integrated in state-of-the-art MILP solvers in the 1990s (Gu et al. 1998). The generation of flow cover inequalities is often derived from covers (Nemhauser and Wolsey 1988; Gu et al. 1999b), while the approach described in Sect. 6.1, based on single-arc inequalities, is from Chouman et al. (2017). Sequence-independent lifting for cover inequalities was proposed in Balas (1975), and was subsequently strengthened in Gu et al. (2000), where the approach is generalized to mixed 0-1 programs. Sequence-independent lifting for flow cover and flow pack inequalities is due to Atamtürk (2001).

Heuristics for generating cut-sets to derive valid inequalities can be found in Ortega and Wolsey (2003); Achterberg and Raack (2010); Chouman et al. (2017). Metric inequalities are recognized as essential ingredients of any cutting-plane algorithm for multicommodity network design problems (Costa et al. 2009); see also Chap. 5. The importance of cutting-plane procedures in state-of-the-art MILP solvers, as well as the related computational issues, are discussed in Atamtürk and Savelsbergh (2005); Bixby and Rothberg (2007).

Benders Decomposition

As mentioned above, Benders decomposition originated from the seminal paper of Benders (1962). A successful application of the method was presented in Geoffrion and Graves (1974). Research on improving and generalizing the approach has culminated in automatic Benders decomposition now offered in state-of-the-art MILP solvers (Bonami et al. 2020).

Solving the LP relaxation by Benders decomposition to warm start the generation of cuts for the MILP model is due to McDaniel and Devine (1977). This algorithmic refinement has been applied to the *MCFND* (Costa et al. 2012) and to the *MUFND* (Zetina et al. 2019). Accelerating the generation of Benders feasibility cuts for network design through maximum flow computations is also common (Ljubić et al. 2012; Zetina et al. 2019).

Embedding Benders decomposition within a B&C framework has been used at least since the late 1990s. The idea can be found in the network design literature, for instance, in Sridhar and Park (2000); Fortz and Poss (2009); Ljubić et al. (2012). The integration of additional valid inequalities in the resulting BB&C algorithm is advocated as a strong advantage of the approach. For example, strong linking inequalities and cover inequalities are generated for the network design problem considered in Ljubić et al. (2012).

The concept of Pareto-optimal Benders cuts is due to Magnanti and Wong (1981). It has since been used in many implementations of Benders decomposition, in particular for the *MCFND* (Costa et al. 2012; Naoum-Sawaya and Elhedhli 2013). Popular stabilization techniques for Benders decomposition are based on the “in-out" principle (Fischetti et al. 2016), where a convex combination of the current master problem solution and a stability center is separated, and on the addition of local branching constraints (Baena et al. 2020), which are gradually "reversed" to guarantee convergence. A recent survey of computational issues and algorithmic refinements of Benders decomposition applied to the stochastic *MCFND* (see Chap. 9) can be found in Rahmaniani et al. (2018).

Branch-and-Price Algorithms

Column generation in LP started with the seminal papers of Ford and Fulkerson (1958), on the multicommodity maximum flow problem, and Dantzig and Wolfe (1960), on its generalization to block-structured LP models. Its first use for solving integer programs can be found in the work of Gilmore and Gomory (1961, 1963) on the cutting stock problem. Applying column generation at each node of the B&B tree was introduced in Desrosiers et al. (1984), while the term "branch-and-price" was popularized in the 1990s (Barnhart et al. 1998).

B&P algorithms for the *FCTP* have been proposed recently (Roberti et al. 2015; Mingozzi and Roberti 2018). Roberti et al. (2015) solve by column generation the Dantzig–Wolfe reformulation associated with the relaxation of the demand constraints at the destinations. In Mingozzi and Roberti (2018), column generation is used to solve the Dantzig–Wolfe reformulation associated with the variable splitting approach presented in Sect. 2.4; see also Zhao et al. (2018).

Column generation for the *MCFND* was first proposed in Crainic et al. (2001), where the bundle method, which can be interpreted as a stabilized variant of column generation (Ben Amor et al. 2009), is used to optimize the Lagrangian duals for both the shortest path and the knapsack relaxations. Note, however, that the Dantzig–Wolfe reformulations used in Crainic et al. (2001) do not exploit the structure, as they introduce one master problem variable per (aggregated) solution of the Lagrangian subproblem. It is only in Frangioni and Gorgone (2014) that the decomposable structure of the shortest path relaxation is exploited to develop a bundle method that solves (very efficiently) the Dantzig–Wolfe reformulation presented in Sect. 2.2. Concerning the knapsack relaxation, the "quasi-separable" structure revealed in the Dantzig–Wolfe reformulation presented in Sect. 2.3 is the topic of Frangioni et al. (2020). In Akhavan Kazemzadeh et al. (2021), the facility location relaxation, presented in Sect. 2.4, is solved by a stabilized column generation method that exploits its separability by node.

To the best of our knowledge, the only papers on exact algorithms for the *MCFND* that use column generation are due to Kliewer and Timajev (2005) and Gendron and Larose (2014). The first one makes use of the knapsack relaxation, where the Lagrangian dual at each node is solved by a bundle method (with an aggregated master problem). The second contribution solves the arc-based model at each node by dynamic generation of both multicommodity flow variables and strong linking inequalities.

Connections with Heuristic Methods

Slope scaling heuristics for the *FCTP* are presented in Kim and Pardalos (1999). The approach is generalized to the *MCFND* in Eksioglu et al. (2002); Crainic et al. (2004). A variant of slope scaling, called capacity scaling, is proposed for the *MCFND* in Katayama et al. (2009). Embedding slope scaling within a Lagrangian heuristic can be found in Gendron and Gouveia (2017), for a piecewise linear multicommodity network design problem, and in Akhavan Kazemzadeh et al. (2021), for the *MCFND*. Using slope scaling heuristics to provide cuts to the Benders master problem for the *MCFND* can be found in Costa et al. (2012). For the *MCFND*, an algorithm that alternates between the solution of the LP relaxation and a restricted problem derived from that solution, as described in Sects. 9.3 and 9.4, is presented in Gendron et al. (2018). At each step, a combinatorial Benders cut is added to the LP relaxation to identify a new solution. More sophisticated algorithms along the same lines, based on column-and-row generation (adding strong linking and cover inequalities to a path-based model), are proposed in Hewitt et al. (2010) for the *MCFND* and in Hewitt et al. (2013) for the unsplittable variant of the *MCFND*. The MILP heuristic based on a population of elite solutions, which we refer to at the end of Sect. 9, is known as "polishing" and is due to Rothberg (2007).

Parallel Algorithms
The classification of parallel B&B algorithms presented in Sect. 10 is due to Gendron and Crainic (1994b). The master-slave, collegial and hybrid parallelization strategies are implemented and tested on a multicommodity fixed-charge location problem in Bourbeau et al. (2000). The design of parallel B&B algorithms is an ongoing topic of research, see, for instance (Ralphs et al. 2003; Carvajal et al. 2014; Eckstein et al. 2015). Parallel computing has been heavily used to solve network design problems, but mostly in the context of advanced heuristic methods, although the exact algorithm of Hewitt et al. (2013) is parallelized. More details on parallel heuristic algorithms are given in Chap. 4.

12 Conclusions and Perspectives

In this chapter, we have studied exact algorithms for solving fixed-charge network design problems. A recurring theme in all the sections of this chapter is the importance of exploiting structure to develop efficient algorithms. For instance, we have seen that, for the (single-commodity) *FCTP*, relatively straightforward B&B algorithms can be used, but that they benefit from the integration of Lagrangian-based filtering by reduced costs, often called "penalties" in the literature, see, e.g., Bell et al. (1999). For large-scale instances of the *FCTP*, new models that can be derived from Dantzig–Wolfe reformulations associated with the Lagrangian decomposition (variable splitting) technique, presented in Sect. 2.4, have been shown to be very effective (Mingozzi and Roberti 2018). Such models are solved through column generation and are strengthened with the addition of cuts, yielding advanced branch-price-and-cut (BP&C) implementations.

For the more difficult (multicommodity) *MCFND*, researchers have identified effective and efficient Lagrangian relaxation approaches. So far, with the exception of the B&B algorithm based on subgradient optimization, presented in Sect. 5, most of these relaxations have been exploited in the context of deriving heuristics (Akhavan Kazemzadeh et al. 2021). These heuristics are clearly essential ingredients in the quest for optimal solutions, but more significant effort must be deployed to develop exact methods that can solve very large-scale instances. The development of BP&C algorithms and their parallel implementations are promising avenues of research in this direction.

It is noteworthy that state-of-the-art MILP solvers can nowadays perform advanced algorithmic techniques that very few people would have think possible not long ago. For instance, the ability of the solvers to detect subproblem structures and to generate cuts has significantly improved in the last 25 years. Solvers can not only generate cover and flow cover inequalities, presented in Sect. 4, but they are also able to exploit the multicommodity structures present in many network design problems (Achterberg and Raack 2010). Another significant development in recent years is the integration in MILP solvers of advanced decomposition methods, in particular Benders decomposition. This has enabled researchers to take advantage of general-purpose techniques, such as Pareto-optimal cuts and stabilization, now implemented in the MILP solver, and to focus on the particular aspects of the decomposition that exploits the structure of the problem they wish to solve. Recent examples of this line of research can be found in Fischetti et al. (2016); Zetina et al. (2019). Following this trend, we encourage researchers to see MILP solvers as allies, rather than adversaries, in the quest for optimal solutions to very large-scale network design problem instances. Indeed, through the addition of cuts, MILP solvers already exploit a lot of the structure present in network design models and can be used to find optimal solutions to (smaller) subproblems that exhibit multicommodity flow or facility location structures (Akhavan Kazemzadeh et al. 2021).

References

Achterberg, T., Koch, T., & Martin, A. (2005). Branching rules revisitied. *Operations Research Letters, 33*, 42–54.

Achterberg, T., & Raack, C. (2010). The MCF-separator: Detecting and exploiting multi-commodity flow structures in MIPs. *Mathematical Programming Computation, 2*, 125–165.

Agarwal, Y., & Aneja, Y. (2012). Fixed-charge transportation problem: Facets of the projection polyhedron. *Operations Research, 60*(3), 638–654.

Agarwal, Y. K., & Aneja, Y.P. (2017). Fixed charge multicommodity network design using *p*-Partition facets. *European Journal of Operational Research, 258*, 124–135.

Akhavan Kazemzadeh, M. R., Bektas, T., Crainic, T. G., Frangioni, A., Gendron, B., & Gorgone, E. (2021). Node-based Lagrangian relaxations for multicommodity capacitated fixed-charge network design. *Discrete Applied Mathematics*, in press http://dx.doi.org/10.1016/j.dam.2020.12.024.

Applegate, D., Bixby, R. E., Chvátal, V., & Cook, W. (1995). Finding cuts in the TSP. Technical Report 95-05, DIMACS.

Atamtürk, A. (2001). Flow pack facets for the single node fixed charge flow polytope. *Operations Research Letters, 29*, 107–114.

Atamtürk, A. (2005). Cover and pack inequalities for (mixed) integer programming. *Annals of Operations Research, 139*, 21–38.

Atamtürk, A., Gómez, A., & Küçükyavuz, S. (2016a). Three-partition flow cover inequalities for constant capacity fixed-charge network flow problems. *Networks, 67*, 299–315.

Atamtürk, A., Küçükyavuz, S., & Tezel, B. (2016b). Path cover and path pack inequalities for the capacitated fixed-charge network flow problem. *SIAM Journal on Optimization, 27*(3), 1943–1976.

Atamtürk, A., & Savelsbergh, M. W. P. (2005). Integer-programming software systems. *Annals of Operations Research, 140*, 67–124.

Baena, D., Castro, J., & Frangioni, A. (2020). Stabilized Benders methods for large-scale combinatorial optimization, with application to data privacy. *Management Science, 66*(7), 3051–3068.

Balakrishnan, A., Magnanti, T. L., & Wong, R. (1989). A dual-ascent procedure for large-scale uncapacitated network design. *Operations Research, 37*(5), 716–740.

Balas, E. (1975). Facets of the knapsack polytope. *Mathematical Programming, 8*, 146–164.

Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W. P., & Vance, P. H. (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations Research, 46*, 316–329.

Barr, R. S., Glover, F., & Klingman, D. (1981). A new optimization method for large scale fixed charge transportation problems. *Operations Research, 29*(3), 448–463.

Bell, G. J., Lamar, B. W., & Wallace, C. A. (1999). Capacity improvement, penalties, and the fixed charge transportation problem. *Naval Research Logistics, 46*, 341–355.

Ben Amor, H. M. T., Desrosiers, J., & Frangioni, A. (2009). On the choice of explicit stabilizing terms in column generation. *Discrete Applied Mathematics, 157*, 1167–1184.

Benders, J. F. (1962) Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik, 4*, 238–252.

Benichou, M., Gauthier, J. M., Girodet, P., Hentges, G., Ribiere, G., & Vincent, O. (1971). Experiments in mixed-integer programming. *Mathematical Programming, 1*, 76–94.

Bixby, R. E., & Rotberg, E. (2007). Progress in computational mixed integer programming—a look back from the other side of the tipping point. *Annals of Operations Research, 149*, 37–41.

Bonami, P., Salvagnin, D., & Tramontani, A. (2020). Implementing automatic Benders decomposition in a modern MIP solver. In D. Bienstock, G. Zambelli (Eds.) *Integer programming and combinatorial optimization—IPCO 2020*. Lecture notes in computer science, vol. 12125 (pp. 78–90).

Bourbeau, B., Crainic, T. G., & Gendron, B. (2000). Branch-and-bound parallelization strategies applied to a depot location and container fleet management problem. *Parallel Computing, 26*, 27–46.

Cabot, A. V., & Erenguc, S. S. (1984). Some branch-and-bound procedures for fixed-cost transportation problems. *Naval Research Logistics, 31*, 145–154.

Cabot, A. V., & Erenguc, S. S. (1986). Improved penalties for fixed cost linear programs using Lagrangian relaxation. *Management Science, 32*, 856–869.

Carvajal, R., Ahmed, S., Nemhauser, G., Furman, K., Goel, V., & Shao, Y. (2014). Using diversification, communication and parallelism to solve mixed-integer linear programs. *Operations Research Letters, 42*, 186–189.

Chouman, M., Crainic, T. G., & Gendron, B. (2017). Commodity representations and cut-set-based inequalities for multicommodity capacitated fixed-charge network design. *Transportation Science, 51*(2), 650–667.

Chouman, M., Crainic, T. G., & Gendron, B. (2018). The impact of filtering in a branch-and-cut algorithn for multicommodity capacitated fixed-charge network design. *EURO Journal of Computational Optimization, 6*, 143–184.

Cook, W. (2012). Markowitz and Manne + Eastman + Land and Doig = Branch and Bound. In M. Grötschel (Ed.) *Optimization Stories, Documenta Mathematica, extra volume* (pp. 227–238)

Costa, A. M. (2005). A survey on Benders decomposition applied to fixed-charge network design problems. *Computers and Operations Research, 32*, 1429–1450.

Costa, A., Cordeau, J. F., & Gendron, B. (2009). Benders, metric and cutset inequalities for multicommodity capacitated network design. *Computational Optimization and Applications, 42*, 371–392.

Costa, A., Cordeau, J. F., Gendron, B., & Laporte, G. (2012). Accelerating Benders decomposition with heuristic master problem solutions. *Pesquisa Operacional, 32*(1), 3–19.

Crainic, T. G., Frangioni, A., & Gendron, B. (2001). Bundle-based relaxation methods for multicommodity capacitated fixed charge network design. *Discrete Applied Mathematics, 112*, 73–99.

Crainic, T. G., Gendron, B., & Hernu, G. (2004). A slope scaling/lagrangean perturbation heuristic with long-term memory for multicommodity capacitated fixed-charge network design. *Journal of Heuristics, 10*, 525–545.

Dantzig, G. B., Fulkerson, D. R., & Johnson, S. M. (1954). Solution of a large scale traveling salesman problem. Technical Report P-510, Santa Monica: RAND corporation.

Dantzig, G. B., & Wolfe, P. (1960). Decomposition principle for linear programs. *Operations Research, 8*(1), 101–111.

Desrosiers, J., Soumis, F., & Desrochers, M. (1984). Routing with time windows by column generation. *Networks, 14*, 545–565.

Eastman, W. L. (1958). *Linear programming with pattern constraints*. Ph.D. Thesis, Cambridge: Department of Economics, Harvard University.

Eckstein, J., Hart, W. E., & Phililips, C. A. (2015). PEBBL: An object-oriented framework for scalable parallel branch and bound. *Mathematical Programming Computation, 7*, 429–469.

Eksioglu, S. D., Pardalos, P. M., & Romeijn, H. E. (2002). A dynamic slope scaling procedure for the fixed-charge cost multi-commodity network flow problem. In P. M. Pardalos, V. K. Tsitsiringos (Eds.) *Financial engineering, E-commerce and supply chain*. Applied Optimization, vol. 70 (pp. 247–270). Berlin: Springer.

Fisher, M. L. (2004). The Lagrangian relaxation method for solving integer programming problems. *Management Science, 50*(12), 1861–1871.

Fischetti, M., Ljubić, I., & Sinnl, M. (2016). Redesigning Benders decomposition for large-scale facility location. *Management Science, 63*, 2146–2162.

Ford, L. R., & Fulkerson, D. R. (1958). A suggested computation for maximal multicommodity network flows. *Management Science, 5*, 97–101.

Fortz, B., & Poss, M. (2009). An improved Benders decomposition applied to a multi-layer network design problem. *Operations Research Letters, 37*(5), 359–364.

Frangioni, A. (2005). About Lagrangian methods in integer optimization. *Annals of Operations Research, 139*, 163–193.

Frangioni, A., & Gorgone, E. (2014). Bundle methods for sum-functions with “easy" components: Applications to multicommodity network design. *Mathematical Programming A, 145*, 133–161.

Frangioni, A., Gendron, B., & Gorgone, E. (2017). On the computational efficiency of subgradient methods: A case study with Lagrangian bounds. *Mathematical Programming Computation, 9*, 573–604.

Frangioni, A., Gendron, B., & Gorgone, E. (2020) Quasi-separable Dantzig–Wolfe reformulations for network design. In M. Baïou, B. Gendron, O. Günlük, A. R. Mahjoub (Eds.) *Combinatorial optimization—ISCO 2020*. Lecture Notes in Computer Science, vol. 12176 (pp. 227–236)

Gendron, B. (2011). Decomposition methods for network design. *Procedia Social and Behavioral Sciences, 20*, 31–37.

Gendron, B. (2019). Revisiting Lagrangian relaxation for network design. *Discrete Applied Mathematics, 261*, 203–218.

Gendron, B., & Crainic, T. G. (1994a). Relaxations for multicommodity capacitated network design problems. *Publication CRT-965, Centre for Research on Transportation*. Montreal: University of Montreal.

Gendron, B., & Crainic, T. G. (1994b). Parallel branch-and-bound algorithms: Survey and synthesis. *Operations Research, 42*(6), 1042–1066.

Gendron, B., Crainic, T. G., & Frangioni, A. (1999). Multicommodity capacitated network design. In B. Sansò, P. Soriano, (Eds.) *Telecommunications network planning*. Berlin: Springer (pp. 1–19).

Gendron, B., & Gouveia, L. (2017). Reformulations by discretization for piecewise linear integer multicommodity network flow problems. *Transportation Science, 51*(2), 629–649.

Gendron, B., Hanafi, S., & Todosijevic, R. (2018). Matheuristics based on iterative linear programming and slope scaling for multicommodity capacitated fixed charge network design. *European Journal of Operational Research, 268*, 70–81.

Gendron, B., & Larose, M. (2014). Branch-and-price-and-cut for large-scale multicommodity capacitated fixed-charge network design. *EURO Journal on Computational Optimization, 2*, 55–75.

Geoffrion, A. M. (1974). Lagrangean relaxation for integer programming. *Mathematical Programming Studies, 2*, 82–114.

Geoffrion, A. M., & Graves, G. W. (1974). Multicommodity distribution system design by Benders decomposition. *Management Science, 20*(5), 822–844.

Gilmore, P. C., & Gomory, R. E. (1961). A linear programming approach to the cutting-stock problem. *Operations Research, 9*, 849–859.

Gilmore, P. C., & Gomory, R. E. (1963). A linear programming approach to the cutting-stock problem—Part II. *Operations Research, 11*, 863–888.

Gomory, R. E. (1958). Outline of an algorithm for integer solutions to linear programs. *Bulletin of the American Mathematical Society, 64*, 275–278.

Görtz, S., & Klose, A. (2009). Analysis of some greedy algorithms for the single-sink fixed-charge transportation problem. *Journal of Heuristics, 15*, 331–349.

Görtz, S., & Klose, A. (2012). A simple but usually fast branch-and-bound algorithm for the capacitated facility location problem. *INFORMS Journal on Computing, 24*(4), 597–610.

Göthe-Lundgren, M., & Larsson, T. (1994). A set covering reformulation of the pure fixed charge transportation problem. *Discrete Applied Mathematics, 48*, 245–259.

Gu, Z., Nemhauser, G. L., & Savelsbergh, M. W. P. (1998). Lifted cover inequalities for 0-1 integer programs: Computation. *INFORMS Journal on Computing, 10*, 427–437.

Gu, Z., Nemhauser, G. L., & Savelsbergh, M. W. P. (1999a). Lifted cover inequalities for 0-1 integer programs: Complexity. *INFORMS Journal on Computing, 11*, 117–123.

Gu, Z., Nemhauser, G. L., & Savelsbergh, M. W. P. (1999b). Lifted flow cover inequalities for mixed 0-1 integer programs. *Mathematical Programming, 85*, 439–467.

Gu, Z., Nemhauser, G. L., & Savelsbergh, M. W. P. (2000). Sequence independent lifting in mixed integer programming. *Journal of Combinatorial Optimization, 4*, 109–129.

Hammer, P. L., Johnson, E. L., & Peled, U. N. (1975). Facets of regular 0-1 polytopes. *Mathematical Programming, 8*, 179–206.

Hewitt, M., Nemhauser, G. L., & Savelsbergh, M. W. P. (2010). Combining exact and heuristic approaches for the capacitated fixed-charge network flow problem. *INFORMS Journal on Computing, 22*(2), 314–325.

Hewitt, M., Nemhauser, G. L., & Savelsbergh, M. W. P. (2013). Branch-and-price guided search for integer programs with an application to the multicommodity fixed-charge network flow problem. *INFORMS Journal on Computing, 25*(2), 302–316.

Holmberg, K., & Hellstrand, J. (1998). Solving the uncapcitated network design problem by a Lagrangian heuristic and branch-and-bound. *Operations Research, 46*(2), 247–259.

Holmberg, K., & Yuan, D. (2000). A Lagrangian heuristic based branch-and-bound approach for the capacitated network design problem. *Operations Research, 48*(3), 461–481.

Katayama, N., Chen, M., & Kubo, M. (2009). A capacity scaling heuristic for the multicommodity capacitated network design problem. *Journal of Computational and Applied Mathematics, 232*(1), 90–101.

Kennington, J., & Unger, E. (1976). A branch-and-bound algorithm for the fixed-charge transportation problem. *Management Science, 22*(10), 1116–1126.

Kim, D., & Pardalos, P. M. (1999). A solution approach to the fixed charge network flow problem using a dynamic slope scaling procedure. *Operations Research Letters, 24*, 195–203.

Kliewer, G., & Timajev, L. (2005). Relax-and-cut for capacitated network design. In G. S. Brodal, S. Leonardi, (Eds.) *Algorithms—ESA 2005*. Lecture Notes in Computer Science, vol. 3669 (pp. 47–58).

Klose, A. (2008). Algorithms for solving the single-sink fixed-charge transportation problem. *Computers and Operations Research, 35*(6), 2079–2092.

Klose, A., & Görtz, S. (2007). A branch-and-price algorithm for the capacitated facility location problem. *European Journal of Operational Research, 179*, 1109–1125.

Lamar, B. W., & Wallace, C. A. (1997). Revised-modified penalties for fixed charge transportation problems. *Management Science, 43*(10), 1431–1436.

Land, A. H., & Doig, A. G. (1960). An automatic method of solving discrete programming problems. *Econometrica, 28*, 497–520.

Letchford, A. N., & Souli, G. (2019). New valid inequalities for the fixed-charge and single-node flow polytopes. *Operations Research Letters, 47*, 353–357.

Little, J. D. C., Murty, K. G., Sweeney, D. W., & Karel, C. (1963). An algorithm for the traveling salesman problem. *Operations Research, 11*, 972–989.

Ljubić, I., Putz, P., & Salazar-González, J. J. (2012). Exact approaches to the single-source network loading problem. *Networks, 59*(1), 89–106.

Magnanti, T. L., & Wong, R. T. (1981). Accelerating Benders decomposition: algorithmic enhancement and model selection criteria. *Operations Research, 29*(3), 464–484.

Magnanti, T. L., & Wong, R. T. (1984). Network design and transportation planning: Models and algorithms. *Transportation Science, 18*(1), 1–55.

Magnanti, T. L., Mireault, P., & Wong, R. T. (1986). Tailoring Benders decomposition for uncapacitated network design. *Mathematical Programming Studies, 26*, 112–154.

Manne, A. S., & Markowitz, H. M. (1957). On the solution of discrete programming problems. *Econometrica, 25*, 84–110.

McDaniel, D., & Devine, M. (1977). A modified Benders' partitioning algorithm for mixed integer programming. *Management Science, 24*(3), 312–319.

Mingozzi, A., & Roberti, R. (2018). An exact algorithm for the fixed charge transportation problem based on matching source and sink patterns. *Transportation Science, 52*(2), 229–238.

Minoux, M. (1989). Network synthesis and optimum network design problems: Models, solution methods and applications. *Networks, 19*, 313–360.

Nemhauser, G. L., & Wolsey, L. A. (1988). *Integer and Combinatorial Optimization*. New York: Wiley.

Naoum-Sawaya, J., & Elhedhli, S. (2013). An interior-point Benders based branch-and-cut algorithm for mixed integer programs. *Annals of Operations Research, 210*, 33–55.

Ortega, F., & Wolsey, L. A. (2003). A branch-and-cut algorithm for the single-commodity, uncapacitated, fixed-charge network flow problem. *Networks, 41*(3), 143–158.

Padberg, M. W., & Rinaldi, G. (1987). Optimization of a 532-city symmetric traveling salesman problem by branch and cut. *Operations Research Letters, 6*, 1–7.

Padberg, M. W., Van Roy, T. J., & Wolsey, L. A. (1985). Valid linear inequalities for fixed charge problems. *Operations Research, 33*, 842–861.

Palekar, U. S., Karwan, M. H., & Zionts, S. (1990). Branch-and-bound method for the fixed charge transportation problem. *Management Science, 36*(9), 1092–1105.

Rahmaniani, R., Crainic, T. G., Gendreau, M., & Rei, W. (2017). The Benders decomposition algorithm: A literature review. *European Journal of Operational Research, 259*, 801–817.

Rahmaniani, R., Crainic, T. G., Gendreau, M., & Rei, W. (2018). Accelerating the Benders decomposition method: Application to stochastic network design problems. *SIAM Journal on Optimization, 28*(1), 875–903.

Ralphs, T. K., Ladányi, L., & Saltzman, M. J. (2003). Parallel branch, cut, and price for large-scale discrete optimization. *Mathematical Programming B, 98*, 253–280.

Roberti, R., Bartolini, E., & Mingozzi, A. (2015). The fixed charge transportation problem: An exact algorithm based on a new integer programming formulation. *Management Science, 61*(6), 1275–1291.

Rossi, F., van Beek, P., & Walsh, T. (2006). *Handbook of Constraint Programming*. Amsterdam: Elsevier.

Rothberg, E. (2007). An evolutionary algorithm for polishing mixed integer programming solutions. *INFORMS Journal on Computing, 19*(4), 534–541.

Savelsbergh, M. W. P. (1994). Preprocessing and probing techniques in mixed integer programming. *ORSA Journal on Computing, 6*(4), 445–454.

Sellmann, M., Kliewer, G., & Koberstein, A. (2002). Lagrangian cardinality cuts and variable fixing for capacitated network design. In R. Möhring, & R. Raman (Eds.) *Algorithms—ESA 2002*, Lecture Notes in Computer Science, vol. 2461 (pp. 845–858).

Sridhar, V., & Park, J. S. (2000). Benders-and-cut algorithm for fixed-charge capacitated network design problem. *European Journal of Operational Research, 125*, 622–632.

Stallaert, J. I. A. (1997). The complementary class of generalized flow cover inequalities. *Discrete Applied Mathematics, 77*, 73–80.

Van Roy, T. J., & Wolsey, L. A. (1987). Solving mixed integer programming problems using automatic reformulation. *Operations Research, 35*, 45–57.

Wolsey, L. A. (1975). Faces of linear inequalities in 0-1 variables. *Mathematical Programming, 8*, 165–178.

Wolsey, L. A. (1998) *Integer Programming*. New York: Wiley.

Wolsey, L. A. (2003). Strong formulations for mixed integer programs: Valid inequalities and extended formulations. *Mathematical Programming B, 97*, 423–447.

Zetina, C. A., Contreras, I., & Cordeau, J. F. (2019). Exact algorithms based on Benders decomposition for multicommodity uncapacitated fixed-charge network design. *Computers and Operations Research, 111*, 311–324.

Zhao, Y., Larsson, T., Rönnberg, E., & Pardalos, P. M. (2018). The fixed charge transportation problem: A strong formulation based on Lagrangian decomposition and column generation. *Journal of Global Optimization, 72*, 517–538.

Chapter 4
Heuristics and Metaheuristics for Fixed-Charge Network Design

Teodor Gabriel Crainic and Michel Gendreau

1 Introduction

While the methods based on exact solution principles described in Chap. 3 provide means to solve network design problem instances that have become of significant size over the years, it is important to realize that they are not the only way by which one can address network design problems. Actually, over the last 60 years or so, various approaches have been proposed to derive approximate solutions for various types of network design problems. These approaches have been used mostly to deal with the larger instances that are typically encountered in the context of real-life applications, which cannot be addressed by exact methods. In some cases, constraints on the time available for deriving good feasible solutions to complex problems has been a strong incentive for resorting to heuristics and metaheuristics.

Heuristic and metaheuristic approaches differ from the mathematical-programming based methods of Chap. 3 in many ways. A key difference is that they are based on the exploration of a *search space*, which is often quite different from the notion of feasible space of exact approaches. We will examine the notion of search space more thoroughly in the next section, as well as other important concepts.

Approximate approaches to network design problems can be broken down into three large classes: first, so-called *classical heuristics*, which rely on fairly simple

T. G. Crainic
CIRRELT and AOTI, Université du Québec à Montréal, Montréal, QC, Canada
e-mail: TeodorGabriel.Crainic@cirrelt.net

M. Gendreau (✉)
CIRRELT and MAGI, Polytechnique Montréal, Montréal, QC, Canada
e-mail: Michel.Gendreau@cirrelt.net

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_4

rules for building and improving tentative solutions; second, *metaheuristics*, which are methods that rely on sophisticated search strategies to derive very good (often near-optimal) solutions to the problem at hand; and third, *matheuristics*, which combine algorithmic components from metaheuristics with procedures derived from exact methods applied to the model formulation. In this chapter, we will examine these three classes, as well as parallel methaheuristics, which are methods that leverage the power of parallel computing to find better approximate solutions.

This chapter is organized as follows. In Sect. 2, we first present a number of basic concepts that are central to the way in which heuristics and metaheuristics tackle combinatorial problems. Section 3 is devoted to the more traditional heuristics, while the two following sections are devoted to the two main classes of metaheuristics, i.e., *neighborhood-based* metaheuristics and *population-based* metaheuristics. The first class refers to methods that follow a trajectory of solutions, moving at each step from a current tentative solution to a different neighbor. Population-based methods rely on the the application of various operations on a population of several possible solutions to the problem at hand with the objective of identifying an interesting one at the end of the procedure; in several of these methods, the operations considered mimick processes that are observed in the evolution of species. Some important applications of matheuristics to network design problem are then presented in Sect. 6. Section 7 follows and presents solution approaches that use parallel computing; while these methods often combine ideas and elements from the previous sections, several go much further and are among the best meta- or matheuristics for network design problems. Bibliographical notes are provided in Sect. 8. We summarize the main conclusions of this chapter and provide some research perspectives in Sect. 9.

In Sects. 3–5, we review in different subsections several methods that fall under the general heading of the section. For each of these methods, we first briefly recall the general principles of the method. This is followed in most cases by a presentation of the application of the method to one or several of the network design problems described in Chap. 2. In some cases, when we are not aware of any direct application of a method to network design problems in transportation and logistics, the presentation of relevant applications is deferred to Sects. 6 or 8.

2 Basic Concepts

We first examine the notion of *search space*, which is central in heuristics and metaheuristics. Furthermore, the exploration of this search space is conducted normally by considering *neighborhoods* or *populations* of solutions. We review these basic concepts, as well as a few others, in the following.

2.1 Search Space

The *search space* that will be explored by the search is a key component of the methods examined in this chapter. This notion refers to the space of solutions that can be considered through the exploration. While it may seem natural to equate the search space with the set of feasible solutions to a problem, there are many situations where this is not attractive.

To better understand this, consider the single-commodity fixed-charge transportation problem (FCTP) and the formulation proposed in Sect. 2.3 of Chap. 2. This problem is defined on a network $\mathcal{G} = \{\mathcal{N}, \mathcal{A}\}$, the set $\mathcal{A}$ encompassing the possible design arcs, between the origin (source) and destination (sink) nodes of set $\mathcal{N} = \mathcal{N}^o \cup \mathcal{N}^d$, among which the selection is to be made. For simplicity sake, we assume in the following that the underlying bipartite graph is complete. Each design arc is characterized by a *fixed* selection cost, f_{ij}, a *unit* flow *transportation cost*, c_{ij}, and a *capacity*, u_{ij}, limiting the volume of commodity one may assign to the arc. The objective is to fulfill at minimal total cost, computed as the sum on the total fixed and transportation costs, the demand for transportation between the origins, each with *supply* (*availability*) $w_i > 0$ of the given commodity, and destinations, each with a *demand* (*request*) $w_i < 0$ of the same commodity.

Two types of decision variables are defined: x_{ij} continuous variables that refer to the amount of flow going from vertex i to vertex j, and y_{ij} binary variables that indicate whether arc (i, j) is used in the solution or not. Thus, one could imagine that the search space would consist of feasible pairs of $(\mathbf{x}, \mathbf{y})$ vectors. However, one can do much better because of the close relationships between the $\mathbf{x}$ and $\mathbf{y}$ vectors that make up any optimal solution. In particular, in an optimal solution, one will have $y_{ij} = 1$ *iff* $x_{ij} > 0$. One could thus define the search space with respect to the $\mathbf{x}$ variables only. An alternate approach, which is much more easily implemented, defines the search space with respect to the $\mathbf{y}$ variables. For any given value $\bar{\mathbf{y}}$ of the $\mathbf{y}$ vector, a complete solution can be recovered by solving an auxiliary transportation problem:

$$\text{Minimize} \sum_{(i,j)\in\mathcal{A}} c_{ij}x_{ij} \tag{4.1}$$

$$\text{Subject to} \quad \textstyle\sum_{j\in\mathcal{N}_i^+} x_{ij} = |w_i|, \quad \forall\, i \in \mathcal{N}^o, \tag{4.2}$$

$$\textstyle\sum_{j\in\mathcal{N}_i^-} x_{ji} = |w_i|, \quad \forall\, i \in \mathcal{N}^d, \tag{4.3}$$

$$x_{ij} \leq u_{ij}\bar{y}_{ij}, \qquad \forall\, (i, j) \in \mathcal{A}, \tag{4.4}$$

$$x_{ij} \geq 0, \qquad \forall\, (i, j) \in \mathcal{A}. \tag{4.5}$$

where constraints (4.2) and (4.3) enforce the supply ($w_i, i \in \mathcal{N}^o$) and demand ($w_i, i \in \mathcal{N}^d$) conditions at origin and destination nodes, respectively, while constraint (4.4) limits the flow on each selected arc to its capacity, u_{ij}.

It is important to note that, in many methods, the search space is often not limited to feasible solutions. In fact, many extremely successful metaheuristics rely heavily on the possibility of considering infeasible solutions during the search, especially when dealing with very constrained problems. This is usually managed by adding penalties to the objective to account for violations of some constraints (see Gendreau et al. 1994, for a very successful use of these ideas).

2.2 Neighborhoods

Several of the methods that we discuss in this chapter, notably, Local Search improvement heuristics and neighborhood-based metaheuristics, rely heavily upon the concept of *move*, which refers to the transition from one solution in the search space to another by the application of some transformation. In many methods, these transformations are rather limited and give rise to solutions that are just slightly modified, but this is not always the case and moves implying major changes to the solutions were proposed for several methods.

The set of all the solutions that can be produced by applying an allowable move to a given solution S gives rise to the *neighborhood* $\mathscr{N}(S)$ of this solution S. The definition of the neighborhood structures (there could be one or several depending on the complexity of the method) that will be used in a search process is one of its central features. The definition of these neighborhood structures is highly dependent on the choices made for the search space, since neighborhoods allow us to go from one element in the search space to another (or, more correctly, define a set of other elements of the search space that one could reach in a single move).

Consider again the fixed-charge transportation problem and suppose that we have chosen to explore the space of the binary y variables. In this search space, a neighborhood structure that is commonly used is the so-called *add-drop neighborhood*, which associates to any given $\bar{\mathbf{y}}$ vector the set of all $\mathbf{y}$ vectors that differ from $\bar{\mathbf{y}}$ in only a single entry (either we *add* an arc (i', j') for which we had previously $\bar{y}_{i'j'} = 0$, or we drop an arc for which $\bar{y}_{i'j'} = 1$). Thus the add-drop neighborhood $N_{AD}(\bar{\mathbf{y}})$ is the set $\{\mathbf{y} | y_{ij} = \bar{y}_{ij}, (i, j) \in \mathscr{A} \setminus (i', j')$ and $y_{i'j'} = 1 - \bar{y}_{i'j'}, \forall (i', j') \in \mathscr{A}\}$.

An alternate neighborhood structure for the same search space is that of the *swap neighborhood*, which corresponds to changing simultaneously two entries in the $\bar{\mathbf{y}}$ vector: a null one that now takes the value 1 and a positive one that becomes 0.

Neighborhood-based methods usually require that, at each iteration, the solutions in the neighborhood $\mathscr{N}(S)$ of the current solution S be examined to find an attractive candidate solution for the next iteration. There are several ways of performing this exploration. A very common one consists in examining all the elements of $\mathscr{N}(S)$ and in selecting the one with the best objective function value. This is called *best-improvement* search. If neighborhoods are large, one could instead stop the procedure as soon as a solution that improves on S is found, which is *first-improvement* search. More sophisticated ways to perform the evaluation in the neighborhood $\mathscr{N}(S)$ have been proposed, mostly to reduce the computational burden of each iteration. These are often refereed to as *candidate-list strategies*.

2.3 Populations

As their name implies, population-based metaheuristics rely on the exploitation of populations (i.e., samples) of solutions of the problem at hand to explore a chosen search space. Usually, most of these solutions are feasible ones, but in some cases, infeasible solutions may be considered as well. Central to population-based methods is the process of constructing new solutions by combining elements or features of existing ones.

There is a wide range of population-based metaheuristics and the specific mechanisms that they use thus vary quite a lot, but any population-based metaheuristic must address a number of fundamental questions:

- How is the initial population created?
- At each iteration, how are chosen existing population members to create new ones? (*selection*)
- How are features from selected population members selected? (*crossover* or other combination procedure)
- What other modifications are performed on elements from the population? (*mutation* or *education*)
- How are less interesting members of the population deleted?
- When should the search process stop?

We will see in Sect. 5 the answers provided by different methods to these questions.

2.4 Evaluating the Performance of Heuristics and Metaheuristics

In general, the evaluation of the performance of heuristics and metaheuristics is a much more complex issue than for exact methods. Among other things, it is not just a matter of measuring how much computing (CPU) time a method requires to find the optimal solution to a problem. Furthermore, except in very special cases, there are very few theoretical results that one can derive in this area. The evaluation of heuristics and metaheuristics thus relies heavily on empirical computational experiments performed on suitable sets of benchmark instances. Such benchmarks should be reasonably representative of the types of actual instances that one would like to solve with the methods at hand.

One must also avoid the pitfall of using the same sets of instances to calibrate the parameters of the methods being tested and to derive performance measures.

If this is not done, one may end up in a situation where the parameter values are too well adapted to the instances at hand (i.e., this is a case of *over-fitting*) with the consequence that the performance observed on the benchmark instances would not generalize to other instances, thus leading to completely erroneous conclusions. Ideally, one would also like to have in benchmark instances that are small enough to be tackled by exact methods to allow for the estimation of the optimality gaps between heuristic solutions and actual optimal solutions.

Another issue that is far from trivial is deciding how much time or how many iterations to allow to heuristics, since many of these methods do not have obvious a priori stopping criteria. With additional time, the quality of solutions improves, but often in a very different fashion. Different methods thus display different *performance profiles* over time, which makes a direct comparison often rather difficult.

A factor that often complicates comparisons between various methods aimed at solving the same class of problems is the fact that different authors use different sets of test instances, which makes it almost impossible to derive any meaningful comparisons between the methods. We are not aware of any widely used benchmark for the fixed-charge transportation problem. The situation is different for the multicommodity capacitated fixed-charge network design problem (MCFND). Most authors who proposed solution methods for the MCFND have relied on the same set of benchmark instances to assess the performance of this method. This benchmark was proposed in Gendron and Crainic (1994), and it is made up of two sets of instances. The main set, which is called set **C**, is made up of 43 instances having between 20 and 100 nodes, between 100 and 700 arcs, from 10 to 400 commodities, different fixed to variable costs ratios, and loose or tight capacity constraints, providing the means to thoroughly assess the performance of algorithms on a wide range of problem features; this is the set used by most authors in their computational experiments. A second set, called set **R**, was created to study more systematically the impact of problem characteristics on algorithmic performance. It is made up of 18 basic networks, having from 10 to 20 nodes, 25 to 300 arcs, and 10 to 200 commodities; for each network, 9 instances are created by considering three levels of fixed cost and three levels of capacity tightness, thus yielding a total of 162 instances. In this chapter, we will refer to these sets of benchmark instances as sets **C** and **R** of the Gendron-Crainic benchmark.

3 Classical Heuristics

When dealing with large or difficult problems in combinatorial optimization, heuristic methods have been used since the beginnings of operations research. We refer to methods used since the early ages of operations research as *classical* ones. In general, one distinguishes between constructive and improvement heuristics, but these are often used in conjunction to provide better-quality solutions.

3.1 Constructive Heuristics

The main objective pursued in constructive heuristics is simply to obtain some feasible solution for the problem at hand, with the hope that this solution will be "good enough" for the intended usage, or that it can serve as initial solution for a more involved heuristic procedure. The archetypical example of a constructive heuristic is the so-called *greedy heuristic*, which builds a solution one element at the time, at the lowest cost possible (assuming that the problem is a minimization one).

To illustrate how such a method works, let us consider the fixed-charge transportation formulation recalled in Sect. 2 (see also Sect. 2.3 of Chap. 2). One way of applying a greedy approach for this problem is to compute the unit linearized cost $c'_{ij} = f_{ij}/u_{ij} + c_{ij}$ for all arcs $(i, j) \in \mathscr{A}$, and to select the arc with the minimum value. One then assigns a flow of value $x_{ij} = \min\{|w_i|, |w_j|\}$ to arc (i, j). The values for w_i and w_j are then decreased by x_{ij}, since the transportation requests for nodes i and j have been partially fulfilled. The process then goes through another iteration, without taking into account the arc (i, j), which can no longer be used, until all requests have been assigned. When the problem is defined on a complete bipartite graph and when demands are balanced, i.e., when $\sum_{i\in\mathscr{N}^o} w_i = \sum_{j\in\mathscr{N}^d} w_j$, this process leads to a feasible solution.

Greedy methods are fast but yield results of dubious quality in many cases, particularly for network design problems. It has been experimentally shown, for example, that the last arcs selected by the greedy heuristic above for the transportation problem are among the most costly ones in the network. A constructive method alleviating such shortcomings for network design makes use of the same unit linearized arc cost defined above, but is based on solving the linear programming relaxation of the problem with those costs, and selecting the arcs with positive flow in the optimal solution. More precisely, one first solves the minimum cost transportation problem (4.1)–(4.5) with $c_{ij} = c'_{ij}$. Then, $y_{ij} = 1$ if $x_{ij} > 0$ in the optimal solution, and 0, otherwise. This procedure has experimentally produced very good feasible solutions not only for the fixed-charge transportation problem, but also more broadly for the single and the multicommodity capacitated fixed-charge network design formulations.

3.2 Improvement Methods (Local Search)

Improvement methods are the natural complement to constructive heuristics, in the sense that these are methods that are meant to produce sequences of feasible solutions that improve on one another with respect to the objective function of the problem at hand. Most of these methods are based on the principles of *Local Search*, which iteratively applies "local" modifications to a so-called *current solution* in such a way that the modified solution improves the objective value at each step. This monotonicity condition ensures that the method will not cycle. It will eventually reach a solution such that no improving one can be determined by the application of the possible local modifications. Whenever this happens, the search has found a *local optimum* and terminates.

It must be emphasized that Local Search methods rely heavily rely on the concepts of *search space* and *neighborhoods* presented in Sect. 2. It must be noted, however, that traditional Local Search methods rarely consider infeasible solutions in their exploration process.

3.2.1 Basic Local Search

The basic principle of Local Search is simply to explore in an iterative fashion the selected search space for the problem at hand by performing at each iteration a move from the *current solution* S to an improving feasible one selected in its neighborhood $\mathscr{N}(S)$, until a local optimum is encountered. The improving solution chosen at each iteration can be selected according to best-improvement or first-improvement rules.

3.2.2 A Local Approach Search for the Fixed-Charge Transportation Problem

We now describe a fairly simple, yet effective, approach for the fixed-charge transportation problem. This Local Search method relies on the exploration of the extreme points of the polyhedron of the transportation problem defined by equations

$$\textstyle\sum_{j \in \mathscr{N}_i^+} x_{ij} = |w_i|, \quad \forall\, i \in \mathscr{N}^o, \tag{4.6}$$

$$\textstyle\sum_{j \in \mathscr{N}_i^-} x_{ji} = |w_i|, \quad \forall\, i \in \mathscr{N}^d, \tag{4.7}$$

$$x_{ij} \le u_{ij}, \qquad \forall\, (i, j) \in \mathscr{A}, \tag{4.8}$$

$$x_{ij} \ge 0, \qquad \forall\, (i, j) \in \mathscr{A}. \tag{4.9}$$

These extreme points correspond to feasible basic solutions of the transportation problem. It has been known since the 1950s (Hirsch and Dantzig 1954), that an optimal solution of the fixed-charge transportation problem can be found in one of the extreme points of this polyhedron. Thus, one can simply explore the search space defined by these extreme points (feasible bases of the system (4.6)–(4.9)). A natural way to do so is to consider *pivot* operations from one feasible basic solution to another, and use them to define a neighborhood structure on this search space. It is also interesting to note that, since bases of the system (4.6)–(4.9) correspond to spanning trees of the underlying bipartite graph, one can also interpret the search using this neighborhood as searching the space of the spanning trees of the bipartite graph, where adjacent spanning trees only differ in two edges.

A Local Search procedure for the fixed-charge transportation problem can thus start from any basic solution to (4.6)–(4.9), and proceed by performing pivots on this system. Let's note $\bar{\mathbf{x}}$ the vector of continuous variables corresponding to the current extreme point solution. The cost of this solution is then easily obtained as the sum of the variable $\sum_{(i,j)\in\mathscr{A}} c_{ij}x_{ij}$ and fixed costs f_{ij} of arcs (i, j) for which $x_{ij} > 0$.

This basic Local Search can be expected to rapidly run into a local optimum and thus to terminate prematurely. One way to improve its performance consists in considering, when running into a local optimum, an extended neighborhood that contains extreme points that are two pivots away from the current solution (i.e., solutions that are obtained by pivoting into the current basis a pair of non-basic variables). Obviously, the exploration of the set of extreme points that are two pivots away is significantly more expensive than just looking at adjacent extreme points, but it yields significantly better solutions. Defining several such neighborhoods, e.g., 1-pivot, 2-pivots, 3-pivots, . . . , and rules specifying when and how the search passes from one neighborhood to another leads to metaheuristics.

4 Neighborhood-Based Metaheuristics

As we already mentioned, neighborhood-based metaheuristics rely on following a trajectory of related solutions to the problem at hand; hence, they are also referred to as *trajectory-based methods*. In many ways, these methods can be seen as generalizations of the Local Search methods of the previous section, using the same basic concepts of search spaces and neighborhoods, with a few twists that we now explain.

All neighborhood-based metaheuristics rely on more sophisticated search strategies than Local Search, in particular by including more than one search heuristic. In this way, they perfectly illustrate the definition of metaheuristics (and matheuristics) as *heuristics guiding other heuristics*. Moreover, in the improvement methods of the previous section, one normally chooses the feasible neighbor with the best objective function value to become the new current solution and the search terminates whenever $\mathscr{N}(S)$ does not contain any improving feasible neighbor. This is not the case in neighborhood-based metaheuristics, which often use different rules for selecting the new current solution and do not terminate when they encounter local optima.

We first focus on Tabu Search methods, which have seen very successful applications to classical network design problems. We then present rapidly four other neighborhood-based metaheuristics, which have been integrated into hybrid methods or matheuristics, or applied to complex network design problems that are beyond the scope of this chapter: Simulated Annealing, Iterated Local Search, Greedy Randomized Adaptive Search Procedure, and Variable Neighborhood Search.

4.1 Tabu Search

Traditional Local Search methods are plagued by the fact that their exploration of the search space terminates whenever they run out of improving neighbors, i.e., whenever they run into a *local optimum* of the problem with respect to the chosen neighborhood structure. In many problems, this can lead to very poor heuristic solutions. The key ideas of Tabu Search is (1) to continue exploring the search space even when a local optimum is encountered, and (2) to *learn* during the exploration about the search space, the trajectory, and the search behavior (e.g., identifying variables or zones critical to good solutions). Continuing the exploration beyond a local optimum may, however, easily lead to cycling: after moving from a local optimum to one of its neighbors, the search could very well move back to the just-visited local optimum. To prevent this from happening, one must forbid some moves by declaring then *tabu* for a certain time length or number of iterations. In practice, one records in some *short-term memory* key information on the moves that were performed in the most recent iterations and declares tabu either the reverse moves or moves involving particular configurations of the solution attributes (e.g., paths for commodities recently involved in moves). Note that the tabu status may be lifted if the candidate solution is *improving* with respect to an *aspiration criterion*. The latter is generally defined as a minimum threshold by which the candidate solution must better than the local or global current best solution (e.g., the objective-function value of the candidate must be lower by at least $\alpha\%$ than the value of the current best), in order for the tabu status to be lifted.

Actually, Tabu Search goes well beyond simply exploiting short-term memory. Longer-term memories are used to implement two key ideas: *search intensification* and *search diversification*.

The idea behind search intensification is, from times to times, to interrupt the regular search process to explore more thoroughly portions of the search space in which good solutions, e.g., the best known solution, have been encountered. This more thorough search is often accomplished by switching to a different neighborhood structure: for instance, if one uses an add-drop neighborhood structure for regular search, then intensification could use a swap neighborhood.

Diversification can be seen as the complement of intensification. Here, the main objective is to insure that the search covers a wide portion of the search space, i.e., that several different possible solutions are examined. Diversification is often based on long-term memories, such as *frequency memory*, which records the number of times that some solution element, e.g., a design arc, has appeared in the current solution. The basic idea is to implement algorithmic mechanisms to “force” these less frequent elements into the current solution, thus redirecting the search to unexplored or little explored parts of the search space. Diversification has become a key component of most meta- and matheuristic strategies.

4.1.1 Tabu Search for the Fixed-Charge Transportation Problem

A natural way of applying Tabu Search to the fixed-charge transportation problem is by extending the ideas of Sect. 3.2.2 with respect to the choice of search space and neighborhood structure. Thus, one can implement a Tabu Search metaheuristic using the set of extreme points of the polyhedron defined by the system (4.6)–(4.9) as search space, and a neighborhood structure defined by pivot operations. Because of the more sophisticated search mechanisms available in Tabu Search, one can just consider 1-pivot moves and adjacent extreme points as neighbors of a given solution.

The search can be initiated from any feasible solution to (4.6)–(4.9). Since moves correspond to pivot operations, they can be defined in terms of flow (continuous) variables entering the basis, or equivalently, the arcs of the spanning tree to which this basis corresponds. In that context, tabu restrictions are defined with respect to the arcs of that spanning tree: (1) an arc leaving the spanning tree and becoming non-basic is prevented from reentering the basis for ξ_{in} iterations; (2) when an arc becomes basic, it must remain in the solution for at least ξ_{out} iterations. The values for these *tabu tenures* can be fixed throughout the search or be reset at some random values in a given interval during the search. These tabus prevent some solutions in the neighborhood of the current solution to be considered as candidates for the next move, unless these solutions are improving with respect to an aspiration criterion.

To reduce the computational burden, one can consider at each iteration only the moves involving non-basic arcs originating from a single origin. In the next iteration, the selected origin is moved to the next one. This *candidate list strategy* is disabled when all moves considered are tabu.

When the basic search process starts to stagnate (i.e., a large number of iterations have been performed without improving the best solution found), it is a good idea to consider intensifying the search in the hope of identifying better solutions. One way of inducing intensification is to reduce the impact of the fixed cost of arcs that have been present in the basis of the current solution for a large number of iterations. By reducing the contribution of the fixed cost of these arcs in the objective, these arcs become more attractive and thus more likely to be again present in the spanning tree. This reduction of fixed cost is applied until an overall better solution is found or for a certain number of moves.

Diversification can be encouraged in a fashion similar to the one used for intensification, but instead reducing the impact of the fixed cost of arcs that have seldom been present in the basis of the current solution. The length of this diversification phase can be fixed. Another option for diversification is to directly force into the spanning tree arcs that have been out of the basis for the longest time.

Because they do not stop at local optima, Tabu Search heuristics could, in theory, go on for ever. We thus need stopping criteria. In its simplest form, these could be a pre-defined number of iterations or, more usually, a number of iterations without improvement of the best solution found. In the context of a complex search strategy involving both intensification and diversification phases, it makes more sense to stop the search after a given number of such phases and return the best solution found.

4.1.2 Tabu Search for the Multicommodity Capacitated Fixed-Charge Network Design Problem

We now examine how Tabu Search can be used to tackle multicommodity capacitated fixed-charge network design (MCFND) formulations (Sect. 3, Chap. 3). We briefly recall the notation of the MCFND. The problem is defined on a network $\mathscr{G} = (\mathscr{N}, \mathscr{A})$ shared by several commodities represented by set $\mathscr{K}$. The demand w_i^k to satisfy for any commodity $k \in \mathscr{K}$ is defined at each node $i \in \mathscr{N}$ and, for simplicity, we assume that $\sum_{i \in \mathscr{N}} w_i^k = 0, k \in \mathscr{K}$, i.e., the demand is balanced for each commodity. Each potential design arc $(i, j) \in \mathscr{A}$ is characterized by its fixed cost f_{ij}, charged whenever the arc is included in the optimal design, its capacity u_{ij}, limiting the total flow of all commodities on the arc, and commodity-specific unit transportation costs $c_{ij}^k, k \in \mathscr{K}$. The MCFND minimizes the sum of the costs to select the arcs to be included in the design and the transportation costs to move the commodity flows, within the transportation capacities of the selected arcs. Using binary design variables $y_{ij}, (i, j) \in \mathscr{A}$, and continuous multicommodity flow variables $x_{ij}^k \geq 0, (i, j) \in \mathscr{A}, k \in \mathscr{K}$, the arc-based MCFND model is written as

$$\text{Minimize} \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + \sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}} c_{ij}^k x_{ij}^k \tag{4.10}$$

$$\text{Subject to} \quad \textstyle\sum_{j\in\mathscr{N}_i^+} x_{ij}^k - \sum_{j\in\mathscr{N}_i^-} x_{ji}^k = w_i^k, \quad \forall i \in \mathscr{N}, \forall k \in \mathscr{K}, \tag{4.11}$$

$$\textstyle\sum_{k\in\mathscr{K}} x_{ij}^k \leq u_{ij} y_{ij}, \qquad \forall (i, j) \in \mathscr{A}, \tag{4.12}$$

$$x_{ij}^k \geq 0, \qquad \forall (i, j) \in \mathscr{A}, \forall k \in \mathscr{K}, \tag{4.13}$$

$$y_{ij} \in \{0, 1\}, \qquad \forall (i, j) \in \mathscr{A}. \tag{4.14}$$

Tabu Search has been applied quite successfully to complex facility location problems, which display strong similarities with network design problems (in fact, many facility location problems can be easily transformed into equivalent network design problems). Several of these successful implementations were based on the exploration of the search space of binary location (i.e., design) variables using the add-drop and swap neighborhood structures described in Sect. 2.2. This type of approach would thus seem attractive for tackling the MCFND: once design variables have been set to given values given by the vector $\bar{\mathbf{y}}$, corresponding continuous flow variables can be easily recovered by solving the auxiliary *minimum cost multicommodity network flow* (MCMNF) problem:

$$\text{Minimize} \sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}} c_{ij}^k x_{ij}^k \tag{4.15}$$

$$\text{Subject to} \quad \textstyle\sum_{j\in\mathscr{N}_i^+} x_{ij}^k - \sum_{j\in\mathscr{N}_i^-} x_{ji}^k = w_i^k, \quad \forall i \in \mathscr{N}, \forall k \in \mathscr{K}, \tag{4.16}$$

$$\textstyle\sum_{k\in\mathcal{K}} x_{ij}^k \leq u_{ij}\bar{y}_{ij}, \qquad \forall (i,j) \in \mathcal{A}, \tag{4.17}$$

$$x_{ij}^k \geq 0, \qquad \forall (i,j) \in \mathcal{A}, \forall k \in \mathcal{K}, \tag{4.18}$$

$$y_{ij} \in \{0,1\}, \qquad \forall (i,j) \in \mathcal{A}. \tag{4.19}$$

Unfortunately, this fairly straightforward approach performs very poorly for network design, which presents a more complex network structure compared to location. Moves that close arcs may have a strong impact on the current solution, when closing paths that are used by many commodities between several origin-destination pairs, but moves that open new arcs are very often ineffective, since, in most cases, they will not lead directly to the creation of new paths for the commodities.

Pivot-Based Neighborhoods

The ideas underlying the solution methods presented in Sects. 3.2.2 and 4.1.1 offer more promising alternatives. Here again, the key idea is to consider as search space the set of extreme points of the MCMNF problem, with all arcs open, i.e., the problem obtained by deleting the $\bar{y}_{ij}$'s from the linear program (4.15)–(4.18). This search space can be explored by performing pivots from one feasible solution to another. The original c_{ij}^k arcs costs may be used or the linearized ones, i.e., $c_{ij}^{\prime k} = f_{ij}/u_{ij} + c_{ij}^k$, $(i,j) \in \mathcal{A}, k \in \mathcal{K}$. One generally expects to obtained better results by resorting to the linearized costs. As for the values of the design variables y_{ij}, they are easily recovered by setting $y_{ij} = 1$, when $\sum_{k\in\mathcal{K}} x_{ij}^k > 0$, and 0, otherwise.

At this point, it is important to recall that large MCMNF problems are solved more effectively using path-flow formulations and column generation techniques. This suggests switching to the path-based formulation and exploring the extreme points of the linear capacitated multicommodity flow subproblem of this formulation. Let $\mathcal{P}^k$ be the set of feasible paths in $\mathcal{G}$ for commodity $k \in \mathcal{K}$, let's define the decision variable h_p^k as the volume of commodity k moved on path $p \in \mathcal{P}^k$.

$$\text{Minimize} \sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} \sum_{p\in\mathcal{P}^k} c_{ij}^k \delta_{ij}^p h_p^k \tag{4.20}$$

$$\text{Subject to} \qquad \textstyle\sum_{p\in\mathcal{P}^k} h_p^k = d^k, \qquad \forall k \in \mathcal{K}, \tag{4.21}$$

$$\textstyle\sum_{k\in\mathcal{K}} \sum_{p\in\mathcal{P}^k} \delta_{ij}^p h_p^k \leq u_{ij}, \quad \forall (i,j) \in \mathcal{A}, \tag{4.22}$$

$$h_p^k \geq 0, \qquad \forall k \in \mathcal{K}, \forall p \in \mathcal{P}^k, \tag{4.23}$$

where δ_{ij}^p indicates whether (i.e., $\delta_{ij}^p = 1$) or not (i.e., $\delta_{ij}^p = 0$) arc $(i, j) \in \mathscr{A}$ belongs to path $p \in \cup_{k\in\mathscr{K}} \mathscr{P}^k$ (thus, $x_{ij}^k = \sum_{p\in\mathscr{P}^k} \delta_{ij}^p h_p^k, \ \forall (i, j) \in \mathscr{A}$).

In this formulation, pivots are made with respect to the path variables h_p^k (again, path costs may be the original or the linearized ones). A move thus corresponds to entering a currently unused path-flow variable $h_{p'}^{k'}$, $k' \in \mathscr{K}$, in the basis, which, in turn, forces out of the basis a path p'', of any commodity, for which the flow is driven to zero. Moves are evaluated by computing the difference in the total cost of the current solution, which is the sum of the variation of the total cost for the continuous variables plus the difference in the fixed cost of the binary design variables whose status has changed from open to closed or vice-versa. To prevent cycling, tabus are imposed on pivoting variables out of the basis, by assigning a random tabu tenure to each path entering the basis.

As in any path-based formulation, the number of paths can be huge. One must thus start the algorithm with a limited number of paths for each commodity and resort to column generation to identify additional ones. In this tabu search procedure, a column generation phase could be performed, for example, after a given number of moves without observing an improvement in the best known solution. As for search diversification, it could be performed after a set number of column generation phases without improvement of the objective. It is induced by closing a small subset of often-used arcs (as recorded in frequency memories) for some time.

A Better Approach: Using Cycle-Based Neighborhoods

While the Tabu Search described above proved quite effective, especially when dealing with problems with a small number of commodities, its performance is not totally satisfactory when one must solve network design instances with a large number of commodities, as one often encounters in practical applications. The main reason for this is that the moves considered in the extreme-point neighborhood only consider flow modifications of a single commodity at the time. This turns out to be too myopic for instances with several commodities.

One way to address this is to revisit approaches based on the exploration of the search space of design (i.e., binary) decision variables. As we have mentioned earlier, simple neighborhood structures for this search space, such as add-drop and swap, are clearly ineffective. One must thus look for new neighborhood structures that allow for a thorough and efficient search of the space of design variables. In this quest, one must first note that in order to significantly modify a solution in a network design problem, one must be able to open and close simultaneously sequences of arcs that make up subpaths. Furthermore, given a specific complete solution (i.e., including the values of the continuous variables), possible flow movements can only take place along cycles in the *residual graph* with respect to this solution.

The cycle-based neighborhood relies on the identification of cycles of given capacities in the residual graph. This is achieved by considering sets of candidate arcs as starting points for creating cycles; then, a labeling heuristic is used to identify low-cost cycles containing each of the candidate arcs. For each solution, once the cycle has been identified, the flow pattern is adjusted by solving exactly the associated MCMNF problem defined by (4.15)–(4.18), where $\bar{\mathbf{y}}$ is the design solution being evaluated. One of the most advantageous features of this new neighborhood structure is that it allows significant modifications of the current solution at each iteration, involving simultaneous flow changes for several commodities on several arcs.

To speed up the exploration of the cycle neighborhood, the flows of all commodities are aggregated in residual graphs. Because of that, the MCMNF problem may turn out to be infeasible, which requires the use of a suitable *restoration* procedure to retrieve a feasible solution.

The intrinsic power of the cycle neighborhood allows for the use of a fairly simple global search strategy. Hence, search intensification, implemented through flow modifications of single commodities, is invoked when very good solutions are obtained. With respect to search termination, very simple criteria, such as the total number of iterations or the global CPU time elapsed, can be used.

A comprehensive computational experimentation on the **C** instances of the Gendron-Crainic benchmark has shown that Tabu Search based on cycle neighborhoods is much more effective than the pivot-based approach described previously.

4.2 Other Neighborhood-Based Metaheuristics

In this section, we describe rapidly four families of metaheuristics, which, to the best of our knowledge, have not been yet applied directly to the solution of basic network design problems in the context of transportation and logistics. The main reason for presenting these families is that most of them are used as part of hybrid methods or matheuristics presented later in the chapter. In some cases, these methods have been applied to more complex variants of the problems that we discuss, as will be mentioned in the bibliographical notes of Sect. 8.

It is important to note that all these methods could be used, by themselves, to tackle the fixed-charge transportation or the multicommodity capacitated fixed-charge network design problems using some of the neighborhoods defined earlier.

4.2.1 Simulated Annealing

Simulated Annealing (SA) is one of the oldest and simplest metaheuristics. According to an analogy with the cooling of material in a heat bath, solutions to an optimization problem correspond to configurations of particles and the value of the objective function to the energy of the system for a given configuration. The trajectory followed by an SA procedure can be interpreted as a *controlled random walk* in the search space: at each step, a random solution is generated in the neighborhood of the current solution; if this new solution leads to an improved solution, the new solution is accepted and becomes the current one; if the tentative move deteriorates the objective, it is performed subject to a probabilistic acceptance criterion, which depends on the magnitude of the deterioration and a search control parameter, called the *temperature*. This temperature is slowly decreased with time, making non-improving moves more and more difficult to accept. Interestingly, under suitable assumptions on the number of temperature levels and the number of iterations per level, Markov chain theory can be used to show that SA should converge asymptotically to the global optimum of the problem at hand. In practice, however, these assumptions cannot be fulfilled within reasonable computation times. More relevant is the fact that, if one records the best solution observed during the random walk in the search space, a high-quality solution can often be identified within a reasonable computational effort.

Deterministic versions were also proposed under the names of *deterministic annealing* and *threshold acceptance*.These methods are similar in the sense that they are neighborhood search methods in which deterioration of the objective up to a *threshold* is accepted, the threshold decreasing as the algorithm progresses.

4.2.2 Iterated Local Search

Iterated Local Search (ILS) is a rather straightforward metaheuristic that, in its simplest form, combines basic Local Search with the concept of *perturbation*. Thus, ILS escapes from local optima by applying random perturbations to the current local optimum $s^\star$ to produce an intermediate solution s', to which Local Search is again applied. This leads to another local optimum solution $s'^\star$, which can be selected according to some *acceptance criterion*; if the solution $s'^\star$ does not satisfy the criterion, the search returns to $s^\star$, from which a new perturbed solution s'' is created. In many implementations, the acceptance criterion is very simple: $s'^\star$ is accepted as the new current solution only if its objective function value is better than the value of $s^\star$. Obviously, more sophisticated acceptance criteria may be used, and perturbation schemes may be devised to account for the past history of the search.

4.2.3 Greedy Randomized Adaptive Search Procedure

Greedy Randomized Adaptive Search Procedure (GRASP) is a fairly straightforward iterative metaheuristic that combines the simplicity of greedy solution heuristics with the power of randomized algorithms to tackle difficult combinatorial problems. Typically, each iteration of a GRASP metaheuristic is made up of two phases: a construction phase and a Local Search phase. In the former phase, a solution to the problem at hand is constructed by selecting one element at the time. The selection process follows greedy principles, but in randomized fashion: instead of selecting the most attractive element, a *restricted candidate list* (RCL) containing a subset of the most attractive elements is maintained and an element of RCL is chosen at random to be included in the solution. This procedure is repeated until a full solution is constructed. In the second phase, a suitable Local Search procedure is applied to the solution just constructed. It is important to note that the procedure can be applied a large number of times, since the choices made in the construction phase, being randomized, will lead to different solutions. One can thus consider a large number of possible solutions to the problem at hand. GRASP typically stops after performing a given number of iterations.

4.2.4 Variable Neighborhood Search

Variable Neighborhood Search (VNS) is a neighborhood-based metaheuristic that strives to perform a more effective exploration of the search space by exploiting several neighborhoods, but *not simultaneously*, i.e, different neighborhoods are considered in sequence. In its simpler version, which is called *Variable Neighborhood Descent (VND)*, neighborhoods are considered individually until one runs into a local optimum; when this happens one switches to the next neighborhood in the sequence. Several strategies can be applied with respect to the exploration of the neighborhoods. One may, e.g., restart the search using the first neighborhood or consider neighborhoods cyclically. The search terminates when none of the considered neighborhoods can lead to a better solution; this corresponds to having found a local optimum for each of these neighborhood structures. In general, if several neighborhoods are examined, the final solution should be an excellent one.

Other forms of VNS consider more intricate search mechanisms. One important mechanism is the use of a *shaking function* to generate points at random in some neighborhood before starting the exploration. Many of the advanced versions of VNS combine deterministic and stochastic changes of neighborhoods.

5 Population-Based Metaheuristics

As was already mentioned, population-based metaheuristics explore a suitably defined search space for the problem at hand by evolving a population of solutions through the applications of combination mechanisms and other procedures, which depend on the specific method considered. There is a wide range of population-based metaheuristics, but, in this chapter, we will focus on three families that have proved useful for solving network design problems: (1) Genetic and Evolutionary Algorithms, (2) Path Relinking procedures, and (3) Scatter Search.

5.1 Genetic Algorithms/Evolutionary Algorithms

Genetic Algorithms (GAs) are probably the first metaheuristic, going back to the mid-1970s, more than 10 years before the term metaheuristic was coined by Glover (1986). The basic GA is based on an analogy with the Darwinian evolution principles, the fundamental idea being to replicate the evolution of a population of *individuals*, which represent solutions to the problem at hand. Solutions and, indirectly, the search space are encoded as *chromosomes*. In early implementations of GA, chromosomes were simply bit strings, but natural encoding has become the norm in later implementations. The selection of individuals is performed according to their *fitness*, high fitness values corresponding to highly-desirable individuals. The fitness measure may be the objective function value associated with the individual (solution), or more complex measures accounting, e.g., for the distance from an "ideal" or the average measure for the population.

The basic GA mechanisms revolve around three key operators: *selection*, which identifies which individuals should be selected as *parents* for *reproduction*, i.e., to be used for creating new individuals; *crossover*, which creates one or two *offsprings* from the genetic material (i.e., features) of a pair of parents; and *mutation*, which randomly modifies the value of some *gene* (basic element of a chromosome). Selection is based on the fitness of individuals, often with stochastic elements, such as in *roulette-wheel selection*; the rationale is that one hopes that better offspring will be obtained from the best parents. Basic crossover operators, based on the bit-string representation of chromosomes, were originally proposed, but, over time, more sophisticated crossovers were designed for specific classes of problems.

Typically, a GA runs for a number of *generations* (iterations). In each generation, some individuals are selected to reproduce; then, new individuals are created from these by the application of the crossover operator; and mutation is applied to their offspring. The resulting new individuals are then added to the population, usually replacing less fit ones or all of them. It should be noted that some GAs simply create and introduce offspring in the population without considering generations.

A critical element in the successful application of GA is the need to maintain *diversity* inside the population: if individuals in the population become too similar, the method loses its ability to properly explore the search space. This realization has led to the introduction of the concept of *biased fitness*, which combines diversity considerations in the fitness function.

Over time, it has also become obvious that pure GA methods, implementing the traditional operators only, are usually not powerful enough to address hard combinatorial problems. This has led to the development of hybrid GAs that use some form of neighborhood search to improve offspring. In this case, one often talks of an *education* operator.

Evolutionary Algorithms (EAs) can be seen as an extension and generalization of GAs. Mutation plays a much more important role than recombination, mutation operators being often much more sophisticated than in GAs. Moreover, in many EAs, the representation of individuals includes strategy parameters in addition to the solution vector.

5.1.1 A Genetic Algorithm for the Fixed-Charge Transportation Problem

Several GAs have been proposed for tackling the FCTP. We present a fairly recent implementation.

This GA explores the set of basic solutions of the standard formulation of the transportation problem obtained by opening all arcs of the bipartite graph, as in Sects. 3.2.2 and 4.1.1. However, the representation of these solutions is quite different from what we have seen in these subsections. The GA uses a so-called *priority-based encoding* of solutions. In this scheme, each solution (i.e., individual or chromosome) is represented using a vector of length $(m + n)$, where m is the number of sources and n is the number of of destinations. Each chromosome corresponds to a permutation of the integers between 1 and $(m + n)$. The first m genes (entries) of the chromosome are associated with sources and the n last ones with destinations. The value of a specific gene indicates the priority given to a source or a destination when decoding the chromosome, with $(m + n)$ being the highest priority and 1 the lowest. The decoding of a chromosome is performed by considering the residual supply (resp. demand) in source (resp. destination) nodes and entries in the cost matrix for the problem, in a process reminescent of the greedy procedure of Sect. 3.1. The main difference, however, is that arcs on which flows will be added are selected on the basis of the priority-based representation. Considering the properties of the transportation problem, the decoding of a chromosome always result in a feasible solution whose fitness can be evaluated easily.

Because of its specific nature, the priority-based encoding requires the use of specialized crossover and mutation operators. Two new crossover operators are proposed: the *order of priority exchange* crossover (OPEX) and the *priority exchange* crossover (PEX). Both of these crossovers return two priority-based encoded offsprings from two parents in the same encoding. A specialized mutation operator, based on the OPEX crossover applied to two segments of a chromosome, is also used. The method uses a mixed selection strategy for population management. Computational experiments showed that the proposed GA would systematically outperform GAs with a spanning-tree based representation of chromosomes. Furthermore, the method can easily be adapted to tackle quadratic flow transportation costs.

5.1.2 A Genetic Algorithm for the Multicommodity Capacitated Fixed-Charge Network Design Problem

Let us now describe a fairly straighforward application of GA mechanisms to solve the MCFND. The search space for this implementation is the space of feasible binary design vectors. The fitness of any $\bar{\mathbf{y}}$ vector is given by the sum of the fixed costs of the arcs open in $\bar{\mathbf{y}}$ and the optimal value of the MCMNF problem associated to $\bar{\mathbf{y}}$ and defined by (4.15)–(4.18). The best individuals are thus those with a low fitness.

The method proceeds on a generation by generation basis. In each generation, a number of pairs of parents are selected for reproduction on the basis of their rank (w.r.t. to fitness) within the population, in accordance with roulette-wheel selection schemes. This selection scheme avoids a premature convergence of the population because of the dominance of *super-individuals* who would be repeatedly selected.

Two crossover operators are examined. Both of them yield a pair of offsprings from each pair of parents. The first one is the *uniform crossover*, which is based on the application of a randomly generated binary uniform mask m of the same length as the chromosomes. Each entry in m takes the value 0 or 1 with probability 0.5. When $m(i) = 1$, the i-th gene of the first offspring O_1 is copied from the i-th gene of the first parent P_1 and the i-th gene of the second offspring O_2 is copied from the i-th gene of the second parent P_2. When $m(i) = 0$, the roles of P_1 and P_2 are reversed.

A second crossover, called the *frequency crossover*, attempts to incorporate some of the knowledge gathered from the characteristics of good solutions to the problem at hand. Let ϕ_i be the frequency of apparition of arc i in good solutions. A randomly generated binary mask m in which the probability that $m(i) = 1$ is equal to ϕ_i is defined. The frequency crossover then works as follows: when $m(i) = 1$ and the i-th gene of the first parent P_1 is also equal to 1, then the i-th gene of the first offspring O_1 is set equal to 1 and the i-th gene of the second offspring O_2 is copied from the i-th gene of the second parent P_2; otherwise (i.e., when $m(i) = 0$ or when the i-th gene of the first parent P_1 is equal to 0), then the i-th gene of the first offspring O_1 is copied from the i-th gene of the second parent P_2 and the i-th gene of the second offspring O_2 is copied from the i-th gene of the first parent P_1, as in the uniform crossover when $m(i) = 0$. Feasible offsprings are added to the population. When the population reaches a given threshold, a number of least fit individuals are eliminated according to a $(\lambda + \mu)$ population management strategy.

The GA may be stopped after a set number of generations, of new offsprings, or when the population diversity becomes too low. This GA is also part of a hybrid parallel search method, which will be described in Sect. 7, and it proved quite successful in this context.

5.2 Path Relinking

Path Relinking was specifically developed to intensify and diversify the exploration of promising portions of the search space of combinatorial optimization problems. As such, it is not meant to be used as a stand-alone method, but rather to complement other solution procedures, which are often neighborhood-based metaheuristics.

The basic idea is to first construct, using some other method, an initial *reference set* of solutions (this is the initial population for this method). This reference set should contain, as much as possible, excellent solutions (they are called *elite* solutions), but one should also strive to maintain some level of diversity of solutions in the reference set. The main step of the method consists of choosing in the reference set two solutions, an *initial solution* and a *guiding solution*, and then following a path in the search space from the initial solution towards the guiding one, by gradually introducing in the initial solution features of the guiding solution. The rationale behind this procedure is that paths linking good solutions have an excellent chance of containing even better solutions. When improving solutions are found on a path, they can be added to the reference set. The exploration of the path between the initial and the guiding solutions can be stopped before reaching the guiding solution. At that time, new initial and guiding solutions are selected and the process repeated. Path Relinking can stop after a set CPU time or a given number of selections of initial and guiding solutions.

5.2.1 Path Relinking for the Multicommodity Capacitated Fixed-Charge Network Design Problem

Path Relinking was applied to the MCFND problem in the context of an extensive computational study. The basic metaheuristic applied was the Tabu Search that uses cycle-based neighborhoods. Different strategies were considered for collecting elite solutions and creating the reference set:

- (S1): Best solutions
- (S2): Best local minima found during the search
- (S3): Local minima that improve those already in the reference set
- (S4): Solutions far from those previously chosen and better than the worst one
- (S5): Best solutions, then extend with solutions far from those already chosen
- (S6): Best solutions, then extend with solutions close to those already chosen

Several criteria were also considered for the selection of the initial and the guiding solutions:

- (C1): Best and worst
- (C2): Best and the second best
- (C3): Best and the solution with the maximum Hamming distance from best
- (C4): Randomly
- (C5): The most distant solutions
- (C6): Worst and best

Trajectories between initial and guiding solutions are explored using a variant of the cycle-based neighborhood in which the flows of a single commodity are modified each time. This procedure allows to progressively introduce in the current solution arcs that are present in the guiding solution, but not not in the current one, and to remove arcs that do not belong to the guiding solution. The best solution obtained during a relinking process is added to the reference set, if it improves the best overall solution.

Computational experiments on both the **C** and the **R** instances of the Gendron-Crainic benchmark highlighted a number of important conclusions. First, it is not necessary to use a large reference set; in the experiments, using a reference set of size 6 yielded the better performance. Second, diversity and long relinking paths provide the most effective exploration and the best solutions in the end; hence, the combination of strategy (S3) and criterion (C5) led to the best results overall. Third, when it is properly used and calibrated, Path Relinking is an effective approach for tackling multicommodity capacitated fixed-charge network design problem.

5.3 Scatter Search

Scatter Search is another technique that has been proposed to combine good known solutions for a problem in the hope of finding even better ones. It shares many concepts with Path Relinking, such as the use of a reference set. The basic mechanism is to perform a weighted linear combination of two or more vectors that represent solutions extracted from the reference set. As such, it is therefore naturally suited to tackle continuous optimization problems. It can, however, be applied to problems with integer decision variables, such as network design problems.

5.3.1 Scatter Search for the Multicommodity Capacitated Fixed-Charge Network Design Problem

The search space for the application of Scatter Search to the MCFND encompasses the binary design variables. Since we are dealing with integer variables, the basic Scatter Search mechanism is adapted by resorting to sophisticated rounding procedures to fix some of the binary variables and letting others be free.

At the start of the algorithm, the cycle-based Tabu Search is applied to create an initial population of solutions, from which the reference set is initialized with with improving local optima, i.e., local optima better than solutions already in the set.

At each iteration, a *candidate set* CS of L solutions is created by including the best solution in the reference set, the solution that is the farthest away from it in Hamming distance, and a number of randomly selected elements of the reference set. The binary vectors for these L solutions are then combined using weights $\omega_l, \forall l \in CS$, to compute a *desirability factor* m_{ij} for each arc $(i, j) \in \mathscr{A}$. The desirability factors are then compared for each arc to two thresholds t_c and t_o, with $0 < t_c < t_o < 1$:

- When $0 \leq m_{ij} < t_c$, we close arc (i, j) in the new solution;
- When $t_o < m_{ij} \leq 1$, we open arc (i, j) in the new solution;
- When $t_c \leq m_{ij} \leq t_o$, we leave arc (i, j) undecided in the new solution.

Four different weighting schemes were tested for combining vectors of the candidate set. These are:

- Voting (V): $\omega_l = 1, \forall l \in CS$;
- Cost (C): $\omega_l = 1/$(cost difference between solution l and the best solution);

- Distance (H): $\omega_l = 1/(\text{Hamming distance between sol. } l \text{ and the best solution})$;
- Frequency (F): $\omega_l =$ frequency of arc (i, j) in the best solutions.

The incomplete solution obtained from the combination procedure is then further processed. First, a special minimum cost multicommodity network flow problem, similar to (4.15)–(4.18), is solved using CPLEX. In this MCMNF problem, arcs that have been closed by the rounding scheme have their capacity set to 0, while the variable cost of undecided arcs is set to $f_{ij}/u_{ij} + c_{ij}$. A feasible solution for the MCFND is extracted from the MCMNF optimal solution by opening all arcs with flow; this solution is then improved by applying the cycle-based Tabu Search.

Preliminary testing on a small set of instances permitted to identify the best values for the thresholds: 0.4 for t_c and 0.6 for t_o. Extensive computational experiments on the **C** instances of the Gendron-Crainic benchmark tested different values for the size L of the candidate set and the size of the reference set, as well as the various weighting schemes. These experiments led to the following conclusions. First, there should be at least 20 elements in the reference set. Second, results with only two elements in the candidate set are clearly inferior; results with $L = 3, 4$, or 5 are globally comparable; in practice, it seems that $L = 3$ or 4 is the best choice. Third, while all combinations of L and weighting schemes can produce on some instance results that are better than those of the Path Relinking implementation of Sect. 5.2.1, on average, none of the combinations does as well. Fourth, the Frequency weighting scheme is clearly inferior to the other three. Overall, the observed performance seems to indicate that the full potential of Scatter Search has not been exploited in the proposed method.

5.3.2 An Improved Scatter Search-Evolutionary Algorithm for the Multicommodity Capacitated Fixed-Charge Network Design Problem

We now consider a more involved method involving Scatter Search to tackle MCFND problems. This method can be decomposed into three major phases: an Initialization phase, a Scatter Search phase, and an Education phase based on Iterated Local Search (ILS). While the Initialization phase is executed only once at the beginning of the algorithm to produce an initial population of solutions stored in reference set R, the two other phases are performed iteratively to improve solutions in set R. The termination criterion of the algorithm is the elapsed CPU time.

The Initialization phase seeks to produce a population of solutions that are of good quality, but also diverse. The solutions considered at this step include both the binary design variables and the flow variables. These initial solutions are constructed by gradually constructing shortest paths for all commodities between their origin and their destination; the sequence in which commodities are routed is randomized and single or multiple paths may be constructed for each commodity. λ solutions are created, but only μ with $\mu < \lambda$ are retained. In fact, the first μ solutions created are first assigned to R and each one of the remaining $(\lambda - \mu)$ is considered for replacing the worst solution in R, if it cost is lower than that the cost of the best solution in R or if it dominates some solution in R both in terms of solution quality and solution diversity (measured in terms of Hamming distance to the best solution).

It should be noted that the search that takes place in the following phases is performed in the space of the binary design variables. The values of the flow variables for a given design are obtained by solving the associated MCMNF problem, as in several of the other solution methods that we have seen for the MCFND.

In the Scatter Search phase, solution recombination takes place. In each iteration, a total of 2μ offsprings are created, each one being determined by combining features from a *candidate set* CS made up of κ solutions. The selection procedure for choosing the solutions that make up the candidate set is probabilistic, but it does not involve the cost of solutions directly. Instead, as the method proceeds, information on the performance of solutions with respect to the Education phase is recorded, for each, in a measure which is called its *solvency ratio* (SR). For any given solution s, SR is the ratio of the number of times that an offspring of solution s has been included in R (after education) to the number of times that solution s has been selected to produce an offspring. It is thus a measure of the effectiveness of solution s to produce useful offsprings for future generations. The recombination process itself is based, for each arc (i, j), on a weighted sum of the design variables of the solutions of CS for this arc. The weight given to each depends upon its objective value and a correction factor to lessen the importance of solutions that have appeared often in CS. This computation allows to determine a *preferred status* of open or closed for each arc in the offspring. The corresponding binary vector is then a tentative design solution, for which one can solve the associated MCMNF problem with a linear programming solver. If this problem is feasible, the offspring can proceed; if it is infeasible, the design must be repaired using a process similar to the one performed in the Initialization phase. At the end of the Scatter Search phase, the best μ offsprings, in terms of solution cost, are retained to proceed to the next phase.

In the Education phase, the μ best offsprings coming from the Scatter Search phase go through a process aimed at improving their quality. Each offspring is educated individually by going through an ILS procedure. This ILS procedure has two components: a neighborhood search that can be seen as an extension of the cycle-based Tabu Search of Sect. 4.1.2, and a perturbation strategy, called *Ejection Cycles*, which partially modifies the current solution using information stored in a long-term memory of the search.

This method was applied to the **C** instances of the Gendron-Crainic benchmark. The results obtained show that this method is very competitive with the best existing ones (including the methods described in the next section) and could identify new best solutions for some instances.

6 Matheuristics

Matheuristics is a broad term that covers all solution approaches that combine exact solution procedures with metaheuristic methods. As such, there is no set recipe for deriving matheuristics; each matheuristic is a unique method and should be described as such for the time being. Perhaps, in the future, someone will propose a taxonomy of matheuristics and provide a more insightful description of these methods. In this chapter, we present four interesting matheuristics that have been proposed for tackling network design problems.

6.1 A Local Branching Matheuristic for the Multicommodity Capacitated Fixed-Charge Network Design Problem

Local Branching (LB) is a matheuristic that was developed in the early 2000's to leverage the power of mixed integer programming (MIP) solvers. It is particularly aimed at combinatorial optimization problems that can be modeled with binary decision variables. The central idea of LB is to explore partially the space of feasible binary vectors using depth-first tree search. At each iteration, the search is limited to considering a fairly small neighborhood of a *reference solution* by the addition of so-called *local branching constraints*, which require solutions to differ from the reference solution by no more than k values. This defines what is called a *k-Opt* neighborhood. The reference solution is updated whenever an improving incumbent is found. In theory, if the underlying tree search was performed completely, LB would be an exact method, but the addition of CPU time limits and bounds on the depth of some branches of the search tree turns it into an approximate procedure.

Network design problems with their binary design decision variables are natural candidates for the application of LB, which was thus applied to the MCFND. The MIP formulation used in the application is the arc-based formulation of Chap. 2 to which are added the *strong linking constraints* described in this chapter, as well as the local branching constraints. An important feature of the method as implemented is that the MIP considered at each node of the search tree is not solved to optimality; as soon as a feasible solution that improves upon the value of the reference solution is found, the values of the design variables are fixed and optimal values for the flow variables are derived from the resulting MCMNF. This procedure guarantees that the proper objective function value is determined for the corresponding design.

Computational experiments on the **C** instances of the Gendron-Crainic benchmark showed that an LB-based matheuristic could lead to improved solutions for several instances in reasonable CPU times.

6.2 A Matheuristic Combining Exact and Heuristic Approaches for the Multicommodity Capacitated Fixed-Charge Network Design Problem

We now describe a method that was developed with the objective of producing provably high-quality solutions quickly for the MCFND. The method relies on a neighborhood search procedure that explores the space of binary design variables using a MIP solver, in a fashion reminescent of the method presented in the previous subsection. In each iteration, a subset of design variables of the arc-based formulation of the MCFND is thus chosen and "freed", while the other binary variables are held fixed, and the resulting MIP is solved. Several strategies for choosing which arcs should be freed are used; these involve identifying arcs that are used in some commodity paths of the current solution, in paths derived from solving some linear relaxations, in paths that were observed in improving solutions, and so on. Proper choice of the restricted IP and proper seeding of the initial solution used to solve it should result in most cases in improved solutions and thus better upper bounds.

A lower bound on the optimal value of the problem is also computed at each iteration. It is provided by the value of the LP relaxation of the path-based formulation. However, since this bound is known to be weak, valid inequalities are added to this formulation. These include strong linking inequalities between the design and the flow variables, as well as lifted cover inequalities found while solving the small MIP at each iteration.

The proposed approach was tested on a comprehensive set of instances. In particular, it was used to tackle the 37 most difficult **C** instances of the Gendron-Crainic benchmark. On these instances, it clearly outperformed the methods presented in Sects. 4.1.2 and 5.2, both in terms of solution quality and computing times. It was also able to find in a few minutes solutions that were on average only 2.37% more expensive than to those obtained by CPLEX in several hours of computation. When tested on larger instances with 500 nodes, 2000–3000 arcs, and 50–200 commodities, it produced in 15 min of CPU time solutions that were on average more than 20% better than those obtained by CPLEX after 12 h of computation.

It is important to note that this solution approach can tackle not only the MCFND basic formulation presented in Chap. 2, but also its variant in which commodities must be routed on a single path between their origin and their destination. This is partly due to the fact that the mathematical formulations used are not exactly the ones presented in Chap. 2: instead of using variables x_{ij}^k to represent the flow of commodity k on arc (i, j), these variables denote the *fraction of the demand* of commodity k that uses arc (i, j). Single-path constraints for commodities are easily enforced by requiring these x_{ij}^k variables to be binary.

6.3 A Hybrid Simulated Annealing-Column Generation Matheuristic for the Multicommodity Capacitated Fixed-Charge Network Design Problem

This method is a fairly straightforward hybridization of Simulated Annealing to determine the values of the binary design variable and Column Generation to solve the path-based formulation of the MCMNF problem for a given vector of binary variables $\bar{\mathbf{y}}$. Thus, the SA heuristic explores the search space of binary design vectors using moves in the add-drop neighborhood. Several strategies are considered for choosing the arcs to be closed, sometimes complemented by the opening of arcs along some paths when the current solution becomes infeasible. The objective function associated with any solution is the sum of the fixed costs of open design arcs plus the objective value of the corresponding MCMNF problem. Solutions are accepted (and thus moves performed) according to standard SA acceptance rules. The temperature is adjusted after a fixed number of add/drop cycles; it is lowered according to a linear function. In this method, the methods stops after a examining pre-determined number of temperature settings without improvement.

Extensive analysis was performed to tune the various parameters of the method. With the best parameter values, the method was shown to perform quite well. On the 43 **C** instances of the Gendron-Crainic benchmark, it produced better results than the Local Branching algorithm of Sect. 6.1 and CPLEX with a time limit of 600 s. These conclusions were shown to be statistically significant.

6.4 A Cutting-Plane Based Matheuristic for the Multicommodity Capacitated Fixed-Charge Network Design Problem

This matheuristic is based on a new neighborhood structure which relies on cutting planes for its definition. The fashion in which the neighborhood is defined is procedural. Given an incumbent (current) solution, an arc open in this solution is selected to be closed. Several criteria were considered for selecting that arc, but in the end two were retained: (1) the arc with the maximum combined unit cost (including fixed and variable costs), given its current flow; (2) the arc with the highest fixed cost for residual capacity (i.e., the product of the arc fixed cost by the ratio of unused to total capacity). These two criteria point out to open arcs whose usage does not seem to be very efficient. Then, the continuous (LP) relaxation of the MCFND is solved with two additional constraints: (1) a constraint closing the arc selected in the previous step, and (2) a constraint requiring the number of open arcs (other than the one selected to be closed) to remain the same as in the incumbent. If this LP is infeasible, a new open arc is selected to be closed and the procedure is repeated. Once a feasible solution to the LP is found, families of valid inequalities are added to this LP. Three families are considered: (1) strong linking constraints (presented in Chap. 2) between individual commodity flows and design variables; (2) flow cover inequalities, and (3) flow pack inequalities. It should be noted that strong linking constraints are added to the LP as long as they increase its objective value. As for the flow cover and flow packing inequalities, they are identified by applying a separation algorithm presented in Chouman et al. (2009). These valid inequalities and the constraint on the number of open arcs define a so-called *strong LP*. A new MIP sub-model is then defined from the strong LP by adding to it three constraints: (1) one that forces to open arcs that carry flow in both the strong LP solution and the incumbent; (2) a constraint that closes arcs that do not carry flow in both the strong LP solution and the incumbent; and (3) the constraint that forces to close the arc selected at the beginning of the procedure. The resulting MIP is solved by Local Branching, as in the method described in Sect. 6.1. The solution obtained is the desired neighbor to the current solution.

The proposed neighborhood structure was used within a Tabu Search heuristic. The initial solution for this Tabu Search is a feasible one, which is obtained by solving the LP relaxation of the problem, adding valid inequalities as in the procedure that defines the neighborhood structure, and rounding up the value of binary design variables in this strong LP. Two tabu lists are used. The first one records the list of recently closed arcs; these arcs are forced to remain closed for a number of iterations to prevent cycling. The second tabu list is used whenever the new solution produced by the neighborhood structure is infeasible or worse than the incumbent solution. In that case, several moves are attempted from the incumbent until an improving one is found or a pre-defined number of non-improving neighbors is reached; the tabu list is used to record the arcs closed in the unsuccessful attempts. The procedure stops after a pre-defined CPU time or a set number of iterations without improvement of the incumbent.

The values of the parameters of the proposed tabu search heuristic were carefully tuned using a statistical design of the experiments. Computational results on 37 of the 43 **C** instances of the Gendron-Crainic benchmark (the six easiest problems were left out) showed the effectiveness of the proposed methods: it clearly outperformed the oldest methods and did as well or better than the most up-to-date.

7 Parallel Metaheuristics

Parallel/distributed/concurrent computing means that several processes work simultaneously on several processors addressing a given problem instance and aiming to identify the best or a good feasible solution for that instance. Parallel metaheuristics and matheuristics aim for two major goals. The first, common to all parallel-computing developments, is to solve larger problem instances, faster. The second is proper to heuristics and it concerns the robustness of the search, i.e., its capability to offer a consistently high level of performance over a wide variety of problem settings and instance characteristics. Parallel metaheuristics built according to cooperative multi-search strategies have thus proved to be much more robust than sequential versions, offering higher quality solutions, and requiring less extensive, and expensive, parameter-calibration efforts. It is worth noticing that multi-search methods, particularly when based on cooperation, generally display a behavior different from those of the sequential methods involved, offering enhanced performance compared to sequential methods and other parallelization strategies. They are thus acknowledged as proper metaheuristics.

The objective of this section is to present a brief unified overview of the main parallel metaheuristic concepts and strategies. Note that these are of a general nature with respect to the metaheuristic type and combinatorial optimization problems. We will identify network design applications, of course, and will signal relevant particularities for neighborhood- and population-based meta- and matheuristics.

Parallelism follows from a decomposition of the algorithmic work required to address the problem, and the distribution of the resulting tasks to available processors. The decomposition may concern the algorithm, the search space, or the problem structure. Parallel strategies may then be classified according to, on the one hand, what is decomposed and how it is done, and, on the other hand, how the global problem-solving search is controlled, how information is exchanged among tasks and how, eventually, new information is created (the diversity of the searches involved may also be used, but we skip it in the following for chapter-length reasons).

Functional parallelism and *search-space separation* make up the first dimension. Within each decomposition strategy, strategies are then described in terms of *Search Control Cardinality*, specifying whether the global search is controlled by a single (*1-control*—1C) or several (*p-control*—pC*)* processes, which may collaborate or not, as well as of *Search Control and Communications*, addressing how information is exchanged and used to control or guide the search. *Synchronous* and *asynchronous* communications are found in parallel computing. All processes stop, at moments exogenously determined, in the former case, to engage in some form of communication and information exchange. In the latter case, each process is in charge of its own search and of establishing communications and exchanges with other processes. Four categories are defined to reflect the quantity and quality of the information exchanged and shared, as well as the additional knowledge derived from these exchanges (if any); *Rigid* and *Knowledge* synchronization, and *Collegial (C)* and *Knowledge Collegial (KC)* asynchronous strategies. These concepts and their applications to network design is further described in the next subsections.

7.1 Functional Parallel Strategies

Functional or *low-level* parallelism decomposes computing-intensive parts of the algorithm into a number of tasks, which work on the same data or on a dedicated part, and run in parallel. Such strategies thus aim to accelerate the search, without modifying the algorithmic logic, the search space, or the behavior of the sequential metaheuristic. Most low-level parallel strategies belong to the 1C/RS class and are usually implemented according to the classical *master-slave* parallel programming model. A "master" program initiates the exploration from a single solution or population. It executes the sequential metaheuristic (1-control), separating and dispatching computation-intensive tasks, which often corresponds for metaheuristics to the execution of the innermost loop iterations, e.g., evaluating neighbors or individuals, or having ants forage concurrently. Slaves perform the tasks in parallel, and return the results to the master which, once all the results are in, resumes the normal logic of the sequential metaheuristic. The master has complete control on the algorithm execution; it decides the work allocation for all other processors and initiates communications. No communications take place among slave programs.

Functional parallelism is often a low-level component of hierarchical parallelization strategies and may be extremely interesting when the problem requires a significant part of the computing effort to be spent in inner-loop algorithmic components. This is often the case for network design, in particular when search spaces are based on the design decision variables. The evaluation of neighbors and individuals often requires in this case to find the optimal, or at least a very good solution to the multicommodity capacitated network flow subproblem, which may be quite computationally intensive. The parallel evaluation of several neighbors, of the fitness and diversity measures of individuals, or of scatter-search operators on several combinations of solutions in the reference set could be of significant help in those circumstances. Note that similar observations may be made for pivot-based moves. Note also that one could use the *graphical processing units* (*GPU*), ubiquitous within most computers, to further parallelize pivot operations, either as moves, or as part of customized linear-programming solvers. This is an interesting research area.

7.2 Search-Space Separation: Domain-Decomposition Strategies

The general idea of the second major class of parallel strategies is to decompose the search space and to address the problem on each resulting component using a particular solution method. Two main classes of such strategies may be defined: *domain decomposition* and *multi search*. The former explicitly separates the space yielding a number of subproblems to be addressed simultaneously, their solutions being then combined into a complete solution to the original problem, while the latter performs the separation implicitly, through the concurrent explorations of the complete space by several methods, named *solvers* in the following, which may exchange information or not. Multi-search strategies, particularly those based on cooperation principles, are at the core of most successful developments in parallel metaheuristics, particularly for complex, multi-attribute combinatorial problem, as those found in network design and applications. Note that search-space decomposition methods induce different search behaviors and yield different solutions when compared to the corresponding sequential metaheuristics involved.

The basic idea of *domain decomposition* is intuitively simple and appealing: separate the search space into smaller subspaces, address the resulting subproblems by applying the sequential metaheuristic on each subspace, collect the respective partial solutions, and reconstruct an entire solution out of the partial ones. The subspaces may be disjoint, their union yielding the full space, or a certain amount of subspace overlap may be allowed, either explicitly, or implicitly by allowing the search within a given subspace to reach out to some part of one or several other subspaces through, e.g., neighborhood moves or individual crossovers. The separation may be obtained by identifying a subset of variables (and corresponding constraints, eventually) and discarding or fixing the other variables and constraints, the goal being to obtain smaller, easier to address subproblems. For network design, separation could thus be defined along groups of arcs, nodes (commodity origins or destinations), commodities, paths of potential design arcs, etc., based on attributes of the corresponding elements (e.g., proximity of nodes or arcs fixed cost over capacity ratio) or solutions at previous iterations. Discarding does not appear appropriate when considering the commodities and arcs of multicommodity capacitated network design problems, however. Separation by variable fixing (and projection of the corresponding constraints) appears more flexible as one still works on smaller subproblems, but considering the complete vector of decision variables, some of which are fixed. This is also a more general approach and it is part of advanced cooperative search methods described later.

Notice that strict partitioning restricts the solvers to their subspaces, resulting in part of the search space being unreachable and the loss of exploration quality. Covers partially address this issue. Yet, to guarantee that all potential solutions are reachable, one must make overlapping cover the entire search space, which negates the benefits of decomposition. The "change the separation and start again" idea is at the core of the 1C/RS and pC/KS strategies proposed to avoid these drawbacks, where the separation is modified periodically, and the search is restarted using the new decomposition. The master (in the former case, or the collaborating processes, in the latter) applies the decomposition, reconstructs complete solutions, modifies the separation, and determines when stopping conditions are met.

An application of this strategy to the MCFND may partition the search space of full feasible solutions based on commodities and variable fixing. Starting from a given (initial) solution, one forms partitions of commodities sharing at least one design arc (a graph-partitioning method minimizing the number of shared arcs could be used). The design and flow decisions corresponding to commodities not belonging to a given partition are fixed. Each partition corresponds to a smaller, and hopefully easier to address, problem and is explored by a Local Search heuristic or a more complex metaheuristic, as presented in the previous sections of the chapter. The best solutions obtained within the partitions are collected once the explorations of all partitions are completed, a complete solution is built, a new partition is determined, and the search is restarted. Several approaches may be used to combine the partial solutions into a complete one, e.g., fix to 0 all design arcs not used in any partition and solve exactly the reduced problem instance (assuming the size has been sufficiently reduced), or fix to 0 as previously and fix to 1 the design arcs with significant flow with respect to capacity (the fixed cost to flow ratio can also be used) and solve the reduced problem. Notice that, similar to all other sequential and parallel metaheuristics, when variable fixing yields a sufficiently small problem instance, an exact MIP solution method may be used to a explore the large neighborhood consisting of all solutions which may be obtained taking the current one as initial solution. We thus define a matheuristic solution method.

Network design requires heavy computation work at each iteration in most cases of interest. Hence, performing many iterations on several space separations may not always appear appropriate and particular care must be taken of the efficiency of the partition exploration. Yet, as the complexity (e.g., time representations and several interrelated combinatorial decision layers, etc.) and dimensions (number of commodities, arcs, time periods, etc.) of the contemplated network design problems continue to grow, decomposition methods appear increasingly needed, in particular combined with other parallelization strategies, cooperation in particular. This defines a rich and challenging research area.

7.3 Search-Space Separation: Multi-Search Strategies

Independent multi-search was among the first parallelization strategies proposed. It is also the most simple and straightforward pC parallelization strategy, offering a simple tool for looking for a "good" solution without investment in methodological development or coding. It seeks to accelerate the exploration of the search space by initiating simultaneous solvers from different initial points (with or without different search strategies). No attempt is made to take advantage of the multiple solvers running in parallel other than to identify the best overall solution as a final synchronization step. One therefore needs quite a relatively high number of solvers running in parallel to achieve interesting results.

Cooperative multi-search has emerged as one of the most successful meta-heuristic methodologies to address hard optimization problems. Cooperative-search strategies go beyond simultaneous runs of multiple independent solvers, and integrate *cooperation mechanisms* to share, *while the search is in progress*, the information obtained from this diversified exploration of the same problem instance. This sharing, and the eventual creation of new information out of the shared one, yields in most cases a collective output of superior quality compared to independent and sequential search, and makes cooperative multi-search a "new" metaheuristic class.

Cooperative-search strategies are defined by the solvers engaged in cooperation and their interaction mechanism, including the nature of the information shared. The metaheuristic or exact solvers do not need to belong to the same solution-method class and may address either the complete problem at hand, or explore partial problems defined by decomposing the initial problem through mathematical programming or attribute-based heuristic approaches. The decomposition method implicitly defines how a complete solution is built out of partial ones, in the former case. In the latter case, some solvers work on partial problems defined by the particular sets of attributes selected in the decomposition, while others combine the resulting partial solutions into complete solutions to the original problem.

The information-sharing cooperation mechanism specifies how the solvers interact, that is, what information is exchanged and when, how the exchanged information is used globally (if at all), and how each solver acts on the received information, using it within its own search and, thus, transforming it before passing it to other solvers. The goals are to improve the performance of the solvers, and to create a global image, even if not complete and imprecise, of the status of the cooperative search. The status information may then be used to guide solvers, and thus the global search, toward a better performance in terms of solution quality and computational efficiency than the simple concatenation of results obtained by non-cooperating solvers. Exchanged information must be *meaningful* and exchanges must be *timely*.

When and how information is shared specifies the frequency of cooperation activities, who initiates them and when, and whether the concerned solvers must synchronize or not. We do not elaborate further on synchronous communications, partially because of space restrictions, but mostly because synchronization makes parallelism more rigid, and exchanges less timely with respect to the value of new information for solvers. (Notice that most of the discussion on cooperative design applies to synchronous methods as well.) Strategies based on *asynchronous* exchanges thus proved superior to synchronous ones, and are considered as defining the "state-of-the-art" in parallel multi-search metaheuristics. Asynchronous cooperative strategies follow pC/C or pC/KC collegial principles, the main difference being that, "new" knowledge (solutions and more as detailed next) is inferred in the latter case, starting from the information exchanged between solvers.

Asynchronous communications provide the means to build cooperation and information sharing among solvers without incurring synchronization overheads. They also bring adaptability to cooperation strategies, to the extend that the parallel metaheuristic may react more easily and dynamically adapt to the information brought by the other solvers and exploration of the search space, than independent or synchronous search can. These benefits come with potential issues one must care for. For example, the available global-search information available to a given solver may be less "complete" than in a synchronous environment. On the other hand, too frequent data exchanges, combined with simple acceptance rules for incoming information, may induce either a premature solver "convergence" to local optima or an erratic search trajectory similar to a random walk. Hence the interest for applying information-sharing based on quality, meaningfulness, and parsimony principles.

These principles translate into good and diverse solutions being the type of information most often shared, either a single one each time or as a small set. "Good" generally means a local optimum at the end of a search phase, a solution out of an elite set built during the search, or the last solution of an improving sequence of moves. The last case illustrates the parsimony and meaningfulness ideas. One could, clearly, share each improving solution the method identifies. As such solutions are usually found within a sequence of improving moves, most of them are very similar and, thus, they offer little new information to the other solvers, while significantly disturbing their searches and increasing the communication overload of the parallel metaheuristic. On the other hand, the last improving solution of the sequence brings meaningful information about the status of the search of the emitting solver; it also diversifies the shared information with respect to previous communications. Numerous experiments emphasized the importance of diversification in the shared information. Thus, always selecting and sharing the best available solution out of an elite set proved to rapidly decrease the breath of the search, increase the amount of worthless computational work (many solvers search in the same region), and bring the search to be confined within an often-visited region of the search space, or to an early "convergence" to a not-so-good solution. Strategies that select randomly among an elite set, but bias the choice toward good-and-different solutions, proved more efficient and yielded higher-quality solutions.

Context information may also be shared profitably, particularly when embedded in mechanisms used to generate new knowledge or to guide the search. Context refers to data collected by a solver during its exploration, such as the statistical information relative to the presence of particular solution elements in improving solutions (e.g., the medium and long-term memories on selected design arcs or cycles of design arcs built by tabu search), the impact of particular moves on the search trajectory (e.g., the scores of closing/opening design paths within a large adaptive neighborhood search), population diversity measures and individual resilience across generations, etc.

Cooperating solvers may exchange information directly or indirectly. *Direct* exchanges of information, often used in the genetic-based evolutionary literature, occur either when a number of concerned solvers agree on a meeting point in time to share information, or when a solver broadcasts its information to one or several other solvers without prior mutual agreement. The latter case is to be avoided as it requires solvers to include capabilities to store received information without disturbing their own search trajectories until they are ready to consider it. Failure to implement such mechanisms generally results in bad performance, as observed for strategies combining uncontrolled broadcasting of information and immediate acceptance of received data.

Indirect exchanges of information are performed through an independent, "centralized", device (implemented on a single processor or on a layered array of processors) called *memory* in this chapter. The memory serves as shared resource of data for cooperating solvers, which access it according to their own internal logic to post and retrieve information. Classical retrieval mechanisms are based on random selection, which may be uniform or, similar to the case of posted information, biased to favor solutions with high rankings based on solution value and diversity. The memory accepts incoming solutions for as long as it is not full. Acceptance becomes conditional to the relative interest of the incoming solution in terms of value, compared to the "worst" solution in the memory, and diversity, a slightly worse solution being preferred if it increases the diversity of solutions in the memory.

To illustrate, consider three pC/C metaheuristics for the MCFND. The first is based on the principles of repetitive applications of Local Search and perturbation of the resulting local optima (e.g., GRASP and Iterated Local Search). Each solver runs a Local Search for the MCFND on the full problem instance for a given initial solution. Solvers deposit their final local optima into the memory, if improved during the current run of the LS, and retrieve a new starting solution (different from the ones already explored) from the memory. The method may be enriched by having each solver run first a Local Search building an elite set of solutions, followed by a Path Relinking on that elite set. Solvers may then send a small group of solutions (e.g., the Local-Search local optima and the best and most diverse solutions yielded by the Path Relinking) to the memory. In a more advanced pC/KC version, the Path Relinking method works also on the solutions in the memory to construct the new starting solution when requested by a solver.

The second pC/C metaheuristic is a memory-based approach for asynchronous parallel Tabu Search, that may be generalized to most neighborhood-based metaheuristics. Each solver runs a TS for the MCFND (see Sect. 4.1.2) from the same or different initial solutions. Solvers could run the same metaheuristic (from different initial solutions, obviously), but it has been shown that running different metaheuristics (different possibly in the value of certain key parameters only) yields a much better overall search. Each solver sends to the memory its local optima, when improved, and imports a solution from the memory before engaging in a diversification phase. The imported solution is selected randomly, biased by rank (in terms of solution value) or by diversity when the selection is made within an elite set. More advanced pC/KC versions are discussed further in the section.

The third illustration is the *Multi-level Cooperative Search*, which proposes a different pC/C asynchronous cooperative strategy based on the controlled diffusion of information. Solvers are arrayed in a linear, conceptually vertical, communication graph and a local memory is associated to each. Each solver works on the original problem but at a different level of aggregation or "coarsening", the first-level solver addressing the complete original problem. It runs a sequential metaheuristic for the MCFND, and communicates exclusively with the two solvers directly above and below, that is, at a higher and a lower aggregation level, respectively. Each solver shares improved solutions, incoming solutions not being transmitted further until modified locally for a number of iterations to enforce the controlled diffusion of information. The local memory is used to receive the information coming from the immediate neighbors, the solver accessing it at moments dynamically determined according to its internal logic (e.g., before diversification). It is noteworthy that one can implement Multi-level Cooperative Search using a centralized memory by adequately defining the communication protocols. Although not yet fully defined and tested, this idea is interesting as it opens the possibility of richer exchange mechanisms combining controlled diffusion and general availability of global information.

Cooperative strategies including mechanisms to create new information and solutions based on the solutions exchanged belong to the p-control knowledge collegial (pC/KC) class. These strategies build on the flexibility of cooperation in terms of the different metaheuristics and exact methods that can be combined, and on the population of elite solutions being hosted in the centralized memory and continuously enhanced by the cooperating solvers. Mechanisms associated to the memory may thus, e.g., extract and update statistics on the presence of design arcs in the shared solutions (context information on similar statistics within particular solvers may be equally used, when available), possibly with an associated score relative to the solution containing them. Patterns of desirable or undesirable arcs may be thus constructed, and may be evolved to reflect the global status of the search (e.g., initial phase of broad exploration, middle of the solution path starting to reduce the scope of the search, or final phases intensifying in very promising regions before a final choice). Guidance may then be obtained by transmitting arc patterns to solvers, to guide their trajectories toward intensification or diversification phases (enforce the desirability or penalize the selection of certain arcs). Other learning-type mechanisms may be devised (e.g., one may focus on design or commodity paths or trees), this providing a rich field for research. New knowledge may also result from post-optimization applied to solutions in memory, and by generating new solutions out of the elite population in memory by heuristic, genetic, Scatter Search or Path Relinking methods. The new solutions may be returned, on request, to cooperating solvers. They also feed the learning mechanisms associated to the memory.

Historically, two main classes of pC/KC cooperative mechanisms are found in the literature, both based on the idea of exploiting a set of elite solutions, and their attributes, exchanged by cooperating solvers working on the complete problem, but differing in the information kept in memory, *adaptive-memory* and *central-memory*.

Adaptive-memory stores and scores partial elements of good solutions and combines them to create new complete solutions that are then improved by the cooperating solvers. The main idea, as initially proposed and applied, is to keep in memory the individual components (e.g., design arcs, or design or commodity paths or trees) making up the elite solutions found by the cooperating solvers, together with memories counting for each component its frequency of inclusion in the best solutions encountered so far, as well as its score, and rank among the population in memory, computed from the attribute values, in particular the objective value of its respective solutions. Solvers construct solutions out of probabilistically selected (biased by rank) solution components in memory, enhance it by Tabu Search or another metaheuristic and deposit their best solutions in the adaptive memory. The probabilistic selection yields, in almost all cases, a new solution made up of components from different elite solutions, thus inducing a diversification effect.

Central-memory methods generalize pC/C and adaptive-memory strategies. Complete elite solutions are kept in memory, as well as attributes and context information sent by the solvers involved in cooperation or generated in the central memory. Together, this data is used to create new solutions and knowledge to guide the cooperating solvers and the global search. Solvers may perform constructive, improving and post-optimization heuristics, neighborhood- and population-based metaheuristics and matheuristics, as well as exact solution methods on possibly restricted (through variable fixing) versions of the problem. Other than the information received from the cooperating solvers, the central memory keeps newly created information out of the shared data. Statistics-building, information-extraction and learning, and new solution-creation mechanisms provide this new "knowledge". Memories recording the performance of individual solutions, solution components, and solvers may be added to the central memory, and guidance mechanisms based on this knowledge may be gradually built.

We illustrate the basic canvas for central-memory strategies with an early pC/KS metaheuristic combining a genetic solver (Sect. 5.1.2) and several solvers executing the pC/C Tabu Search for the MCFND described above. The TS solvers aggressively explore the search space, building the elite solution set in the memory, while the GA contributes toward increasing the diversity, and hopefully the quality, of the solutions in the memory, which the cooperating TS solvers import. The GA launches once a certain number of elite solutions identified by the TS solvers are recorded in memory, which becomes the initial GA population. Once running, asynchronous migration transfers the best solution of the genetic pool to the memory, as well as solutions from memory toward the genetic population. This strategy was actually implemented and performed well, especially on larger instances. Moreover, it yielded the interesting observation that, while the best overall solution was never found by the GA solver, its inclusion in cooperation allowed the TS solvers to find better solutions, more diversity among solutions in memory translating into a more effective diversification of the global search.

We complete this section by addressing recent developments targeting large or multi-attribute problem settings. The general idea of the new generation of pC/KC meta-heuristics, called *Integrative Cooperative Search* (*ICS*), is to decompose the problem formulation along sets of decision variables to simpler but meaningful problem settings, in the sense that efficient solvers, can be "easily" obtained for the partial problems either by opportunistically using existing high-performing methods or by developing new ones. The decomposition approached evoked relative to domain-decomposition strategies may be used in the ICS context. More complex network structures, e.g., several design layers and time-space network representation, may be decomposed along the layer or time dimensions, respectively. The main components of ICS, to be instantiated for each application, are (1) the decomposition rule; (2) the *Partial Solver Groups* (*PSGs*) addressing the partial problems resulting from the decomposition; (3) the *Integrators* selecting partial solutions from PSGs, combining them, and sending the resulting complete solutions to the *Complete Solver Group* (*CSG*); and (4) the CSG, providing the central memory functionalities of ICS. Notice that, in order to facilitate the cooperation, a unique solution representation, obtained by fixing rather than eliminating variables when defining partial problems, is used throughout ICS. To illustrate consider the generalized network design problem where nodes and arcs are characterized by several attributes (various types, modes, and volumes of capacity, for example) governed by compatibility rules. The problem could then be decomposed along compatible combinations of attributes (a spatial decomposition could be also applied for large networks), Tabu Search solvers could be assigned to each combination (more than one solver could work on each partial problem according to a pC/C parallel strategy), while a population-based method (or group thereof), Genetic Algorithm, Scatter Search, or Path Relinking, could integrate the partial solutions generated by the TS solvers into complete solutions to the original problem. Monitoring, learning, and guidance activities complement the ICS method. Not much work has been reported in this area for network design, but ICS and related developments make up a promising but challenging research perspective.

8 Bibliographical Notes

The use of heuristics to tackle difficult combinatorial optimization problems goes back to the beginnings of operations research. It is therefore not surprising to find some simple heuristics proposed to tackle network design problems, with a clear emphasis on the fixed-charge transportation problem (FCTP). Several papers proposing Local Search methods for the FCTP were published in the late 1960s and the 1970s. Our discussion of Sect. 3.2.2 is based on the paper by Walker (1976), but several other authors proposed methods that exploited similar ideas (Cooper and Drebes 1967; Denzler 1969; Steinberg 1970; Cooper 1975).

From the mid-seventies to the following decade, one saw the rapid development of metaheuristics, which were applied to a large range of combinatorial problems and rapidly became the "go-to approaches" in many circles. While Genetic Algorithms (GAs) came first (Holland 1975), the combinatorial optimization community became keenly aware of the potential of methods that used completely "different" approaches and principles after of the publication of the paper that introduced Simulated Annealing (SA) to a wide audience (Kirkpatrick et al. 1983). This was followed closely by the seminal paper that coined the term *metaheuristics* and presented more formally Tabu Search (TS) (Glover 1986). Other fundamental references on this topic appeared in the following years and laid much of the groundwork for the application of TS (Glover 1989, 1990; Glover and Laguna 1993, 1997) and other metaheuristics, such as Greedy Randomized Adaptive Search Procedure (GRASP) (Feo and Resende 1989, 1995), Variable Neighborhood Search (VNS) (Mladenović and Hansen 1997; Hansen and Mladenović 1999), Path Relinking (PR) and Scatter Search (SS) (Glover 1997; Glover et al. 2000). The case of Iterated Local Search is somewhat stranger since, according to Lourenço et al. (2019), p. 130, "this simple idea has a long history" that goes back to Baxter (1981), but it was formalized systematically in later years. Metaheuristics came of age with the publication in 2003 of the first *Handbook of Metaheuristics* covering a wide range of methods and showing their common points and their successes in the solution of difficult problems (Glover and Kochenberger 2003). Metaheuristics have continued to evolve up to now. An up-to-date introduction to several of the methods covered in this chapter, as well as latest developments, can be found in the latest edition of the Handbook: Tabu Search (Gendreau and Potvin 2019), Simulated Annealing (Delahaye et al. 2019), Variable Neighborhood Search (Hansen et al. 2019), Iterated Local Search (Lourenço et al. 2019), Greedy Randomized Adaptive

Search Procedure (Resende and Ribeiro 2019), Genetic Algorithms (Whitley 2019), and parallel metaheuristics (Crainic 2019). We refer to Silberholz et al. (2019) for a thorough discussion of how to experimentally evaluate the performance of metaheuristics.

Matheuristics were a further development combining advances in algorithm design with the much increased performance of mathematical programming solvers. There is not yet an organized body of literature on the topic, but the seminal paper that proposed the Local Branching method (Fischetti and Lodi 2003) is an excellent introduction to the topic. Furthermore, Local Branching is widely applicable.

After these general comments on solution methods, let us discuss how they were applied to network design problems.

With respect to the FCTP, there are few papers discussing the application of Tabu Search to this problem. The presentation of Sect. 4.1.1 is based on the paper of Sun et al. (1998), which is a standard reference. There are, however, a fairly large number of papers discussing the application of Genetic Algorithms to the problem, but many of these do not deal with the basic version of the problem. Early efforts aimed at using GAs to address the FCTP include the methods proposed by Gottlieb and Paulmann (1998), which examine two possible representations of solutions. Several papers then proposed GAs based on spanning trees to solve the basic FCTP, as well as some of its variants, for instance the FCTP with nonlinear costs (Jo et al. 2007; Hajiaghaei-Keshteli et al. 2010; Molla-Alizadeh-Zavardehi et al. 2014) and the so-called "Step Fixed-Charge Transportation Problem", which is an extension of the FCTP in which different levels of fixed costs may be incurred on an edge depending on the flow that it carries (Molla-Alizadeh-Zavardehi et al. 2014). The presentation of Sect. 5.1.1 refers to the paper by Lofti and Tavakkoli-Moghaddam (2013), which presents a very effective method.

Regarding the various methods for solving the multicommodity capacitated fixed-charge network design problem (MCFND), it became obvious in the early '90's that metaheuristics, such as Tabu Search, were interesting approaches for tackling it. At first, as mentioned in Sect. 2.2, on the basis of the successful applications of TS to complex location problems (see, e.g., Crainic et al. 1993) based on the exploration of the search space of binary location (i.e., design) variables using the add-drop and swap neighborhood structures, it even seemed attractive to contemplate methods based on these neighborhood structures for the MCFND. As we explained in the discussion on neighborhoods, this approach was quite unsatisfactory. The two TS approaches described in Sect. 4.1.2 are based on the papers by Crainic et al. (2000) and Ghamlouche et al. (2003); they were among the best methods available when they were published. Among the other neighborhood-based metaheuristics, we must mention an interesting application of Variable Neighborhood Search to a network design problem with relays (Xiao and Konak 2017).

On the side of population-based methods, the GA described in Sect. 5.1.2 comes from Crainic and Gendreau (1999). It was proposed as part of the cooperative search combining TS and GA solvers described in Sect. 7.

Sections 5.2 and 5.3 present respectively the methods proposed in Ghamlouche et al. (2004) and in Crainic and Gendreau (2007). Another Scatter Search heuristic was proposed in 2005, but for a slightly different version of the MCFND: the undirected variant, in which flows on opposite directions on a given edge must share the same capacity (Alvarez et al. 2005). In this Scatter Search heuristic, solutions are represented as blocks of paths for each origin-destination pair (commodity), such that these paths can handle all the demand for that commodity. The initial pool of solutions is generated using a GRASP procedure. This pool contains, as is usual in Scatter Search implementations, a mixture of good and diverse solutions. The procedure to construct a new solution from a subset taken from the reference set proceeds on a commodity by commodity basis, choosing at each step the block of paths from the solutions in the subset with the lowest cost. A repair procedure is used to restore feasibility. Several strategies are considered for managing the reference set. The method performs well for instances with small numbers of commodities (10 to 50), but performance degrades for larger instances (i.e., 100 commodities).

The improved Scatter Search approach presented in Sect. 5.3.2 has been proposed by Paraskevopoulos et al. (2016). This is the most recent among the various methods that we presented for the MCFND and it yields very good results.

The four matheuristics described in Sect. 6 were first presented, respectively, in Rodríguez-Martin and Salazar-González (2010); Hewitt et al. (2010); Yaghini et al. (2013), and Yaghini et al. (2015). These four methods are competitive and provide the top results. Furthermore, the matheuristic of Hewitt et al. (2010) scales quite well and is capable of handling larger instances than many other approaches. The method that achieves the best results on the benchmark instances is the one by Yaghini et al. (2015). Among sequential methods, it represents the state-of-the-art.

Other recent matheuristics developed for the network design problem with balancing constraints can be found in Vu et al. (2013); Chouman and Crainic (2015). In this problem, an additional constraint that forces both arcs for each pair of arcs (i, j), (j, i) to be either open or closed is added to the model and has a strong impact on the solutions. The first method combines a Tabu Search based on an extension of the cycle-based neighborhood described in 4.1.2, a Path Relinking procedure, which exploits a path-exchange neighborhood, and an exact solution algorithm that is applied to a restriction of the original instance to perform search intensification. The second approach relies on two main elements that can be implemented in the context of a MIP solver: (1) a procedure for generating cutting planes that allows for the efficient computation of tight lower bounds (similar to the approach of Yaghini et al. 2015) , and (2) a variable-fixing procedure to make the resulting restricted problem solvable by a MIP solver in reasonable time. A key feature of the approach is the use of learning mechanisms and memories to record important information about the solution process and to orient it towards promising areas of the solution space. Computational experiments on a large set of test instances confirmed the efficiency of the proposed approach and the high quality of the solutions that it produces.

Before leaving the topic of matheuristics, it is important to mention the large body of litterature that deals with heuristics that rely on mathematical programming formulations. These include among others the so-called *slope scaling* methods that have been proposed by several authors. These methods are discussed in Chap. 3.

There is a long history of successful developments of parallel heuristics and metaheuristics, a certain number of those contributions targeting network design formulations. A number of surveys, taxonomies, and syntheses present a general view of the topic, including Alba (2005); Alba et al. (2013); Crainic and Hail (2005); Crainic (2008); Crainic and Toulouse (2010); Crainic et al. (2014); Crainic (2019); Cung et al. (2002); Melab et al. (2006); Pedemonte et al. (2011); Schryen (2020); Talbi (2009); Talukdar et al. (2003). Performance measures for parallel metaheuristics are discussed in most of these books, as well as in Barr and Hickman (1993); Crainic and Toulouse (1998, 2003).

All surveys and books include a taxonomy characterizing the parallel strategies for metaheuristics. They agree on most counts, even though terms might differ. Thus, fine- and coarse-grained decomposition, low-level and functional parallelism, domain decomposition, diffusion, as well as synchronous and asynchronous communications are examples of common concepts and terms. While also found broadly, search-space decomposition, multi-search parallelism, and cooperative search may be encountered under different names. Thus, for example, *memory*, *pool*, and *data warehouse* (*reference* and *elite set* are also sometimes used) are equivalent terms found in the parallel metaheuristic literature for the sharing data structures of cooperative search (*blackboard* is often used in the computer-science and artificial-intelligence vocabulary). We use the taxonomy and vocabulary of Crainic and Hail (2005), generalizing Crainic et al. (1996, 1997).

Munguía et al. (2017) proposed a search-space decomposition with restarts matheuristic for the MCFND, which inspired the description of Sect. 7. The authors implement a large neighborhood structure explored with a MIP solver. The Local Search explores each partition defined by a group of commodities, by applying this move (i.e., solving exactly) to each combination of decreasing cardinality of those commodities. Several partitions may be obtained by controlling the graph-partitioning algorithm. A solution-recombination method proceeding in stages (two in the implementation described), by grouping iteratively neighboring partitions (which increases the number of variables one may fix to 0) appears particularly interesting. The authors explored various implementations on different computer architectures and obtained very good results. The commodity-based partitioning and the multi-stage solution-recombination mechanisms are noteworthy as they could be combined with learning mechanisms and cooperative search strategies.

We refer the reader to Crainic (2019) for a detailed discussion relative to cooperative multi-search metaheuristics and the associated literature, and to Crainic et al. (1996, 1997) and Toulouse et al. (1996, 1999a, 2004, 1998, 2000) for the issues related to defining mechanisms and parameters for cooperation, including the quality, meaningfulness, and parsimony principles of information sharing.

Memory-based cooperative pC/C search strategies are described in the literature for most metaheuristic classes and combinatorial optimization problems. To the best of our knowledge, Crainic et al. (1996) were the first to propose a memory-based approach for asynchronous Tabu Search in their study of a multicommodity location problem with balancing requirements. The proposed method outperformed in terms of solution quality the sequential version as well as several synchronous and broadcast-based asynchronous cooperative strategies. The method was extended to address the MCFND with similar results (Crainic and Gendreau 2002). The illustrations of pC/C multi-search strategies of Sect. 7 are inspired by the work of Ribeiro and Rosseti (2007) on pC/C GRASP for 2-path network design, and the method of Crainic et al. (1996) and Crainic and Gendreau (2002), respectively.

Multi-level Cooperative Search (Toulouse et al. 1999b) produced excellent results for various problem settings, including graph and hypergraph partitioning (Ouyang et al. 2000, 2002), feature selection in biomedical data (Oduntan et al. 2008), and covering design (Dai et al. 2009), which are close to network design or may be of interest for parallel metaheuristic strategies (e.g., graph partitioning). The network design illustration of Sect. 7 is based is based on Crainic et al. (2006b).

The many contributions found in the literature show that centralized-memory pC/C asynchronous cooperation strategies are generally offering very good results, yielding high-quality solutions. They are also computationally efficient, with no synchronization overhead, no broadcasting, and no need for complex mechanisms to select the solvers that will receive or send information and to control the cooperation. pC/C strategies have also proved efficient in handling the issue of premature "convergence" in cooperative search, by diversifying the information received by solvers through probabilistic selection from the memory and by a somewhat large and diverse population of solutions in the memory; solvers may thus import different solutions even when their cooperation activities are taking place within a short time span. Such good performances and the availability of shared information kept in the memory has brought the question of whether one could design more advanced cooperation mechanisms taking advantage of the information exchanged among cooperating solvers. The pC/KC strategies are the result of this area of research.

The adaptive-memory terminology was coined by Rochat and Taillard (1995) proposing Tabu Search-based heuristics for vehicle routing. Central-memory methods initiated with the research of Crainic et al. (1996, 1997) for multicommodity location problem with balancing requirements and its extension to the MCFND (Crainic and Gendreau 2002). The basic canvas for central-memory strategies of Sect. 7 comes from those developments, as well as the study of Crainic and Gendreau (1999). The pattern-building learning and guidance mechanisms are inspired by the work of Le Bouthillier et al. (2005) for the vehicle routing problem with time windows (see Le Bouthillier 2007, for a dynamic version of this mechanism).

According to our best knowledge, Crainic et al. (2006a) (see also Di Chiara 2006) were the first to propose an ICS method in the context of designing wireless networks, where seven attributes were considered simultaneously. The proposed pC/KC metaheuristic has Tabu Search solvers address limited subsets of attributes, the others being fixed, and a GA combine the partial solutions generated by solvers into complete solutions to the initial problem. A formal definition and evaluation of ICS (in the vehicle routing context) is to be found in Lahrichi et al. (2015).

9 Conclusions and Perspectives

One of the main things that comes to mind when considering heuristic solution procedures for network design problems is the realization that there has been a very steady improvement in the performance of these methods over time: state-of-the-art methods are now capable of producing solutions that are quite close to optimal ones in a fraction of the time required by exact approaches. This can be very important in some practical settings. It is also interesting to note that some of the most recent methods scale up rather well, which means that they are able to tackle instances of significant size, similar to the ones that one is likely to encounter in many practical applications.

These conclusions can be traced, in our opinion, to three main causes. The first is the proper exploitation of the mathematical properties of optimal solutions of fixed-charge problems, namely the fact that, in most cases, they can be found at extreme points of the feasible set. This property has had a deep impact on the definition of search spaces for many methods, making these much more effective. The second reason is the amazing developments in the performance of MIP solvers, which has allowed for the integration of dedicated exact solution procedures within many metaheuristics and matheuristics. The third reason, but not the least, are the advances in the design of solution procedures, which now display a sophisticated architecture to deal with the various challenges of network design problems. It is the combination of these three factors that explain where we stand now in our efforts to tackle these problems.

This having been said, it is important to state that a lot remains to be done: there are indeed several intriguing approaches that one might wish to consider for tackling network design problems. Among these, the Unified Hybrid Genetic Search (UHGS), which combines the exploitation of a carefully managed population of solutions with neighborhood-based search, immediately comes to mind considering the successes obtained by this method on a wide range of difficult vehicle routing problems (see, e.g., Vidal et al. 2012, 2014). Combining the ideas of UHGS with tailored exact solution procedures could lead to very powerful matheuristics. A second fascinating perspective is that of parallel and cooperative search, particularly in the ICS version. Indeed, combining various decomposition strategies, along different dimensions of design problems, with state-of-the-art matheuristics for the resulting problems and the recombination of solutions presents fascinating challenging and very promising perspectives. This is also an area where advanced learning mechanisms find a natural niche to extend the classical memory-based capabilities of metaheuristics.

Last but not least, there is the challenge of metaheuristics and matheuristics for the many variants of network design problems brought by applications. The other chapters of this book present these models and identify the associated algorithmic challenges and perspectives.

References

Alba, E. (Ed.) (2005). *Parallel metaheuristics: A new class of algorithms*. Hoboken, NJ: Wiley.

Alba, E., Luque, G., & Nesmachnow, S. (2013). Parallel metaheuristics: recent advances and new trends. *International Transactions in Operational Research, 20*(1), 1–48.

Alvarez, A. M., González-Velarde, J. L., & De-Alba, K. (2005). Scatter search for network design problem. *Annals of Operations Research, 138*, 159–178.

Barr, R. S., & Hickman, B. L. (1993). Reporting computational experiments with parallel algorithms: issues, measures, and experts opinions. *ORSA Journal on Computing, 5*(1), 2–18.

Baxter, J. (1981). Local optima avoidance in depot location. *Journal of the Operational Research Society, 32*, 815–819.

Chouman, M., & Crainic, T. G. (2015). Cutting-plane matheuristic for service network design with design-balanced requirements. *Transportation Science 49*(1), 99–113.

Chouman, M., Crainic, T. G., & Gendron, B. (2009). A cutting-plane algorithm for multi-commodity capacitated fixed charge network design. Tech. Rep. CIRRELT-2009-20, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et les transports, Université de Montréal, Montréal, QC, Canada.

Cooper, L. (1975). The fixed charge problem-I: A new heuristic method. *Computers & Mathematics with Applications, 1*, 89–95.

Cooper, L., & Drebes, C. (1967). An approximate solution method for the fixed charge problem. *Naval Research Logistics Quarterly, 14*, 101–113.

Crainic, T. G. (2008). Parallel solution methods for vehicle routing problems. In B. L.Golden, S. Raghavan, & E. A. Wasil (Eds.). *The vehicle routing problem: latest advances and new challenges* (pp. 171–198). New York, NY: Springer.

Crainic, T. G. (2019). Parallel metaheuristics and cooperative search. In M. Gendreau, & J.-Y. Potvin (Eds.), *Handbook of Metaheuristics* (3rd ed., pp. 419–451). Berlin: Springer.

Crainic, T.G., & Gendreau, M. (1999). Towards an evolutionary method—cooperating multi-thread parallel tabu search hybrid. In S. Voß, S. Martello, C. Roucairol, & I. H. Osman (Eds.), *Meta-heuristics 98: Theory and applications* (pp. 331–344). Norwell, MA: Kluwer Academic Publishers.

Crainic, T.G., & Gendreau, M. (2002). Cooperative parallel tabu search for capacitated network design. *Journal of Heuristics, 8*(6), 601–627.

Crainic, T.G., & Gendreau, M. (2007) A scatter search heuristic for the fixed-charge multicommodity flow network design problem. In K. Doerner, M. Gendreau, P. Greistorfer, W. J. Gutjahr, R. F. Hartl, & M. Reimann (Eds.), *Metaheuristics—progress in complex systems optimization* (pp. 25–40). New York, NY: Springer.

Crainic, T. G., & Hail, N. (2005). Parallel meta-heuristics applications. In E. Alba (Ed.), *Parallel metaheuristics: A new class of algorithms* (pp. 447–494). Hoboken, NJ: Wiley

Crainic, T. G., & Toulouse, M. (1998). Parallel metaheuristics. In T. G. Crainic, & G. Laporte (Eds.), *Fleet management and logistics* (pp. 205–251). Norwell, MA: Kluwer Academic Publishers.

Crainic, T. G., & Toulouse, M. (2003). Parallel strategies for meta-heuristics. In F. Glover, & G. Kochenberger (Eds.), *Handbook in metaheuristics* (pp. 475–513). Norwell, MA: Kluwer Academic Publishers.

Crainic, T. G., & Toulouse, M. (2010) Parallel meta-heuristics. In M. Gendreau, J.-Y. Potvin (Eds.). *Handbook of metaheuristics* (2nd ed., pp. 497–541). Berlin: Springer.

Crainic, T. G., Gendreau, M., Soriano, P., & Toulouse, M. (1993). A Tabu search procedure for multicommodity location/allocation with balancing requirements. *Annals of Operations Research, 41*, 359–383.

Crainic, T. G., Toulouse, M., & Gendreau, M. (1996). Parallel asynchronous Tabu search for multicommodity location-allocation with balancing requirements. *Annals of Operations Research, 63*, 277–299.

Crainic, T. G., Toulouse, M., & Gendreau, M. (1997). Towards a taxonomy of parallel tabu search algorithms. *INFORMS Journal on Computing, 9*(1), 61–72.

Crainic, T. G., Gendreau, M., & Farvolden, J. M. (2000). A simplex-based Tabu search method for capacitated network design. *INFORMS Journal on Computing, 12*(3), 223–236.

Crainic, T. G., Di Chiara, B., Nonato, M., & Tarricone, L. (2006a) Tackling electrosmog in completely configured 3G networks by parallel cooperative meta-heuristics. *IEEE Wireless Communications, 13*(6), 34–41.

Crainic, T. G., Li, Y., & Toulouse, M. (2006b). A first multilevel cooperative algorithm for the capacitated multicommodity network design. *Computers & Operations Research, 33*(9), 2602–2622.

Crainic, T. G., Davidović, T., & Ramljak, D. (2014). Designing parallel meta-heuristic methods. In M. Despotovic-Zrakic, V. Milutinovic, & A. Belic (Eds.), *High performance and cloud computing in scientific research and education* (pp. 260–280). Hershey, PA: IGI Global

Cung, V. D., Martins, S. L., Ribeiro, C. C., & Roucairol, C. (2002). Strategies for the parallel implementations of metaheuristics. In C. Ribeiro, P. Hansen (Eds.), *Essays and surveys in metaheuristics* (pp. 263–308). Norwell, MA: Kluwer Academic Publishers.

Dai, C., Li, B., & Toulouse, M. (2009). A multilevel cooperative Tabu search algorithm for the covering design problem. *Journal of Combinatorial Mathematics and Combinatorial, Computing 68*, 35–65.

Delahaye, D., Chaimatanan, S., & Mongeau, M. (2019) Simulated annealing: From basics to applications. In M. Gendreau, J.-Y. Potvin (Eds.), *Handbook of metaheuristics* (3rd ed., pp 1–35). Cham: Springer.

Denzler, D. R. (1969). An approximate algorithm for the fixed charge problem. *Naval Research Logistics Quarterly, 16*, 411–416.

Di Chiara, B. (2006). Optimum planning of 3G cellular systems: Radio propagation models and cooperative parallel meta-heuristics. PhD thesis, Dipartimento di ingegneria dell'innovatione, Universitá degli Studi di Lecce, Lecce, Italy.

Feo, T. A., & Resende, M. G. C. (1989). A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters, 8*, 67–71.

Feo, T. A., & Resende, M. G. C. (1995) Greedy randomized adaptive search procedures. *Journal of Global Optimization, 6*, 109–134.

Fischetti, M., & Lodi, A. (2003). Local branching. *Mathematical Programming Series B, 98*, 23–47.

Gendreau, M., & Potvin, J.-Y. (2019). Tabu search. In M. Gendreau, J.-Y. Potvin (Eds.), *Handbook of Metaheuristics* (3rd ed., pp. 37–55). Cham: Springer.

Gendreau, M., Hertz, A., & Laporte, G. (1994). A Tabu search heuristic for the vehicle routing problem. *Management Science, 40*(10), 1276–1290.

Gendron, B., & Crainic, T. G. (1994). Relaxations for multicommodity network design problems. Publication CRT-965, Centre de recherche sur les transports, Université de Montréal, Montréal, QC, Canada.

Ghamlouche, I., Crainic, T. G., & Gendreau, M. (2003). Cycle-based neighbourhoods for fixed-charge capacitated multicommodity network design. *Operations Research, 51*(4), 655–667.

Ghamlouche, I., Crainic, T. G., & Gendreau, M. (2004). Path relinking, cycle-based neighbourhoods and capacitated multicommodity network design. *Annals of Operations Research, 131*, 109–133.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research, 1*(3), 533–549

Glover, F. (1989). Tabu search—part I. *ORSA Journal on Computing, 1*(3), 190–206.

Glover, F. (1990). Tabu search—part II. *ORSA Journal on Computing, 2*(1), 4–32.

Glover, F. (1997). A template for scatter search and path relinking. In J. Hao, E. Lutton, E. Ronald, M. Schoenauer, & D. Snyers (Eds.), *Artificial evolution, lecture notes in computer science* (Vol. 1363, pp. 13–54). Berlin: Springer.

Glover, F., & Kochenberger, G. (Eds.) (2003). *Handbook of metaheuristics*. Norwell, MA: Kluwer Academic Publishers.

Glover, F., & Laguna, M. (1993). Tabu search. In C. Reeves (Ed.), *Modern heuristic techniques for combinatorial problems* (pp. 70–150). Oxford: Blackwell Scientific Publications.

Glover, F., & Laguna, M. (1997) *Tabu search*. Norwell, MA: Kluwer Academic Publishers.

Glover, F., Laguna, M., & Martí, R. (2000). Fundamentals of scatter search and path relinking. *Control and Cybernetics, 39*(3), 653–684.

Gottlieb, J., & Paulmann, L. (1998). Genetic algorithms for the fixed charge transportation problem. In *Proceedings of the 1998 IEEE International Conference on Evolutionary Computation* (pp. 330–335)

Hajiaghaei-Keshteli, M., Molla-Alizadeh-Zavardehi, S., & Tavakkoli-Moghaddam, R. (2010) Addressing a nonlinear fixed charge transportation problem using a spanning tree based genetic algorithm. *Computers & Industrial Engineering, 59*, 259–271.

Hansen, P., Mladenović, N. (1999). An introduction to variable neighborhood search. In S. Voß, S. Martello, C. Roucairol, & I. H. Osman (Eds.), *Meta-heuristics 98: Theory & applications* (pp. 433–458). Norwell, MA: Kluwer Academic Publishers.

Hansen, P., Mladenović, N., Brimberg, J., Pérez, J. A. M. (2019). Variable neighborhood search. In: M. Gendreau, J.-Y. Potvin (Eds.), *Handbook of metaheuristics* (3rd ed., pp. 57–97). Cham: Springer.

Hewitt, M., Nemhauser, G. L., & Savelsbergh, M. W. (2010) Combining exact and heuristic approaches for the capacitated fixed-charge network flow problem. *INFORMS Journal on Computing, 22*, 314–325.

Hirsch, W. M., & Dantzig, G. B. (1954) *Notes on linear programming part XIX, the fixed charge problem. Memorandum* (Vol. 1383). Santa Monica, CA: Rand Research

Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor, MI: University of Michigan Press

Jo, J. B., Li, Y., & Gen, M. (2007). Nonlinear fixed charge transportation problem by spanning tree-based genetic algorithm. *Computers & Industrial Engineering, 53*, 290–298.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983) Optimization by simulated annealing. *Science, 220*, 671–680.

Lahrichi, N., Crainic, T. G., Gendreau, M., Rei, W., Crisan, G. C., & Vidal, T. (2015) An integrative cooperative search framework for multi-decision-attribute combinatorial optimization. *European Journal of Operational Research, 246*(2), 400–412.

Le Bouthillier, A. (2007). Recherches coopératives pour la résolution de problèmes d'optimisation combinatoire. PhD thesis, Département d'informatique et de recherche opérationnelle, Université de Montréal, Montréal, QC, Canada.

Le Bouthillier, A., Crainic, T. G., & Kropf, P. (2005). A guided cooperative search for the vehicle routing problem with time windows. *IEEE Intelligent Systems, 20*(4), 36–42.

Lofti, M. M., & Tavakkoli-Moghaddam, R. (2013). A genetic algorithm using priority-based encoding with new operators for fixed charge transportation problems. *Applied Soft Computing, 13*(5), 2711–2726.

Lourenço, H. R., Martin, O. C., & Stützle, T. (2019). Iterated local search: Framework and applications. In: M. Gendreau, & J.-Y. Potvin (Eds.), *Handbook of metaheuristics* (3rd ed., pp. 129–168). Cham: Springer.

Melab, N., Talbi, E. G., Cahon, S., Alba, E., & Luque, G. (2006). Parallel metaheuristics: Models and frameworks. In E.L. Ghazali Talbi (Ed.), *Parallel combinatorial optimization* (pp. 149–162). New York, NY: Wiley.

Mladenović, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research, 24*:1097–1100

Molla-Alizadeh-Zavardehi, S., Mahmoodirad, A., & Rahimian, M. (2014). Step fixed charge transportation problems. *Indian Journal of Science and Technology, 7*(7), 949–954.

Munguía, L. M., Ahmed, S., Bader, D. A., Nemhauser, G. L., Goel, V., & Shao, Y. (2017). A parallel local search framework for the fixed-charge multicommodity network flow problem. *Computers & Operations Research, 77*, 44–57.

Oduntan, I. O., Toulouse, M., Baumgartner, R., Bowman, C., Somorjai, R., & Crainic, T. G. (2008). A multilevel Tabu search algorithm for the feature selection problem in biomedical data sets. *Computers & Mathematics with Applications, 55*(5), 1019–1033.

Ouyang, M., Toulouse, M., Thulasiraman, K., Glover, F., & Deogun, J. S. (2000). Multi-level cooperative search: Application to the netlist/hypergraph partitioning problem. In *Proceedings of International Symposium on Physical Design* (pp. 192–198). New York, NY: ACM Press.

Ouyang, M., Toulouse, M., Thulasiraman, K., Glover, F., & Deogun, J. S. (2002). Multilevel cooperative search for the circuit/hypergraph partitioning problem. *IEEE Transactions on Computer-Aided Design, 21*(6), 685–693.

Paraskevopoulos, D. C., Bektaş, T., Crainic, T. G., & Potts, C. N. (2016). A cycle-based evolutionary algorithm for the fixed-charge capacitated multi-commodity network design problem. *European Journal of Operational Research, 253*(1), 265–279.

Pedemonte, M., Nesmachnow, S., & Cancela, H. (2011). A survey of parallel ant colony optimization. *Applied Soft Computing, 11*(8), 5181–5197.

Resende, M. G. C., & Ribeiro, C. C. (2019). Greedy randomized adaptive search procedures: Advances and extensions. In M. Gendreau, & J.-Y. Potvin (Eds.), *Handbook of Metaheuristics* (3rd ed., pp. 169–220). Cham: Springer.

Ribeiro, C. C., & Rosseti, I. (2007). Efficient parallel cooperative implementations of GRASP heuristics. *Parallel Computing, 33*(1), 21–35.

Rochat, Y., & Taillard, E. D. (1995). Probabilistic diversification and intensification in local search for vehicle routing. *Journal of Heuristics, 1*(1), 147–167.

Rodríguez-Martin, I., & Salazar-González, J. J. (2010). A local branching heuristic for the capacitated fixed-charge network design problem. *Computers & Operations Research, 37*, 575–581.

Schryen, G. (2020). Parallel computational optimization in operations research: A new integrative framework, literature review and research directions. *European Journal of Operational Research, 287*(1), 1–18.

Silberholz, J., Golden, B., Gupta, S., & Wang, X. (2019). Computational comparison of metaheuristics. In M. Gendreau, & J.-Y. Potvin (Eds.). *Handbook of metaheuristics* (3rd ed., pp. 581–604). Cham: Springer.

Steinberg, D. I. (1970). The fixed charge problem. *Naval Research Logistics Quarterly, 17*, 217–236.

Sun, M., Aronson, J. E., McKeown, P. G., & Drinka, D. (1998). A Tabu search procedure for the fixed charge transportation problem. *European Journal of Operational Research, 106*, 441–446.

Talbi, E. G. (Ed.) (2009). *Metaheuristics: From design to implementation*. Hoboken, NJ: Wiley.

Talukdar, S., Murthy, S., & Akkiraju, R. (2003). Assynchronous teams. In F. Glover, & G. Kochenberger (Eds.), *Handbook in metaheuristics* (pp. 537–556) Norwell, MA: Kluwer Academic Publishers.

Toulouse, M., Crainic, T. G., & Gendreau, M. (1996). Communication issues in designing cooperative multi thread parallel searches. In I. H. Osman, & J. P. Kelly (Eds.), *Meta-heuristics: Theory & applications* (pp. 501–522). Norwell, MA: Kluwer Academic Publishers.

Toulouse, M., Crainic, T. G., Sansó, B., & Thulasiraman, K. (1998). Self-organization in cooperative search algorithms. In *Proceedings of the 1998 IEEE International Conference on Systems, Man, and Cybernetics* (pp. 2379–2385). Madisson, WI: Omnipress.

Toulouse, M., Crainic, T. G., & Sansó, B. (1999a). An experimental study of systemic behavior of cooperative search algorithms. In S. Voß, S. Martello, C. Roucairol, & I. H. Osman (Eds.), *Meta-heuristics 98: Theory & applications* (pp. 373–392). Norwell, MA: Kluwer Academic Publishers.

Toulouse, M., Thulasiraman, K., & Glover, F. (1999b). Multi-level cooperative search: a new paradigm for combinatorial optimization and an application to graph partitioning. In P. Amestoy, P. Berger, M. Daydé, I. Duff, V. Frayssé, L. Giraud, & D. Ruiz (Eds.), *Fifth International Euro-Par Parallel Processing Conference, Lecture Notes in Computer Science* (Vol. 1685, pp. 533–542). Heidelberg: Springer.

Toulouse, M., Crainic, T. G., & Thulasiraman, K. (2000). Global optimization properties of parallel cooperative search algorithms: A simulation study. *Parallel Computing, 26*(1), 91–112.

Toulouse, M., Crainic, T. G., & Sansó, B. (2004). Systemic behavior of cooperative search algorithms. *Parallel Computing, 30*(1), 57–79.

Vidal, T., Crainic, T. G., Gendreau, M., Lahrichi, N., & Rei, W. (2012). A hybrid genetic algorithm for multi-depot and periodic vehicle routing problems. *Operations Research, 60*(3), 611–624.

Vidal, T., Crainic, T. G., Gendreau, M., & Prins, C. (2014). A unified solution framework for multi-attribute vehicle routing problems. *European Journal of Operational Research, 234*(3), 658–673.

Vu, D. M., Crainic, T. G., & Toulouse, M. (2013). A three-stage matheuristic for the capacitated multi-commodity fixed-cost network design with design-balance constraints. *Journal of Heuristics, 19*, 757–795.

Walker, W. E. (1976). A heuristic adjacent extreme point algorithm for the fixed charge problem. *Management Science, 22*, 587–596.

Whitley, D. (2019). Next generation genetic algorithms: A user's guide and tutorial. In M. Gendreau, & J.-Y. Potvin (Eds.), *Handbook of metaheuristics* (3rd ed., pp. 245–274). Cham: Springer.

Xiao, Y., & Konak, A. (2017). A variable neighborhood search for the network design problem with relays. *Journal of Heuristics, 23*, 137–164.

Yaghini, M., Rahbar, M., & Karimi, M. (2013). A hybrid simulated annealing and column generation approach for capacitated multicommodity network design. *Journal of the Operational Research Society, 64*, 1010–1020.

Yaghini, M., Karimi, M., Rahbar, M., Sharifitabar, H. (2015). A cutting-plane neighborhood structure for fixed-charge capacitated multicommodity network design problem. *INFORMS Journal on Computing, 27*(1), 48–58.

Part II
Advanced Problems and Models

Chapter 5
Multicommodity Multifacility Network Design

Alper Atamtürk and Oktay Günlük

1 Introduction

Here we consider multicommodity network design models, where capacity can be added to arcs of the network using integer multiples of facilities, possibly with varying capacities. This class of models appear frequently in telecommunication network capacity expansion problems, train scheduling with multiple locomotive options, supply chain and service network design problems. In the single-facility network design problem, one installs multiples of only a single type of facility on the arcs of the network. Routing vehicles with identical capacity in a logistics network and installing a communication network with only one cable type are examples of the single-facility network design problem. In the multifacility problem, one may install different types of facilities with varying capacities, such as fiberoptic cables with varying bandwidths, production lines or machines with different rates, or a fleet of heterogeneous vehicles with varying capacities. The optimization problem seeks to decide how many facilities of each type to install on the network so as to meet the demand for each commodity at the least cost. We present the precise problem description and the associated formulation in the next section.

Different versions of the problem are obtained depending on how the flow is routed in the network. In the *unsplittable* flow version, only a single path is allowed to route the flow from its source to its destination, which requires integer variables to

A. Atamtürk
Department of Industrial Engineering and Operations Research, University of California, Berkeley, CA, USA
e-mail: atamturk@berkeley.edu

O. Günlük (✉)
School of Information Engineering and Operations Research, Cornell University, Ithaca, NY, USA
e-mail: oktay.gunluk@cornell.edu

model its route. This is the case, for instance, in telecommunication networks using multiprotocol label switching (MPLS) technique, production and distribution with single sourcing, and express package delivery. The *splittable* case, which assumes that flow can be routed using multiple directed paths, is obviously a relaxation of the unsplittable case; therefore, valid inequalities for the splittable case are also valid for the unsplittable case.

In addition, the capacity created by the facilities can be *directed*, *bidirected*, or *undirected*. In the bidirected case, if a certain facility is installed on an arc, then the same facility also needs to be installed on the reverse arc. In the undirected case, the total flow on an arc and its reverse arc is limited by the capacity of the undirected edge associated with the two arcs. Here we consider the directed case where the total flow on an arc is limited by the total (directed) capacity of the arc. In a recent paper, the authors of this chapter describe how to generate valid inequalities for the bidirected and undirected network design problems using valid inequalities for the directed case.

In this chapter we review strong valid inequalities for the multicommodity multifacility network design problem. Throughout, we emphasize three fundamental techniques that have been effective in deriving strong inequalities for network design problems. These are the metric inequalities for projecting out the continuous flow variables, mixed-integer rounding from appropriate base relaxations, and shrinking the network to a small k-node graph. The valid inequalities for the network design problem are obtained by applying these techniques to different relaxations of the problem, namely, arc-set, cut-set and partition relaxations. The basic inequalities derived from these relaxations are also utilized, with certain adaptations, in robust and survivable network design problems.

This chapter is organized as follows. In the next section, we introduce the notation used in the chapter and give a formal definition of the multicommodity multifacility design problem. Section 3 reviews the metric inequalities for projecting out the multicommodity flow variables, the mixed-integer rounding technique, as well as a simplification obtained by shrinking the network for deriving valid inequalities from a smaller network. These techniques play a central role in deriving strong valid inequalities for the network design problem. Section 4 reviews the inequalities from single arc capacity constraints for the splittable as well as the unsplittable cases. Section 5 reviews the valid inequalities from two-partitions for single as well as multifacility cases. Section 6 generalizes the inequalities in the previous section to a higher number of partitions.

2 Problem Formulation

Let $G = (N, A)$ be a directed graph (network), with node set N and arc set $A \subseteq N \times N$. Each arc $a \in A$ has a given existing capacity $\bar{c}_a \geq 0$ and the network design problem consists of installing additional capacity on the arcs of the network using an (integral) combination of a given set of capacity types. The objective of the problem is to minimize the sum of the flow routing cost and the capacity expansion cost. Throughout we assume that the data is rational.

The demand data for the problem is given by matrix $T = \{t_{ij}\}$, where $t_{ij} \geq 0$ is the amount of (directed) traffic that should be routed from node $i \in N$ to $j \in N$. Using matrix T, we can define a collection of commodities, each of which has a certain supply and demand at each node of the network. As discussed in Chap. 2, there are different ways to formulate the same network design problem by changing what is meant by a commodity. Using a minimal vertex cover of the so-called demand graph, one can obtain the smallest number of commodities required to formulate a given problem instance correctly. Computationally, formulations with fewer commodities may be more desirable as their continuous relaxations solve faster. For the sake of concreteness, we will use the aggregated commodity description but we note that most of the discussion below does not depend on how the commodities are defined. Let $K \subseteq N$ denote the collection of nodes with positive supply, i.e.,

$$K = \left\{ i \in N : \sum_{j \in N} t_{ij} > 0 \right\}.$$

We use w_i^k to denote the net demand of node $i \in N$ for commodity $k \in K$. More precisely, let $w_i^k = t_{ki}$ for $i \neq k$ and $w_k^k = -\sum_{j \in N} t_{kj}$ for $k \in K$. Note that each node $k \in K$ is the unique supplier of commodity k and flow of each commodity in the network needs to be disaggregated to obtain an individual routing for origin-destination pairs. We also note that the aggregated commodity description can only be used if the flow routing cost does not depend on both the origin and the destination of the traffic that needs to be routed.

New capacity can be installed in the network using integer multiples of facilities M, where a single unit of facility $m \in M$ provides capacity c_m. Without loss of generality, we assume that $c_m \in \mathbb{Z}$ for all $m \in M$ and $c_1 < c_2 < \cdots < c_{|M|}$. In this setting the network design problem involves installing enough additional capacity on the arcs of the network so that traffic can be routed simultaneously without violating arc capacities. For $i \in N$, let

$$N_i^+ = \{j \in N : (i, j) \in A\} \text{ and } N_i^- = \{j \in N : (j, i) \in A\}$$

denote the neighbors of node $i \in N$. Let integer variables $y_{ma} \geq 0$ indicate the number of facilities of type $m \in M$ installed on arc $a \in A$ and continuous variables $x_a^k \geq 0$ denote the amount of flow of commodity $k \in K$ routed on arc $a \in A$. Using this notation, the following constraints define the feasible region of the multicommodity multifacility network design problem:

$$\sum_{j \in N_i^+} x_{ij}^k - \sum_{j \in N_i^-} x_{ji}^k = w_i^k, \qquad k \in K,\ i \in N, \tag{5.1}$$

$$\sum_{k\in K} x_{ij}^k - \sum_{m\in M} c_m y_{mij} \leq \bar{c}_{ij}, \qquad (i,j)\in A. \tag{5.2}$$

Then the network design problem is stated as:

$$\text{(NDP)} \qquad \min\left\{dy + fx \ : \ (x,y)\in P^{\text{ND}}\right\},$$

where d and f are cost vectors of appropriate size and

$$P^{\text{ND}} = \text{conv}\left\{(x,y)\in \mathbb{R}_+^{A\times K}\times \mathbb{Z}_+^{A\times M} \ : \ (5.1) \text{ and } (5.2)\right\}.$$

As a concrete example with two facilities, consider a given arc $a \in A$. The total capacity $c_1y_{1a} + c_2y_{2a}$ given by the integer variables y_{1a} and y_{2a} has cost $d_{1a}y_{1a} + d_{2a}y_{2a}$. Assuming economies of scale, let $d_{1a}/c_1 > d_{2a}/c_2$ and remember that $c_1 < c_2$. In this case, we can write the cost function $f(z)$ required to generate z units of capacity (at the least cost) as:

$$h(z) = \lfloor z/c_2\rfloor d_{2a} + \min\{d_{2a}, \lceil (z - \lfloor z/c_2\rfloor c_2)/c_1\rceil d_{1a}\}$$

which is a piecewise linear function. Figure 5.1 illustrates an example with $3d_{1a} < d_{2a} < 4d_{1a}$.

We also note that, when $f = 0$ it is possible to project out the multicommodity flow variables from P^{ND} to obtain a formulation in the space of only the discrete capacity variables. This capacity formulation requires an exponential number of constraints and is discussed in Sect. 3.1.

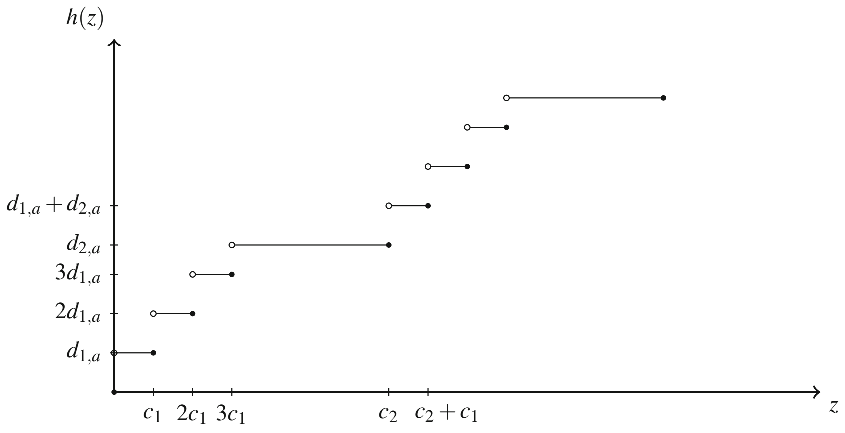


Fig. 5.1 The piecewise linear capacity installation cost function $h(z)$

3 Preliminaries

In this section we discuss three fundamental approaches that are useful in generating strong cutting planes for P^{ND}. We start with the metric inequalities, which give a generalization of the well-known max-flow min-cut theorem to multicommodity flows. We then discuss how valid inequalities can be generated by shrinking the network to one with a few nodes to obtain inequalities from simpler sets. Finally, we describe the mixed-integer rounding procedure, which is an effective method to produce valid inequalities for general mixed-integer sets.

3.1 Metric Inequalities

Metric inequalities are introduced by Iri (1971), and Onaga and Kakusho (1971) as a generalization of the max-flow min-cut duality to multicommodity flows. Consider the following capacitated multicommodity flow set

$$F = \left\{ x \in \mathbb{R}_+^{A\times K} \;:\; (5.1) \text{ and } \sum_{k\in K} x_a^k \le c_a, \quad a \in A \right\},$$

where $c \in \mathbb{R}_+^A$ denotes the arc capacities. By Farkas' Lemma, the set F is non-empty if and only if the following *metric inequalities*

$$\sum_{a\in A} c_a v_a \ge \sum_{i\in N} \sum_{k\in K} w_i^k u_i^k \tag{5.3}$$

hold for all $(v, u) \in D$, where

$$D = \left\{ (v, u) \in \mathbb{R}_+^A \times \mathbb{R}^{N\times K} \;:\; v_{ij} \ge u_j^k - u_i^k, \;\; k \in K, \; (i, j) \in A, \; u_k^k = 0 \right\}.$$

In other words, the arc capacity vector c can accommodate a feasible routing of the commodities if and only if it satisfies all metric inequalities generated by the non-empty cone D. Note that for any fixed $\bar{v} \in \mathbb{R}_+^A$, a point $(\bar{v}, u(\bar{v})) \in D$ maximizes the right-hand-side of (5.3) when $u(\bar{v})$ corresponds to the shortest path lengths using $\bar{v}$ as edge weights (hence the name "metric inequality"). Therefore, it suffices to consider metric inequalities where the vector $u \in \mathbb{R}^{N\times K}$ satisfies this property.

When there is only a single commodity, the max-flow min-cut theorem gives a nice characterization of the important extreme rays of the cone D. More precisely, in this case it suffices to consider vectors $v \in \{0, 1\}^A$, where $v_{ij} = 1$ if and only if $i \in S$ and $j \notin S$ for some $S \subset N$ that contains the source of the commodity but not all the sinks.

Now consider the set P^{ND} and a given capacity vector $y \in \mathbb{R}^{A\times M}$ together with the existing arc capacities $\bar{c}$ and demand w. The metric inequality generated by $(u, v) \in D$ becomes

$$\sum_{a\in A}(\bar{c}_a + \sum_{m\in M} c_m y_{ma})v_a \geq \sum_{i\in N}\sum_{k\in K} w_i^k u_i^k. \tag{5.4}$$

As it is possible to check if there is a violated metric inequality in polynomial time (by solving a linear program), one can project out the flow variables from P^{ND} and obtain a "capacity" formulation in the space of the capacity variables only. Clearly this approach can be applicable only if there is no flow routing cost, i.e., $f = 0$ in problem (NDP). We also note that as inequalities (5.4) do not have flow variables, they only depend on the demand matrix and not on what commodity definition is used for the flow variables. Consequently, the right-hand-side of inequality (5.4) reduces to $\sum_{i\in N}\sum_{k\in N} t_{ki} u_i^k$.

The *integral metric inequalities* are obtained by rounding up the right-hand-side of metric inequalities associated with integral vectors $(u, v) \in D$,

$$\sum_{a\in A}\sum_{m\in M} c_m y_{ma} v_a \geq \left\lceil \sum_{i\in N}\sum_{k\in K} w_i^k u_i^k - \sum_{a\in A} \bar{c}_a v_a \right\rceil$$

The basic cut-set inequalities discussed in Sect. 5 are special cases of the integral metric inequalities. While the metric inequalities different from the cut-set inequalities (Sect. 5) can be useful in strengthening the convex relaxations, their separation requires more computational effort.

3.2 Node Partition Inequalities

Consider a partition of the node set of the directed graph $G = (N, A)$ into $p < |N|$ disjoint subsets: $N = \cup_{t=1}^p N_t$. By shrinking these node subsets into singleton nodes, one obtains a simplified directed graph $\tilde{G} = (\tilde{N}, \tilde{A})$ with p nodes and up to $p(p-1)$ arcs. In this new graph, there is an arc $(i, j) \in \tilde{A}$ from node $i \in \tilde{N}$ to node $j \in \tilde{N}$ if and only if the original graph has at least one arc $(u, v) \in A$ from some node $u \in N_i$ to a node in $v \in N_j$. The existing capacity $\bar{c}_{ij}$ on arc $(i, j) \in \tilde{A}$ in the new network equals the sum of the existing capacities of all the arcs from the nodes in N_i to the nodes in N_j; in other words, $\tilde{c}_{ij} = \sum_{u\in N_i}\sum_{v\in N_j} \bar{c}_{uv}$.

Finally, setting the demand of $j \in \tilde{N}$ for commodity $k \in K$ to the total net demand of all nodes in N_j for commodity k in the original problem leads to a smaller network design problem with p nodes. In other words, $\tilde{w}_i^k = \sum_{v\in N_i} w_v^k$ for all $k \in K$ and $i \in \tilde{N}$. Note that one can reduce the number of commodities in the new problem by aggregating the ones with the same source node but in order to keep the notation simple, we do not discuss it here.

Now consider a feasible (integral) solution (x, y) to the original network design problem defined on $G = (N, A)$ with existing capacity vector $\bar{c}$ and commodity demands w. Aggregating the flow and capacity variables as described above, it is easy to see that the resulting flow and the capacity vector $(\tilde{x}, \tilde{y})$ gives a feasible solution to the simplified p-node problem defined on $\tilde{G} = (\tilde{N}, \tilde{A})$ with existing capacity vector $\tilde{c}$ and commodity demands $\tilde{w}$. This observation implies that valid inequalities for the simplified problem on $\tilde{G}$ can be translated to valid inequalities for the original problem on G in the following way. Let

$$\sum_{k \in K} \sum_{(i,j) \in \tilde{A}} \tilde{\alpha}_{ij}^k x_{ij}^k + \sum_{m \in M} \sum_{(i,j) \in \tilde{A}} \tilde{\beta}_{mij} y_{mij} \ \geq \ \gamma \tag{5.5}$$

be a given valid inequality for the simplified problem on $\tilde{G}$. Then the following inequality is valid for the original problem on G

$$\sum_{k \in K} \sum_{(u,v) \in A} \alpha_{uv}^k x_{uv}^k + \sum_{m \in M} \sum_{(u,v) \in A} \beta_{muv} y_{muv} \ \geq \ \gamma, \tag{5.6}$$

where for any $k \in K, m \in M$ and $(u, v) \in A$ with $u \in N_i$ and $v \in N_j$, we set

$$\alpha_{uv}^k = \begin{cases} 0, & \text{if } i = j \\ \tilde{\alpha}_{ij}, & \text{otherwise} \end{cases} \qquad \beta_{muv} = \begin{cases} 0, & \text{if } i = j \\ \tilde{\beta}_{mij}, & \text{otherwise.} \end{cases}$$

3.3 MIR Inequalities

Many valid inequalities that have found use in practice for mixed-integer optimization problems are based on the mixed-integer rounding (MIR) procedure of Nemhauser and Wolsey (1988). Wolsey (1998) illustrates the basic mixed-integer rounding on the following two variable mixed-integer set

$$Q = \Big\{(x, y) \in \mathbb{R} \times \mathbb{Z} \ : \ x + y \ \geq \ b, \ \ x \geq 0\Big\},$$

and shows that the *basic mixed-integer rounding* inequality

$$x + ry \geq r\lceil b \rceil, \tag{5.7}$$

where $r = b - \lfloor b \rfloor$, is valid and facet-defining for Q. Observe that if b is integer valued, inequality (5.7) reduces to $x \geq 0$. Otherwise, the inequality goes through feasible points $(0, \lceil b \rceil)$ and $(r, \lfloor b \rfloor)$, cutting off the fractional vertex $(0, b)$. This basic principle can be applied to more general mixed-integer sets defined by a single *base* inequality as follows. Let

$$P = \Big\{x \in \mathbb{R}^n,\ y \in \mathbb{Z}^l\ :\ ax + cy\ \geq\ b,\ \ x,\ y \geq 0\Big\}$$

where $a \in \mathbb{R}^n$ and $c \in \mathbb{R}^l$. Letting r_j denote $c_j - \lfloor c_j \rfloor$ for $j = 1, \ldots, l$, we can rewrite the base inequality as

$$\sum_{a_j<0} a_j x_j + \sum_{a_j>0} a_j x_j + \sum_{r_j<r} r_j y_j + \sum_{r_j \geq r} r_j y_j + \sum_{j=1}^{l} \lfloor c_j \rfloor y_j\ \geq\ b$$

and relax it by dropping the first term, which is non-positive, and increasing the coefficients of the fourth term, which are non-negative, to obtain the valid inequality

$$\Big(\sum_{a_j>0} a_j x_j + \sum_{r_j<r} r_j y_j\Big) + \Big(\sum_{r_j \geq r} y_j + \sum_{j=1}^{l} \lfloor c_j \rfloor y_j\Big) \geq\ b.$$

As the first two sums above are non-negative and the last two sums are integer valued, treating the first two as the nonnegative continuous variable in the set Q and the second two as the integer variable as in Q, we obtain the *MIR inequality*

$$\sum_{a_j>0} a_j x_j + \sum_{r_j<r} r_j y_j + r\Big(\sum_{r_j \geq r} y_j + \sum_{j=1}^{l} \lfloor c_j \rfloor y_j\Big)\ \geq\ r\lceil b \rceil \tag{5.8}$$

for P. This MIR inequality is generated from the base inequality $ax + cy \geq b$. Notice that, given a mixed-integer set, any valid inequality for it can be used as a base inequality to define a relaxation of the original set. Consequently, any implied inequality leads to an MIR cut. Gomory mixed-integer cuts, for example, can be seen as MIR cuts generated from base inequalities obtained from the simplex tableau.

4 Valid Inequalities from Arc Sets

In this section we review the strong valid inequalities obtained from single-arc capacity relaxations of the multicommodity network design problem. For simplicity of the presentation, we first focus on the single-facility case. We consider both the *splittable-flow arc set:*

$$F_S = \Big\{(x, y) \in [0, 1]^K \times \mathbb{Z} : \sum_{i \in K} a_i x_i \leq a_0 + y\Big\}$$

and the *unsplittable-flow arc set:*

$$F_U = \left\{(x, y) \in \{0, 1\}^K \times \mathbb{Z} : \sum_{i \in K} a_i x_i \leq a_0 + y\right\}.$$

In Sect. 4.3 we consider the multifacility generalizations of these sets.

The set F_U arises in unsplittable multicommodity problems where flow between each source-sink pair needs to be routed on a single path. In these problems, the disaggregated commodity definition is used and the set K contains all node pairs with positive demand. In the formulation, a binary flow variable x_a^k is used for each commodity-arc pair (k, a) that takes on a value of 1 if and only if the commodity is routed through the arc. Consequently, for each arc of the network this formulation has a capacity constraint of the form

$$\sum_{k \in K} d^k x_a^k \leq \bar{c}_a + c y_a, \tag{5.9}$$

where $d^k > 0$ is the demand of commodity $k \in K$, $\bar{c}_a \geq 0$ is the existing capacity, and $c > 0$ is the unit capacity to install. One arrives at F_U by dividing (5.9) by c.

Similarly, one arrives at F_S by redefining the flow variables associated with an arc and a commodity as the fraction of the total supply of that commodity traveling on that arc. In this case flow variables take values in [0, 1] and capacity constraint (5.2) takes the form (5.9). Again, dividing (5.9) by c gives the set F_S.

Without loss of generality, we assume that $a_i > 0$ for all $i \in K$, since if $a_i < 0$, x_i can be complemented and if $a_i = 0$, x_i can be dropped.

4.1 Splittable-Flow Arc Set

In this section we review the valid inequalities for the splittable flow arc set F_S. For $S \subseteq K$, by complementing the continuous variables $x_i,\ i \in S$, we can restate the arc capacity inequality as

$$\sum_{i \in S} a_i(1 - x_i) - \sum_{i \in K \setminus S} a_i x_i + y \geq a(S) - a_0, \tag{5.10}$$

where $a(S)$ stands for $\sum_{i \in S} a_i$. Relaxing the inequality by dropping $x_i,\ i \in K \setminus S$ and applying the MIR inequality, we obtain the *residual capacity inequality*

$$\sum_{i \in S} a_i(1 - x_i) \geq r(\eta - y), \tag{5.11}$$

where $\eta = \lceil a(S) - a_0 \rceil$ and $r = a(S) - a_0 - \lfloor a(S) - a_0 \rfloor$. The residual capacity inequalities together with the inequality $\sum_{i \in K} a_i x_i \le a_0 + y$ and variable bounds are sufficient to describe $conv(F_S)$.

Example 1 Consider the splittable arc set

$$F_S = \left\{ (x, y) \in [0, 1]^5 \times \mathbb{Z} : \frac{1}{3}x_1 + \frac{2}{3}x_2 + \frac{2}{3}x_3 \le y \right\}.$$

The non-dominated arc residual capacity inequalities for F_S with $r > 0$ are

S	r	Inequalities
$\{1\}$	$1/3$	$x_1 \le y$
$\{2\}$	$2/3$	$x_2 \le y$
$\{3\}$	$2/3$	$x_3 \le y$
$\{2, 3\}$	$1/3$	$2x_2 + 2x_3 \le 2 + y$
$\{1, 2, 3\}$	$2/3$	$x_1 + 2x_2 + 2x_3 \le 1 + 2y$

Given a fractional point $(\bar{x}, \bar{y})$, a violated residual capacity inequality, if it exists, can be found by the following simple separation procedure: Let $T = \{i \in K : \bar{x}_i > \bar{y} - \lfloor \bar{y} \rfloor\}$. If $a_0 + \lfloor \bar{y} \rfloor < a(T) < a_0 + \lceil \bar{y} \rceil$ and $\sum_{i \in T} a_i(1 - \bar{x}_i - \lceil \bar{y} \rceil + \bar{y}) + (\lceil \bar{y} \rceil - \bar{y})(a_0 + \lfloor \bar{y} \rfloor) < 0$, then the inequality $\sum_{i \in T} a_i(1 - x_i) \ge r(\eta - y)$ is violated by $(\bar{x}, \bar{y})$. Otherwise, there exists no residual capacity inequality violated by $(\bar{x}, \bar{y})$. Clearly, this procedure can be performed in linear time.

4.2 Unsplittable-Flow Arc Set

In this section we review the valid inequalities for the unsplittable flow arc set F_U. First, consider the related set

$$F_{Ur} = \left\{ (x, y) \in \{0, 1\}^K \times \mathbb{Z} : \sum_{i \in K} r_i x_i \le r_0 + y \right\},$$

where $r_i = a_i - \lfloor a_i \rfloor$, $i \in K \cup \{0\}$. There is a one-to-one relationship between the facets of $\text{conv}(F_U)$ and $\text{conv}(F_{Ur})$. In particular, inequality $\sum_{i \in K} \pi_i x_i \le \pi_0 + y$ defines a facet for $\text{conv}(F_U)$ if and only if inequality $\sum_{i \in K}(\pi_i - \lfloor a_i \rfloor)x_i \le \pi_0 - \lfloor a_0 \rfloor + y$ defines a facet for $\text{conv}(F_{Ur})$. Therefore, we may assume, without loss of generality, that $0 < a_i < 1$ for all $i \in K$ and $0 < a_0 < 1$.

4.2.1 *c*-strong inequalities

For $S \subseteq K$ consider the arc capacity inequality written as (5.10). Relaxing the inequality by dropping x_i, $i \in K \setminus S$ and applying integer rounding, we obtain the so-called *c-strong inequality*

$$\sum_{i \in S} x_i \le c_S + y, \tag{5.12}$$

where $c_S = |S| - \lceil a(S) - a_0 \rceil$. The set S is said to be *maximal c-strong* if $c_{S \setminus \{i\}} = c_S$ for all $i \in S$ and $c_{S \cup \{i\}} = c_S + 1$ for all $i \in K \setminus S$. Inequality (5.12) is facet-defining for $conv(F_U)$ if and only if S is maximal c-strong.

Given a point $(\bar{x}, \bar{y})$, there is a c-strong inequality violated by $(\bar{x}, \bar{y})$ if and only if there exists a set $S \subseteq K$ such that $\sum_{i \in S} \bar{x}_i - c_S > \bar{y}$. Then, a c-strong inequality is violated if and only if $\max_{S \subseteq K} \left\{ \sum_{i \in S} \bar{x}_i - \lfloor a_0 + \sum_{i \in S} (1 - a_i) \rfloor \right\} = \max \left\{ \sum_{i \in K} \bar{x}_i z_i - w \ : \ \sum_{i \in K} (1 - a_i) z_i + a_0 + 1/\lambda \le w, \ z \in \{0, 1\}^K, \ w \in \mathbb{Z} \right\} + 1 > \bar{y}$, where λ is the least common multiple of the denominators of the rational numbers $(1 - a_i)$ and a_0. The last maximization problem with the constant term $-a_0 - 1/\lambda$ is $\mathcal{NP}$-hard. Nevertheless, the separation problem has an optimal solution (z^*, w^*) such that $z_i^* = 1$ if $\bar{x}_i = 1$ and $z_i^* = 0$ if $\bar{x}_i = 0$. Therefore, we can fix such variables to their optimal values and solve the separation problem over $i \in K$ such that $0 < \bar{x}_i < 1$, which in practice can be done very efficiently even by enumeration, as most variables take on values either 0 or 1 in the LP relaxations of network design problems.

4.2.2 *k*-split *c*-strong Inequalities

The c-strong inequalities can be generalized by considering a relaxation of the capacity constraints, where the integer capacity variables are allowed to take values that are integer multiples of $1/k$ for a positive integer k. Thus the *k-split* relaxation takes the form

$$F_U^k = \left\{ (x, y) \in \{0, 1\}^K \times \mathbb{Z} : \sum_{i \in K} a_i x_i \le a_0 + z/k \right\}.$$

Letting $c_S^k = \sum_{i \in S} \lceil k a_i \rceil - \lceil k a(S) - k a_0 \rceil$, we define the *k-split c-strong inequality* as

$$\sum_{i \in S} \lceil k a_i \rceil x_i + \sum_{i \in K \setminus S} \lfloor k a_i \rfloor x_i \le c_S^k + k y. \tag{5.13}$$

The k-split c-strong inequality (5.13) is facet-defining for $conv(F_U)$ if (1) S is maximal c-strong in the k-split relaxation, (2) $f_S > (k-1)/k$ and $a_0 \ge 0$, (3) $a_i > f_S$ for all $i \in S$, $a_i < 1 - f_S$ for all $i \in K \setminus S$, where $f_S = a(S) - a_0 - \lfloor a(S) - a_0 \rfloor$.

Example 2 Consider the unsplittable arc set

$$F_U = \left\{ (x, y) \in \{0, 1\}^5 \times \mathbb{Z} : \frac{1}{3}x_1 + \frac{1}{3}x_2 + \frac{1}{3}x_3 + \frac{1}{2}x_4 + \frac{2}{3}x_5 \le y \right\}.$$

The maximal c-strong inequalities for F_U are:

$$\begin{aligned}
c_S = 0 &: x_1 \le y, \quad x_2 \le y, \quad x_3 \le y, \quad x_4 \le y \\
c_S = 1 &: x_1 + x_2 + x_4 \le 1 + y, \quad x_1 + x_2 + x_5 \le 1 + y, \quad x_2 + x_3 + x_4 \le 1 + y, \\
&\quad x_2 + x_3 + x_5 \le 1 + y, \quad x_1 + x_3 + x_4 \le 1 + y, \quad x_1 + x_3 + x_5 \le 1 + y \\
c_S = 2 &: x_1 + x_2 + x_3 + x_4 + x_5 \le 2 + y
\end{aligned}$$

As the inequalities are maximal, they are facet-defining for $conv(F_U)$. The 2-split c-strong inequality $x_2 + x_3 + x_4 + x_5 \le 2y$ and the 3-split c-strong inequality $x_1 + x_2 + x_3 + 2x_4 + 2x_5 \le 3y$ are also facet-defining for $conv(F_U)$.

4.2.3 Lifted Knapsack Cover Inequalities

Facets different from c-strong and k-split c-strong inequalities can be obtained by lifting cover inequalities from knapsack restrictions of F_U. Let K_0 and K_1 be two disjoint subsets of K and ν be a nonnegative integer. Consider the 0-1 knapsack set $F_U(\nu, K_0, K_1)$ obtained by restricting the capacity variable y to ν, all binary variables indexed with K_0 to 0 and all binary variables indexed with K_1 to 1, i.e., $F_U(\nu, K_0, K_1) \equiv \{(x, y) \in F_U : y = \nu,\ x_i = 0 \text{ for all } i \in K_0 \text{ and } x_i = 1 \text{ for all } i \in K_1\}$. For this knapsack restriction $C = K \setminus (K_0 \cup K_1)$ is called a *cover* if $r = a(C) + a(K_1) - a_0 - \nu > 0$. C is said to be a *minimal cover* if $a_i \ge r$ for all $i \in C$.

The cover inequality $\sum_{i \in C} x_i \le |C| - 1$ is facet-defining for $conv(F_U(\nu, K_0, K_1))$ if and only if C is a minimal cover. One practical way of lifting inequalities is sequential lifting, in which restricted variables are introduced to an inequality one at a time in some sequence. A lifted cover inequality

$$\sum_{i \in C} x_i + \sum_{i \in K_0} \alpha_i x_i + \sum_{i \in K_1} \alpha_i (1 - x_i) + \alpha(\nu - y) \le |C| - 1 \tag{5.14}$$

can be constructed in $O(|K|^3)$ if the capacity variable y is lifted first and such inequalities subsume all c-strong inequalities.

Example 3 For F_U given in Example 2 we list below the lifted cover inequalities of F_U that are not c-strong inequalities.

ν	$(C,\ K_0,\ K_1)$	Inequalities
1	$(\{2,3,4\},\{1,5\},\emptyset)$	$x_2+x_3+x_4+x_5 \le 2y$
1	$(\{1,4,5\},\{2,3\},\emptyset)$	$x_1+x_2+x_4+x_5 \le 2y$ and $x_1+x_3+x_4+x_5 \le 2y$
2	$(\{1,2,3,4\},\emptyset,\{5\})$	$x_1+x_2+x_3+x_4+2x_5 \le 2y+1$
2	$(\{1,2,3,5\},\emptyset,\{4\})$	$x_1+x_2+x_3+2x_4+x_5 \le 2y+1$

Computational results suggest that c-strong inequalities are quite effective in solving unsplittable multicommodity network design problems. Moreover, while the k-split c-strong and the lifted knapsack cover inequalities provide additional strengthening of the relaxations, the marginal impact on top of the basic c-strong inequalities is limited. The latter result may be due to the lack of efficient separation procedures for these inequalities.

4.3 Multifacility Arc Set

In this section we consider the multifacility extension of the arc sets discussed in the previous sections. Dash et al. (2016) study a continuous knapsack set with two integer capacity variables:

$$F_{S2} = \left\{(x,y) \in [0,1]^K \times \mathbb{Z}_+^2 : \sum_{i \in K} a_i x_i \le a_0 + c_1 y_1 + c_2 y_2\right\}.$$

They show that all non-trivial facet-defining inequalities of conv(F_{S2}) are of the form

$$\sum_{i \in S} a_i x_i + \gamma_1 y_1 + \gamma_2 y_2 \ge \beta,$$

where $S \subseteq K$ and $w + \gamma_1 y_1 + \gamma_2 y_2 \ge \beta$ is facet-defining for the set

$$Q(a,b) = \text{conv}\big\{(w,y) \in \mathbb{R} \times \mathbb{Z}_+^2 : w + c_1 y_1 + c_2 y_2 \ge b,\ a \ge w\big\}.$$

It turns out that all facets of $Q(a,b)$ can be enumerated in polynomial time. Therefore, for each $S \subseteq K$ non-trivial facet-defining inequalities for F_{SM} can be obtained from relaxations of the form $Q(a,b)$.

For the general case with many facility types

$$F_{SM} = \left\{(x,y) \in [0,1]^K \times \mathbb{Z}_+^M : \sum_{i \in K} a_i x_i \le a_0 + \sum_{m \in M} c_m y_m\right\}$$

a similar approach of complementing the flow variables for a subset $S \subseteq K$, scaling the base inequality by $c_s,\ s \in M$, and applying mixed-integer rounding gives for

F_{SM} the *multifacility residual capacity inequalities*

$$\sum_{m \in M} \phi_s(c_m) y_m + \sum_{i \in S} a_i (1 - x_i) \geq r_s \eta_s,$$

where $r_s = a(S) - a_0 - \lfloor (a(S) - a_0)/c_s \rfloor c_s$, $\eta_s = \lceil (a(S) - a_0)/c_s \rceil$, and for $k \in \mathbb{Z}$

$$\phi_s(c) = \begin{cases} c - k(c_s - r_s) & \text{if } kc_s \leq c < kc_s + r_s, \\ (k+1) r_s & \text{if } kc_s + r_s \leq c < (k+1) c_s. \end{cases}$$

5 Valid Inequalities from Cut Sets

In this section we review valid inequalities for the network design problem based on relaxations formed over cuts of the network. We first start with the single-facility case and then generalize the inequalities for multiple facilities.

5.1 *Single-Facility Case*

Consider a nonempty two-partition (U, V) of the vertices of the network. Let b^k denote the total supply of commodity k in U for V. Let A^+ be the set of arcs directed from U to V, A^- be the set of arcs directed from V to U, and $A = A^+ \cup A^-$, as shown in Fig. 5.2. As before, x_a^k denotes the flow of commodity k on arc $a \in A$ for $k \in K$. The constraints of the multicommodity network design problem across the cut are

$$x^k(A^+) - x^k(A^-) = b^k, \qquad k \in K, \tag{5.15}$$

$$\sum_{k \in K} x_a^k \leq \bar{c}_a + c y_a, \quad a \in A. \tag{5.16}$$

Then the corresponding multicommodity cut-set polyhedron is defined as

$$F_{MS} = conv\Big\{(x, y) \in \mathbb{R}_+^{A \times K} \times \mathbb{Z}_+^A : (x, y) \text{ satisfies (5.15) and (5.16)}\Big\}.$$

We refer to the single-commodity case of F_{MS} as F_{SS}.

In the following sections we describe valid inequalities for F_{MS} by considering single-commodity relaxations of F_{MS} obtained by aggregating flow variables and balance equations (5.15) over subsets of K. For $Q \subseteq K$ let $x_Q(S) = \sum_{k \in Q} x^k(S)$ and $b_Q = \sum_{k \in Q} b^k$.

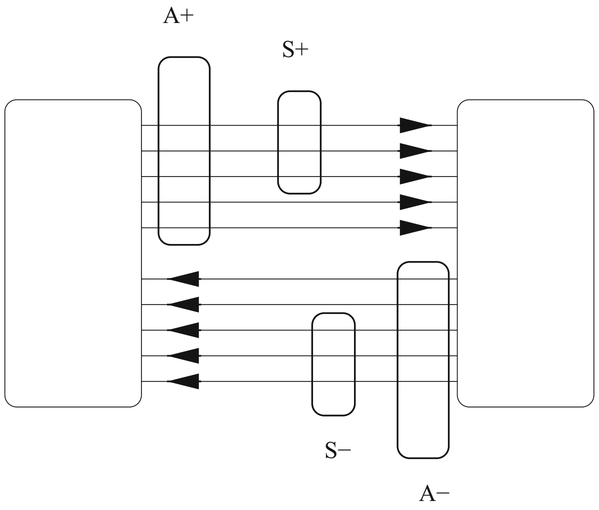


Fig. 5.2 Cut-set relaxation of the network design problem

Cut-Set Inequalities Consider the following relaxation of F_{MS} on the integer capacity variables:

$$\bar{c}(A^+)+cy(A^+) \geq x_K(A^+) \geq b_K.$$

Applying the integer rounding procedure reviewed in Sect. 3.3 to this relaxation, one obtains the so-called *cut-set inequality*

$$y(A^+) \geq \lceil (b_K - \bar{c}(A^+))/c \rceil \tag{5.17}$$

for F_{MS}, which is unique per cut-set relaxation. Finding the best cut-set relaxation is not easy however. For the single-source single-sink case, the problem of finding the best cut set can be posed as an $s - t$ max-cut problem.

Flow-Cut-Set Inequalities The basic cut-set inequalities (5.17) can be generalized by incorporating the flow variables in addition to the capacity variables (see Fig. 5.2). For $S^+ \subseteq A^+$, $S^- \subseteq A^-$ and $Q \subseteq K$ consider the following relaxation of F_{MS} :

$$\bar{c}(S^+)+cy(S^+) + x_Q(A^+ \setminus S^+) - x_Q(S^-) \geq b_Q,$$

$$0 \leq \sum_{k \in Q} x_a^k \leq \bar{c}_a + cy_a, \ \forall a \in A,$$

which is written equivalently as

$$c\big[y(S^+) - y(S^-)\big] + x_Q(A^+ \setminus S^+) + \big[\bar{c}(S^-) + cy(S^-) - x_Q(S^-)\big] \geq b'_Q,$$
$$0 \leq \sum_{k \in Q} x_a^k \leq \bar{c}_a + cy_a, \ \forall a \in A.$$

where $b'_Q = b_Q - \bar{c}(S^+) + \bar{c}(S^-)$. Letting $r_Q = b'_Q - \lfloor b'_Q/c \rfloor c$ and $\eta_Q = \lceil b'_Q/c \rceil$ and observing that $x_Q(A^+ \setminus S^+) \geq 0$, $\bar{c}(S^-) + cy(S^-) - x_Q(S^-) \geq 0$, we can apply the MIR procedure reviewed in Sect. 3.3 to this relaxation to arrive at the flow-cut set inequalities

$$r_Q y(S^+) + x_Q(A^+ \setminus S^+) + (c - r_Q)y(S^-) - x_Q(S^-) \geq r_Q \eta_Q. \qquad (5.18)$$

The flow-cut-set inequalities (5.18) along with the balance, bound, and capacity constraints are sufficient to describe the single-commodity case F_{SS}.

For a given $Q \subseteq K$, observe that flow-cut-set inequalities (5.18) is an exponential class. However, given a solution $(\bar{x}, \bar{y})$, one finds a subset S^+ with the smallest left-hand-side value as follows: if $r_Q \bar{y}_a < \sum_{k \in Q} \bar{x}_a^k$ for $a \in A^+$, then we include a in S^+; if $(c - r)\bar{y}_a < \sum_{k \in Q} \bar{x}_a^k$ for $a \in A^-$, then we include $a \in S^-$.

For a fixed cut of the network, the complexity of separating multicommodity flow cut-set inequalities (5.18) is an open question. Optimization of a linear function over F_{MS} is $\mathcal{NP}$-hard as the facility location problem is a special case of it. For a multicommodity single-facility network design problem of a single arc, cut-set inequalities (5.21) reduce to the residual capacity inequalities (5.11), for which an exact linear-time separation method is known. From here it follows that, for a single-facility problem, if S^+ and S^- are fixed, then one can find a subset of commodities $Q \subseteq K$ that gives a most violated inequality in linear time. Alternatively, if Q is fixed, since the model reduces to a single-commodity, one can find the subsets $S^+ \subseteq A^+$ and $S^- \subseteq A^-$ that give a most violated inequality in linear time as well. However, the complexity of determining Q, S^+ and S^- simultaneously is an open question.

Example 4 Consider the following single-commodity optimization problem with two outflow arcs and one inflow arc:

$$\max\ x_1 + x_2 + x_3 - y_1 - y_2 - y_3 \text{ s.t. } x_1 + x_2 - x_3 = 0.5,\ 0 \leq x_i \leq y_i \in \mathbb{Z},\ i = 1, 2, 3.$$

One of its fractional solutions is $x_1 = y_1 = 0.5$ and all other variables zero. Adding the cut-set inequality

$$y_1 + y_2 \geq 1$$

cuts off this solution, but leads to another fractional solution: $x_1 = y_1 = 1,\ x_3 = y_3 = 0.5$. Adding the flow-cut-set inequality

$$0.5x_1 + y_2 + 0.5x_3 - y_3 \geq 0.5$$

cuts it off, but gives the fractional solution: $x_2 = y_2 = 1, x_3 = y_3 = 0.5$. Adding the flow-cut-set inequality

$$y_1 + 0.5x_2 + 0.5x_3 - y_3 \geq 0.5$$

cuts it off, but this time gives the fractional solution: $x_1 = y_1 = x_2 = y_2 = x_3 = y_3 = 0.5$. Finally, adding the flow-cut-set inequality

$$0.5x_1 + 0.5x_2 + 0.5x_3 - y_3 \geq 0.5$$

leads to an optimal integer solution $x_1 = 0.5$, $y_1 = 1$ and all other variables zero.

5.2 Multifacility Case

Next we consider network design models where one is allowed to install facilities of multiple types with different capacities on the arcs of the network. Let c_m be the capacity of facility of type $m,\ m \in M$. No assumption is made on either the number of facility types or the structure of capacities (other than $c_m > 0$ and rational).

The constraints of the multicommodity multifacility problem across cut A are written as

$$x^k(A^+) - x^k(A^-) = b^k, \qquad k \in K, \tag{5.19}$$

$$\sum_{k \in K} x_a^k \leq \bar{c}_a + \sum_{m \in M} c_m y_{m,a}, \qquad a \in A. \tag{5.20}$$

So the corresponding multicommodity multifacility cut-set polyhedron is

$$F_{MM} = conv\Big\{(x, y) \in \mathbb{R}_+^{A \times K} \times \mathbb{Z}_+^{A \times M} : (x, y) \text{ satisfies (5.19) and (5.20)}\Big\}.$$

For $Q \subseteq K$ and $s \in M$ let $r_{s,Q} = b'_Q - \lfloor b'_Q / c_s \rfloor c_s$, $\eta_{s,Q} = \lceil b'_Q / c_s \rceil$. Then, the following *multicommodity multifacility cut-set inequality* is valid for F_{MM} :

$$\sum_{m \in M} \phi_{s,Q}^+(c_m) y_m(S^+) + x_Q(A^+ \setminus S^+) + \sum_{m \in M} \phi_{s,Q}^-(c_m) y_m(S^-) - x_Q(S^-) \geq r_{s,Q}\eta_{s,Q}, \tag{5.21}$$

where, for $k \in \mathbb{Z}$,

$$\phi_{s,Q}^+(c) = \begin{cases} c - k(c_s - r_{s,Q}) & \text{if } kc_s \leq c < kc_s + r_{s,Q}, \\ (k+1)r_{s,Q} & \text{if } kc_s + r_{s,Q} \leq c < (k+1)c_s, \end{cases}$$

and

$$\phi^-_{s,Q}(c) = \begin{cases} c - kr_{s,Q} & \text{if } kc_s \le c < (k+1)c_s - r_{s,Q}, \\ k(c_s - r_{s,Q}) & \text{if } kc_s - r_{s,Q} \le c < kc_s. \end{cases}$$

The above $\phi^+_{s,Q}$ and $\phi^-_{s,Q}$ are subadditive MIR functions written in closed form. The multicommodity multifacility cut-set inequality (5.21) is facet-defining for F_{MM} if $(S^+, A^+ \setminus S^+)$ and $(S^-, A^- \setminus S^-)$ are nonempty partitions, $r_{s,Q} > 0$, and $b^k > 0$ for all $k \in Q$.

For the single-commodity case F_{SM}, inequalities (5.21) reduce to

$$\sum_{m\in M} \phi^+_s(c_m) y_m(S^+) + x(A^+ \setminus S^+) + \sum_{m\in M} \phi^-_s(c_m) y_m(S^-) - x(S^-) \ge r_s \eta_s. \tag{5.22}$$

In this case, given a cut A for each facility $s \in M$, the multifacility cut-set inequalities (5.22) is an exponential class by the choice of the subsets of arcs S^+ and S^-. However, finding a subset that gives a most violated inequality for a point $(\bar{x}, \bar{y})$ is straightforward. If $\sum_{m\in M} \phi^+_s(c_m)\bar{y}_{m,a} < \bar{x}_a$ for $a \in A^+$, then we include a in S^+, and if $\sum_{m\in M} \phi^-_s(c_m)\bar{y}_{m,a} < \bar{x}_a$ for $a \in A^-$, then we include a in S^-. Since $\phi^+_s(c)$ or $\phi^-_s(c)$ can be calculated in constant time, for a fixed cut A, a violated multifacility cut-set inequality is found in $O(|A||M|)$ if there exists any.

Example 5 We specialize inequality (5.21) for the network design problem with two types of facilities. Let the vectors y_1 and y_2 denote the facilities with capacities $c_1 = 1$ and $c_2 = \lambda > 1$ with $\lambda \in \mathbb{Z}$, respectively. Let Q be a nonempty subset of the commodities. Then by letting $s = 1$, we have $r_{1,Q} = b_Q - \lfloor b_Q \rfloor$ and inequality (5.21) becomes

$$r_{1,Q} y_1(S^+) + (r_{1,Q}\lfloor\lambda\rfloor + \min\{\lambda - \lfloor\lambda\rfloor, r_{1,Q}\}) y_2(S^+) + x_Q(A^+ \setminus S^+) + (1 - r_{1,Q})$$
$$y_1(S^-) + ((1 - r_{1,Q})\lfloor\lambda\rfloor + \min\{\lambda - \lfloor\lambda\rfloor, 1 - r_{1,Q}\}) y_2(S^-) - x_Q(S^-) \ge r_{1,Q}\lceil b_Q \rceil,$$

which, when λ is integer, reduces to

$$\begin{aligned} &r_{1,Q} y_1(S^+) + \lambda r_{1,Q} y_2(S^+) + x_Q(A^+ \setminus S^+) + \\ &\quad (1 - r_{1,Q}) y_1(S^-) + \lambda(1 - r_{1,Q}) y_2(S^-) - x_Q(S^-) \ge r_{1,Q}\lceil b_Q \rceil. \end{aligned} \tag{5.23}$$

Notice that inequality (5.23) is not valid for F_{MM} unless $\lambda \in \mathbb{Z}$. Also by letting $s = 2$, we have $r_{2,Q} = b_Q - \lfloor b_Q/\lambda \rfloor\lambda$. So the corresponding multicommodity two-facility cut-set inequality is

$$\begin{aligned} &\min\{1, r_{2,Q}\} y_1(S^+) + r_{2,Q} y_2(S^+) + x_Q(A^+ \setminus S^+) + \\ &\quad \min\{1, \lambda - r_{2,Q}\} y_1(S^-) + (\lambda - r_{2,Q}) y_2(S^-) - x_Q(S^-) \ge r_{2,Q}\lceil b_Q/\lambda \rceil. \end{aligned}$$

It should be clear that multifacility flow-cut-set inequalities are also valid for a single-facility model with varying capacities on the arcs of the network. The inequalities from cut-set relaxations have been shown to be very effective in solving network design problems in computational studies.

6 Partition Inequalities

Partition inequalities are arguably the most effective cutting planes for the network design problem. These inequalities have non-zero coefficients for only the integer capacity variables that cross a multicut obtained from a partition of the nodes of the network. They generalize the cut-set inequality (5.17) described in Sect. 5.

For ease of exposition, first consider a two-partition of the node set $N = N_1 \cup N_2$. As discussed in Sect. 3.2 shrinking node sets N_1 and N_2 leads to a network with two nodes and two edges (assuming there is at least one arc from a node in N_1 to a node in N_2, and vice versa). Then, the inequality

$$\sum_{m \in M} c_m y_{m12} \geq \sum_{k \in K} \sum_{v \in N_2} w_v^k - \sum_{u \in N_1} \sum_{v \in N_2} \bar{c}_{uv} \ = b$$

must be satisfied by all feasible solutions of the two-node problem. Following the argument in Sect. 3.2, inequality

$$\sum_{m \in M} c_m \left(\sum_{u \in N_1} \sum_{v \in N_2} y_{muv} \right) \geq b \tag{5.24}$$

is valid for the solutions to the (LP relaxation of the) original problem. Notice that inequality (5.24) is a metric inequality (5.4) generated by the vector $v \in \{0, 1\}^A$, where $v_{ij} = 1$ if and only if $i \in N_1$ and $j \in N_2$.

As we assume that all c_m are integral, which is the case in most applications, the inequality (5.24) leads to the integer knapsack cover set

$$X = \left\{ z \in \mathbb{Z}^M : \sum_{m \in M} c_m z_m \geq b, \ z \geq 0 \right\},$$

where the variable z_m stands for the sum $\sum_{u \in N_1} \sum_{v \in N_2} y_{muv}$. Consequently, any valid inequality $\sum_{m \in M} \alpha_m z_m \geq \beta$ for X yields a valid inequality for P^{ND} after replacing each variable z_m with the corresponding sum of the original variables.

The polyhedral structure of the set X when c_{m+1} is an integer multiple of c_m for all $m = 1, \dots, |M| - 1$ has been studied by Pochet and Wolsey (1995) who derive what they call "partition" inequalities and show that these inequalities together with the nonnegativity constraints describe conv(X). They also derive conditions under which partition inequalities are valid in the general case when the divisibility condition does not hold.

The partition inequalities described in Pochet and Wolsey (1995) are obtained by applying the MIR procedure iteratively. More precisely, the first step is to choose a subset $\{j_1, j_2, \ldots, j_r\}$ of the index set M, where $j_i < j_{i+1}$, and therefore, $c_{j_i} < c_{j_{i+1}}$ for all $i = 1, \ldots, r-1$. The inequality $\sum_{m \in M} c_m z_m \geq b$ is then divided by c_{j_r} and the MIR cut based on this inequality is written. The resulting MIR inequality is then divided by $c_{j_{r-1}}$ and the MIR procedure is applied again. This process is repeated with all c_{j_i} for $i = 1, \ldots, r$ to obtain the final inequality. Note that the sequential application of the MIR procedure yields valid inequalities even when the divisibility condition does not hold. However, in this case, they are not sufficient to define $\text{conv}(X)$.

Now consider a three-partition of the node set $N = N_1 \cup N_2 \cup N_3$. Following the discussion on two-partitions, consider the single capacity network design problem with $\tilde{G} = (\tilde{N}, \tilde{A})$ where $\tilde{N} = \{1, 2, 3\}$ and $\tilde{A} = \{a_{12}, a_{13}, a_{21}, a_{23}, a_{31}, a_{32}\}$. Furthermore, let $\bar{c}_a$ and $\bar{t}_a$ respectively denote the existing capacity and traffic demands for $a \in \tilde{A}$. Clearly, any valid inequality for the simplified problem on $\tilde{G}$ can be transformed into a valid inequality for the original problem defined on G.

For the three-node problem, there are three possible two-partitions and for each partition, one can write two possible cut-set inequalities by treating one of the two sets as N_1 and the other as N_2. Consequently, one can write six different two-partition inequalities where each capacity variable appears in exactly two inequalities. Summing all six inequalities leads to

$$\sum_{a \in \tilde{A}} \sum_{m \in M} 2 c_m y_{ma} \geq \sum_{i \in \tilde{N}} \lceil s_i \rceil + \sum_{i \in \tilde{N}} \lceil t_i \rceil, \tag{5.25}$$

where s_i denotes the difference between the traffic leaving node i and the existing capacity on the outgoing arcs. Similarly t_i is the difference between the traffic entering node i and the existing capacity on the incoming arcs. If the right-hand-side of inequality (5.25) is an odd number, dividing the inequality by two and rounding up the right-hand-side yields the following inequality

$$\sum_{m \in M} c_m \sum_{a \in \tilde{A}} y_{ma} \geq \left\lceil \frac{\sum_{i \in \tilde{N}} \lceil s_i \rceil + \sum_{i \in \tilde{N}} \lceil t_i \rceil)}{2} \right\rceil. \tag{5.26}$$

Next we will generate a similar inequality based on metric inequalities. Let $a, b \in \tilde{N}$ be two distinct nodes and define $v^{ab} \in \{0, 1\}^{\tilde{A}}$ to be the vector with components $v_{ab} = v_{ac} = v_{bc} = 1$ and $v_{ba} = v_{ca} = v_{cb} = 0$. Now consider the metric inequality (5.4) generated by v^{ab}:

$$\sum_{m \in M} c_m (y_{mab} + y_{mac} + y_{mbc}) \geq \bar{t}_{ab} + \bar{t}_{ac} + \bar{t}_{bb} - \bar{c}_{ab} - \bar{c}_{ac} - \bar{c}_{bc}.$$

Once again, if fractional, the right-hand-side of this inequality can be rounded up. In addition, more valid inequalities can be generated using the MIR procedure iteratively.

Furthermore, let $c \in \tilde{N}$ be the node different from a and b and note that adding up the metric inequalities generated by v^{ab} and v^{cb}, one obtains

$$\sum_{m \in M} c_m \sum_{a \in \tilde{A}} y_{ma} \geq \lceil d_{ab} \rceil + \lceil d_{cb} \rceil \tag{5.27}$$

where d_{ij} denotes the right-hand-side of the metric inequality generated by v^{ij} for $(i, j) \in A$. Moreover, adding up the metric inequalities generated by v^{ba} and v^{ca} gives a similar inequality with right-hand-side of $\lceil d_{ba} \rceil + \lceil d_{ca} \rceil$. Similarly, v^{ac} and v^{bc} yields an inequality with right-hand-side of $\lceil d_{ac} \rceil + \lceil d_{bc} \rceil$. Adding up two of these inequalities with the larger right-hand-side and dividing the resulting inequality by two leads to a valid inequality of the form (5.26). More precisely, if both $\lceil d_{ba} \rceil + \lceil d_{ca} \rceil$ and $\lceil d_{ab} \rceil + \lceil d_{cb} \rceil$ are larger than $\lceil d_{ac} \rceil + \lceil d_{bc} \rceil$, then the resulting inequality is

$$\sum_{m \in M} c_m \sum_{a \in \tilde{A}} y_{ma} \geq \left\lceil \frac{\lceil d_{ba} \rceil + \lceil d_{ca} \rceil + \lceil d_{ab} \rceil + \lceil d_{cb} \rceil}{2} \right\rceil . \tag{5.28}$$

In addition to inequalities (5.26) and (5.28), it is possible to write similar *total capacity inequalities* by combining some cut-set inequalities with metric inequalities in such a way that the left-hand-side of the inequality has all the capacity variables with a coefficient of two. As all these inequalities have the same left-hand-side, only the one with the largest right-hand-side should be used. For example, if $\bar{t}_{ij} = 1/2$ and $\bar{c}_{ij} = 0$ for $(i, j) \in \tilde{A}$, then the right-hand-side of (5.26) is 3, whereas the right-hand-side of (5.28) is 4 and therefore inequality (5.28) is stronger than (5.26). However, if $\bar{t}_{ij} = 1/3$ and $\bar{c}_{ij} = 0$ for $(i, j) \in \tilde{A}$, then the right-hand-side of (5.26) is still 3, whereas the right-hand-side of (5.28) becomes 2.

Furthermore, as total capacity inequalities have the same form as inequality (5.24), one can define the corresponding integer knapsack cover set from the stronger one and derive further valid inequalities using the MIR procedure iteratively.

Hamid and Agarwal (2015) study the undirected variant of the two-facility network design problem, where the total flow on an arc plus the flow on reverse arc is limited by the capacity of the undirected edge associated with the two arcs. In this case, the authors computationally enumerate the complete list of facets that can be obtained from a given three-partition. Also see Agarwal (2006) for a study of four-partition facets for the undirected variant of the single-facility network design problem. Their computational study suggests that using larger partitions of the node set improves the relaxation but with diminishing returns. More precisely, they observe that two, three and four-partition cuts reduce the optimality gap of the LP relaxation to 12.5%, 6.3%, and 2.6%, respectively.

7 Bibliographical Notes

7.1 Introduction

Capacity expansion problems have been studied in the context of telecommunication (Magnanti and Wong 1984; Minoux 1989; Balakrishnan et al. 1991, 1995) and train scheduling with multiple locomotive options (Florian et al. 1976). The unsplittable flow version of the network design problem appears in telecommunication networks using multiprotocol label switching (MPLS) technique, production and distribution with single sourcing, express package delivery, and train scheduling (e.g., Gavish and Altinkemer 1990; Barnhart et al. 2000; Davarnia et al. 2019). Özbaygin et al. (2018) utilize the splittable flow model to solve the split delivery vehicle routing problem. Atamtürk and Günlük (2018) study the connection between valid inequalities for directed, bidirected and undirected network design problems.

7.2 Problem Formulation and Preliminaries

Günlük (2007) describes how to obtain the smallest number of commodities necessary to formulate a given multicommodity flow problem. Metric inequalities and their extensions have been used for various network design problems by several authors, including Dahl and Stoer (1998), Mirchandani (2000), Labbé and Yaman (2004), Avella et al. (2007), Costa et al. (2009), Bienstock et al. (1998). Mattia (2012) presents computations that illustrate the value of utilizing metric inequalities through a bi-level programming separation procedure. Hamid and Agarwal (2015) show that if inequality (5.5) is facet-defining for the network design problem on $\tilde{G}$ with $\tilde{c}$ and $\tilde{w}$, then inequality (5.6) is facet-defining for P^{ND} provided that $\tilde{\alpha} = 0,\ \gamma > 0$, and node sets $N_1, \ldots, N_p$ induce connected components of G. In addition, Raack et al. (2011) show that the same result holds without the assumption $\tilde{\alpha} = 0$ when $p = 2$ and $|M| = 1$.

7.3 Valid Inequalities from Arc Sets

The arc sets and their generalizations are studied by Magnanti et al. (1993), Atamtürk and Rajan (2002), van Hoesel et al. (2002), Brockmüller et al. (2004), Atamtürk and Günlük (2007), Yaman (2013). Magnanti et al. (1993) introduced the *residual capacity inequality*. Atamtürk and Rajan (2002) gave the polynomial-time separation procedure for residual capacity inequalities. Brockmüller et al. (2004) introduced the *c-strong inequality* (5.12) for F_U. The k-split c-strong inequalities (Sect. 4.2.2) are given by Atamtürk and Rajan (2002). Agra and Constantino (2006) show that all facets of $Q(a, b)$ can be enumerated in polynomial time.

7.4 Valid Inequalities from Cut Sets

A recent review on cut-based inequalities for network design can be found in Raack et al. (2011). Magnanti and Mirchandani (1993) introduce the integer cut-set inequalities for F_{SS}. For the single-source single-sink case, given a solution, Barahona (1996) formulated the problem of finding the best cut set as an $s - t$ max-cut problem. Bienstock and Günlük (1996), Chopra et al. (1998) gave the mixed-integer flow-cut-set generalization (5.18) of the basic cut-set inequalities. Atamtürk (2002) showed that the flow-cut-set inequalities (5.18) along with the balance, bound, and capacity constraints are sufficient to describe the single-commodity case F_{SS}. Magnanti and Mirchandani (1993), Magnanti et al. (1995), Pochet and Wolsey (1995), Bienstock and Günlük (1996), Günlük (1999), Wolsey and Yaman (2016) give valid inequalities for the network design problem with multiple capacities when capacities are divisible. Atamtürk et al. (2001) consider a binary capacity version with no assumption on divisibility. multicommodity multifacility network design problems are considered in Bienstock et al. (1998), Atamtürk (2002).

7.5 Partition Inequalities

The polyhedral structure of the set X without the divisibility assumption on the capacities has been studied by Atamtürk (2003) and Yaman (2007), also see the references therein. Atamtürk et al. (2016) gave three-partition flow cover inequalities for the fixed-charge network design problem. The total capacity inequalities were proposed by Bienstock et al. (1998). Hamid and Agarwal (2015) has enumerated the complete list of facets that can be obtained from a given three-partition for the undirected variant of the two-facility network design problem. Agarwal (2006) has studied the four-partition facets and computationally showed that using larger partitions of the node set improves the LP relaxation but with diminishing returns.

8 Conclusions and Perspectives

In this chapter, we reviewed strong valid inequalities for the multicommodity, multifacility network design problem. Metric inequalities for projecting out continuous flow variables, mixed-integer rounding from appropriate base relaxations, and shrinking the network to a small k-node graph have been the main tools for deriving the inequalities introduced in the literature. Going forward, we expect more recent techniques such as multistep mixed-integer rounding (Dash and Günlük 2006), mingling (Atamtürk and Günlük 2010), and multistep mingling (Atamtürk and Kianfar 2012) that generalize and extend mixed-integer rounding to be useful for deriving new inequalities for this class of network design problems with continuous as well as general integer variables.

We finish this section with comments on computational aspects and challenges in implementing branch-and-cut algorithms. One of the prerequisites of applying the valid inequalities described in this chapter as cutting planes is to automatically recognize the appropriate network structure and identify flow-balance constraints and the relevant capacity constraints as part of a hidden multicommodity, multifacility network structure. Achterberg and Raack (2010) describe algorithms to automatically detect such network structures and generate inequalities from cut-set relaxations. They utilize c-MIR inequalities (Marchand and Wolsey 2001), which are obtained by first complementing bounded variables and then applying the MIR procedure. Given a fractional solution, it is nontrivial to decide which aggregations to apply to flow variables, which variables to complement. Nevertheless, the automatic detection and separation procedure in this paper results in 18% time reduction for a large set of publicly available test problems.

Acknowledgments A. Atamtürk is supported, in part, by grant FA9550-10-1-0168 from the Office of the Assistant Secretary of Defense for Research and Engineering. O. Günlük was hosted by the Numerical Analysis Group at Oxford Mathematical Institute and by the Alan Turing Institute during this project and would like to thank the members of both groups for their hospitality.

References

Achterberg, T., & Raack, C. (2010). The MCF-separator: Detecting and exploiting multi-commodity flow structures in MIPs. *Mathematical Programming Computation, 2*, 125–165.

Agarwal, Y. K. (2006). K-partition-based facets of the network design problem. *Networks, 47*, 123–139.

Agra, A., & Constantino, M. (2006). Description of 2-integer continuous knapsack polyhedra. *Discrete Optimization, 3*, 95–110.

Atamtürk, A. (2002). On capacitated network design cut–set polyhedra. *Mathematical Programming, 92*, 425–437.

Atamtürk, A. (2003). On the facets of mixed–integer knapsack polyhedron. *Mathematical Programming, 98*, 145–175.

Atamtürk, A., & Günlük, O. (2007). Network design arc set with variable upper bounds. *Networks, 50*, 17–28.

Atamtürk, A., & Günlük, O. (2010). Mingling: Mixed-integer rounding with bounds. *Mathematical Programming, 123*, 315–338.

Atamtürk, A., & Günlük, O. (2018). A note on capacity models for network design. *Operations Research Letters, 46*, 414–417.

Atamtürk, A., & Kianfar, K. (2012). n-step mingling inequalities: New facets for the mixed-integer knapsack set. *Mathematical Programming, 132*, 79–98.

Atamtürk, A., & Rajan, D. (2002). On splittable and unsplittable flow capacitated network design arc-set polyhedra. *Mathematical Programming, 92*, 315–333.

Atamtürk, A., Nemhauser, G. L., & Savelsbergh, M. W. P. (2001). Valid inequalities for problems with additive variable upper bounds. *Mathematical Programming, 91*, 145–162.

Atamtürk, A., Gómez, A., & Küçükyavuz, S. (2016). Three-partition flow cover inequalities for constant capacity fixed-charge network flow problems. *Networks, 67*, 299–315.

Avella, P., Mattia, S., & Sassano, A. (2007). Metric inequalities and the network loading problem. *Discrete Optimization, 4*, 103–114.

Balakrishnan, A., Magnanti, T. L., Shulman, A., & Wong, R. T. (1991). Models for planning capacity expansion in local access telecommunication networks. *Annals of Operations Research, 33*, 239–284.

Balakrishnan, A., Magnanti, T. L., & Wong, R. T. (1995). A decomposition algorithm for local access telecommunication network expansion planning. *Operations Research, 43*, 58–76.

Barahona, F. (1996). Network design using cut inequalities. *SIAM Journal on Optimization, 6*, 823–837.

Barnhart, C., Hane, C. A., & Vance, P. H. (2000). Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Operations Research, 48*, 318–326.

Bienstock, D., & Günlük, O. (1996). Capacitated network design - Polyhedral structure and computation. *INFORMS Journal on Computing, 8*, 243–259.

Bienstock, D., Chopra, S., Günlük, O., & Tsai, C. Y. (1998). Minimum cost capacity installation for multicommodity networks. *Mathematical Programming, 81*, 177–199.

Brockmüller, B., Günlük, O., & Wolsey, L. A. (2004). Designing private line networks – Polyhedral analysis and computation. *Transactions on Operational Research, 16*, 7–24.

Chopra, S., Gilboa, I., & Sastry, S. T. (1998). Source sink flows with capacity installation in batches. *Discrete Applied Mathematics, 85*, 165–192.

Costa, A. M., Cordeau, J. F., & Gendron, B. (2009). Benders, metric and cutset inequalities for multicommodity capacitated network design. *Computational Optimization and Applications, 42*, 371–392.

Dahl, G., & Stoer, M. (1998). A cutting plane algorithm for multicommodity survivable network design problems. *INFORMS Journal on Computing, 10*, 1–11.

Dash, S., & Günlük, O. (2006). Valid inequalities based on simple mixed-integer sets. *Mathematical Programming, 105*, 29–53.

Dash, S., Günlük, O., & Wolsey, L. A. (2016). The continuous knapsack set. *Mathematical Programming, 155*, 471–496.

Davarnia, D., Richard, J. P. P., Içyüz, Ay. E., & Taslimi, B. (2019). Network models with unsplittable node flows with application to unit train scheduling. *Operations Research, 67*, 1053–1068.

Florian, M., Bushell, G., Ferland, J., Guérin, G., & Nastansky, L. (1976). The engine scheduling problems in a railway network. *INFOR: Information Systems and Operational Research 14*, 121–138.

Gavish, B., & Altinkemer, K. (1990). Backbone network design tools with economic tradeoffs. *ORSA Journal on Computing, 2*, 58–76.

Günlük, O. (1999). A branch-and-cut algorithm for capacitated network design problems. *Mathematical Programming, 86*, 17–39.

Günlük, O. (2007). A new min-cut max-flow ratio for multicommodity flows. *SIAM Journal on Discrete Mathematics, 21*, 1–15.

Hamid, F., & Agarwal, Y. K. (2015). Solving the two-facility network design problem with 3-partition facets. *Networks, 66*, 11–32.

Iri, M. (1971). On an extension of the max-flow min-cut theorem to multicommodity flows. *Journal of the Operations Research Society of Japan, 13*, 129–135.

Labbé, M., & Yaman, H. (2004). Projecting the flow variables for hub location problems. *Networks, 44*, 84–93.

Magnanti, T. L., & Mirchandani, P. (1993). Shortest paths, single origin-destination network design, and associated polyhedra. *Networks, 23*, 103–121.

Magnanti, T. L., & Wong, R. T. (1984). Network design and transportation planning: Models and algorithms. *Transportation Science, 18*, 1–55.

Magnanti, T. L., Mirchandani, P., & Vachani, R. (1993). The convex hull of two core capacitated network design problems. *Mathematical Programming, 60*, 233–250.

Magnanti, T. L., Mirchandani, P., & Vachani, R. (1995). Modeling and solving the two–facility capacitated network loading problem. *Operations Research, 43*, 142–157.

Marchand, H., & Wolsey, L. A. (2001). Aggregation and mixed integer rounding to solve MIPs. *Operations Research, 49*, 363–371.

Mattia, S. (2012). Separating tight metric inequalities by bilevel programming. *Operations Research Letters, 40*, 568–572.

Minoux, M. (1989). Network synthesis and optimum network design problems: Models, solution methods and applications. *Networks, 19*, 313–360.

Mirchandani, P. (2000). Projections of the capacitated network loading problem. *European Journal of Operational Research, 122*, 534–560.

Nemhauser, G. L., & Wolsey, L. A. (1988). *Integer and combinatorial optimization*. New York: Wiley.

Onaga, K., & Kakusho, O. (1971). On feasibility conditions of multi-commodity flows in networks. *Transactions on Circuit Theory, 18*, 425–429.

Özbaygin, G., Karasan, O., & Yaman, H. (2018). New exact solution approaches for the split delivery vehicle routing problem. *EURO Journal on Computational Optimization, 6*, 85–115.

Pochet, Y., & Wolsey, L. A. (1995). Integer knapsack and flow covers with divisible coefficients: Polyhedra, optimization, and separation. *Discrete Applied Mathematics, 59*, 57–74.

Raack, C., Koster, A. M., Orlowski, S., & Wessäly, R. (2011). On cut-based inequalities for capacitated network design polyhedra. *Networks, 57*, 141–156.

van Hoesel, S. P. M., Koster, A. M. C. A., van de Leensel, R. L. M. J., & Savelsbergh, M. W. P. (2002). Polyhedral results for the edge capacity polytope. *Mathematical Programming, 92*, 335–358.

Wolsey, L. A. (1998). *Integer programming*. New York: Wiley.

Wolsey, L. A., & Yaman, H. (2016). Continuous knapsack sets with divisible capacities. *Mathematical Programming, 156*, 1–20.

Yaman, H. (2007). The integer knapsack cover polyhedron. *SIAM Journal on Discrete Mathematics, 21*, 551–572.

Yaman, H. (2013). The splittable flow arc set with capacity and minimum load constraints. *Operations Research Letters, 41*, 556–558.

Chapter 6
Piecewise Linear Cost Network Design

Antonio Frangioni and Bernard Gendron

1 Introduction

In this chapter, following the trend initiated in Chap. 5, we move decidedly away from “basic” network design models to incorporate more and more sophisticated elements required to accurately model real-world applications. Perhaps the first and most important of these is the fact that the capacity on the arcs does not come in an “all-or-nothing” fashion, but with a more complex cost. This often corresponds to the fact that providing capacity actually boils down to provisioning appropriate *facilities* on the arc (say, dedicating cars/trucks to a leg of a transportation network, or installing stretches of optic fibre or transceivers in a telecommunication network). In some simple cases, such as when the facilities can be installed independently from one another or when they are all identical, the models of the previous chapters can be adapted via the simple trick of *replicating the arcs*: different copies of the same arc are present and can be independently constructed. This leads to replicating the flow variables, which might be disadvantageous, although, as we shall see, some of the best models are doing something similar anyway. Furthermore, in general, the shape of the cost-of-capacity function can be much more complex due to nontrivial interactions between installation of different facilities. For instance, in truckload (TL) transportation, each truck being used would incur a fixed cost and the piecewise linear cost would have a simple staircase shape where each segment corresponds to the number of trucks being used. In less-than-truckload (LTL) transportation,

A. Frangioni (✉)
Dipartimento di informatica, Università di Pisa, Italy
e-mail: frangio@di.unipi

B. Gendron
CIRRELT and Département d’informatique et de recherche opérationnelle,
Université de Montréal, Montréal, QC, Canada
e-mail: Bernard.Gendron@cirrelt.net

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_6

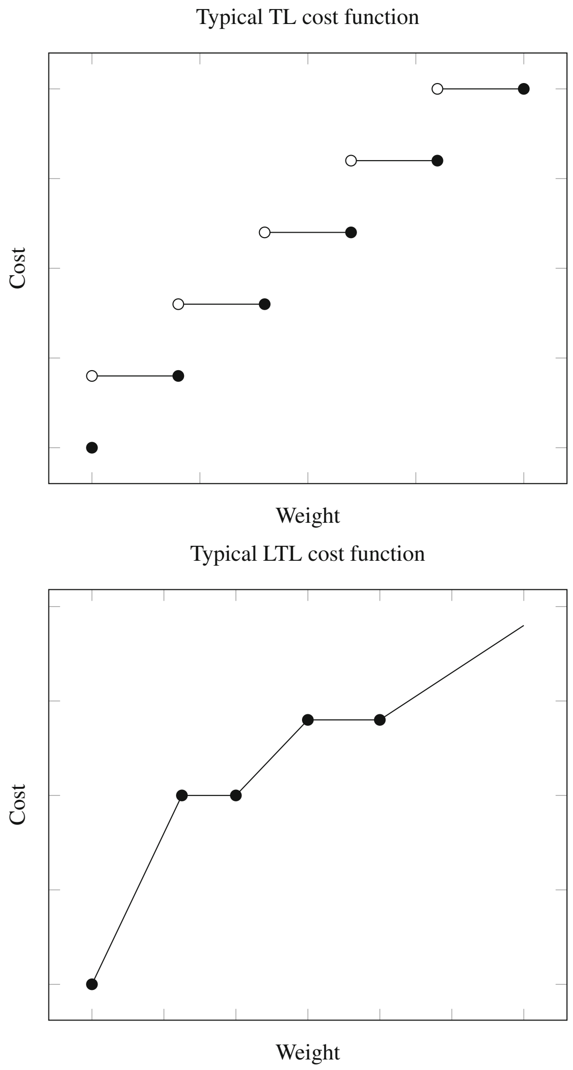


Fig. 6.1 Piecewise linear functions arising in TL and LTL transportation

instead, the costs display economies of scale according to the volume of transported goods, which yields non-convex piecewise linear cost functions. An example of piecewise linear functions corresponding to TL and to LTL contexts in multimodal applications is depicted in Fig. 6.1.

Every reasonable cost-of-capacity function can be approximated as a piecewise linear (possibly, non-convex) one, where the extent of the approximation can be tightly controlled at the cost of the number of breakpoints. Therefore, network design models with piecewise linear arc cost-of-capacity functions can model a large variety of real-world applications. This chapter focuses on such models, considering both a very general case and some more restricted ones that serve to illustrate the main concepts. We show that the best models in terms of tightness of the LP relaxation bound suffer from a significant increase in the number of variables, even more so than the already rather large ones corresponding to simpler cases. Because of this, we review the use of techniques that allow to efficiently solve very large formulations by dynamically generating smaller portions of them. These techniques are instrumental for the practical solution of network design problems with piecewise linear costs.

In Sect. 2, we present formulations for a generic piecewise linear cost network design problem, as well as for the single-facility multicommodity network design problem, a special case of the multifacility multicommodity network design problem studied in Chap. 5. Since these formulations involve very large numbers of variables and constraints, we have to rely on algorithms that implement column-and-row generation. In Sect. 3, we present a framework, called structured Dantzig-Wolfe decomposition, to develop such algorithms.

2 Formulations with Piecewise Linear Costs

We first present a generic network design model with piecewise linear costs. We then present how a particular case of this model can be derived from the special case of the single-facility network design model introduced in Chap. 5. This allows us to give a polyhedral interpretation of this piecewise linear model, comparing it with the formulation with general integer variables shown in Chap. 5.

2.1 *Generic Piecewise Linear Cost Network Design Formulation*

Let $\mathscr{G} = (\mathscr{N}, \mathscr{A})$ be a directed graph, where $\mathscr{N}$ is the set of nodes and $\mathscr{A} \subseteq \mathscr{N} \times \mathscr{N}$ is the set of potential arcs. A maximum flow capacity $u_{ij} > 0$ is associated with each arc $(i, j) \in \mathscr{A}$. Customarily, several commodities, represented by the set $\mathscr{K}$, share the flow capacity on $\mathscr{G}$. A commodity $k \in \mathscr{K}$ is identified by the amount d^k of flow to be shipped between the origin $O(k)$ and the destination $D(k)$; equivalently, as in Chap. 3, the commodity demand w_i^k at each node $i \in \mathscr{N}$ is set to d^k if $i = O(k)$, to $-d^k$ if $i = D(k)$, and to 0 otherwise. Using the notation $x_{ij} = \sum_{k \in \mathscr{K}} x_{ij}^k$ for the total flow on each arc $(i, j) \in \mathscr{A}$, we assume that we have to minimize an objective function that is separable into $|\mathscr{A}|$ piecewise linear cost functions $g_{ij}(x_{ij})$ such that:

- $g_{ij}(0) = 0$;
- $g_{ij}(x_{ij})$ is lower semi-continuous;
- $g_{ij}(x_{ij})$ is non-decreasing.

This last requirement is often not strictly necessary but, besides being reasonable in applications, it can typically be assumed without loss of generality: if some larger capacity point would cost less than a smaller one, one would never buy the latter. The piecewise linear g_{ij} can be described by means of a finite number $s(i, j) \geq 1$ of breakpoints $0 = e_{ij}^0 < e_{ij}^1 < \cdots < e_{ij}^{s(i,j)} = u_{ij}$. In other words, each $s \in \mathscr{S}_{ij} = \{1, \ldots, s(i, j)\}$ identifies one *segment* $[e_{ij}^{s-1}, e_{ij}^s]$ where the function is characterized by a linear cost (or slope) $c_{ij}^s \geq 0$ and a fixed cost (or intercept) $f_{ij}^s \geq 0$. The resulting problem, which we call the *piecewise linear cost network design* problem (*PLCND*), can be seen as a generalization of the multicommodity capacitated fixed-charge network design problem (*MCFND*) studied in Chaps. 2, 3, and 4.

An arc-based mixed-integer linear programming (MILP) formulation can be derived for the *PLCND* using the so-called *multiple choice model* to represent piecewise linear functions. This modeling technique introduces two variables for each $(i, j) \in \mathscr{A}$ and for each $s \in \mathscr{S}_{ij}$: x_{ij}^s is the total flow x_{ij} on arc $(i, j) \in \mathscr{A}$, if it falls in segment s, and is equal to 0 otherwise; y_{ij}^s is equal to 1, if $x_{ij}^s = x_{ij}$, and is equal to 0 otherwise. The *basic model* for the *PLCND* can then be written as:

$$\text{Minimize} \quad \textstyle\sum_{(i,j)\in\mathscr{A}} \sum_{s\in\mathscr{S}_{ij}} f_{ij}^s y_{ij}^s + \sum_{(i,j)\in\mathscr{A}} \sum_{s\in\mathscr{S}_{ij}} c_{ij}^s x_{ij}^s \tag{6.1}$$

$$\text{Subject to} \quad \textstyle\sum_{j\in\mathscr{N}_i^+} x_{ij}^k - \sum_{j\in\mathscr{N}_i^-} x_{ji}^k = w_i^k, \quad \forall i \in \mathscr{N}, \forall k \in \mathscr{K}, \tag{6.2}$$

$$x_{ij}^k \geq 0 \forall (i, j) \in \mathscr{A}, \ \forall k \in \mathscr{K}, \tag{6.3}$$

$$\textstyle\sum_{k\in\mathscr{K}} x_{ij}^k = \sum_{s\in\mathscr{S}_{ij}} x_{ij}^s, \quad \forall (i, j) \in \mathscr{A}, \tag{6.4}$$

$$e_{ij}^{s-1} y_{ij}^s \leq x_{ij}^s \leq e_{ij}^s y_{ij}^s \forall (i, j) \in \mathscr{A}, \ \forall s \in \mathscr{S}_{ij}, \tag{6.5}$$

$$\textstyle\sum_{s\in\mathscr{S}_{ij}} y_{ij}^s \leq 1, \quad \forall (i, j) \in \mathscr{A}, \tag{6.6}$$

$$y_{ij}^s \in \{0, 1\} \forall (i, j) \in \mathscr{A}, \ \forall s \in \mathscr{S}_{ij}. \tag{6.7}$$

The objective function (6.1) minimizes the total piecewise linear costs, computed as the sum of the total fixed and linear costs for arcs and segments included in the optimal solution. Constraints (6.2) are the flow conservation equations for each node and each commodity. Constraints (6.4)–(6.7) capture the definition of the variables x_{ij}^s and y_{ij}^s. Note that the capacity constraints now appear as one of the constraints in (6.5), i.e., the one corresponding to $s = s(i, j)$.

When $s(i, j) = 1$ for each $(i, j) \in \mathscr{A}$, we obtain the arc-based model for the *MCFND*, for the special case of the latter where $c_{ij}^k = c_{ij}$ for each arc $(i, j) \in \mathscr{A}$ and each commodity $k \in \mathscr{K}$. In particular, the corresponding LP relaxation is the so-called weak relaxation, which is known to give a lower bound that is generally far from the optimal value, as seen in Chap. 2. This is true also in the general case where $s(i, j) > 1$. In order to derive tighter LP relaxations, we need either a different approach than the multiple choice model to represent piecewise linear costs, or to add valid inequalities (and possibly additional variables). Unfortunately, the multiple choice model is known to be one of a few different ways to "optimally" represent a piecewise linear (non-convex) function, in that its continuous relaxation (a convex problem) represents the *convex envelope* of the function, i.e., its tightest possible convex approximation. In other words, all other known ways of modeling piecewise linear costs with MILP formulations also end up giving a lower bound that is as weak as the one given by the LP relaxation of the basic model. This, however, only holds while keeping the rest of the formulation unchanged. Indeed, by adding variables, we can introduce valid inequalities that generalize the strong linking constraints derived for the *MCFND*, and therefore significantly strengthen the formulation.

We introduce the additional variables x_{ij}^{ks} for each arc $(i, j) \in \mathscr{A}$, for each commodity $k \in \mathscr{K}$ and for each segment $s \in \mathscr{S}_{ij}$, with the intended semantic that $x_{ij}^{ks} = x_{ij}^{k}$ if $x_{ij}^{s} > 0$, and $x_{ij}^{ks} = 0$ otherwise. These so-called *extended variables* are easily defined in terms of the previous ones:

$$\textstyle\sum_{k\in\mathscr{K}} x_{ij}^{ks} = x_{ij}^{s}, \qquad \forall (i, j) \in \mathscr{A},\ \forall s \in \mathscr{S}_{ij}, \tag{6.8}$$

$$\textstyle\sum_{s\in\mathscr{S}_{ij}} x_{ij}^{ks} = x_{ij}^{k}, \qquad \forall (i, j) \in \mathscr{A},\ \forall k \in \mathscr{K}. \tag{6.9}$$

Note that this reformulation is close to replicating each arc $(i, j) \in \mathscr{A}$ with as many "parallel" copies as there are segments, as hinted at in the Introduction. Indeed, clearly, the original variables x_{ij}^{s} and x_{ij}^{k} could be substituted away using (6.8) and (6.9), leaving only the x_{ij}^{ks} as flows in an expanded graph with $s(i, j)$ copies of each arc $(i, j) \in \mathscr{A}$. However, the two formulations are not the same due to constraints (6.6) that require only one of the copies to be picked up. Furthermore, just adding these additional variables does not improve, per se, the LP relaxation (and, instead, makes it much larger and more costly to solve); however, it allows to also add the valid inequalities

$$x_{ij}^{ks} \leq b_{ij}^{ks} y_{ij}^{s}, \qquad \forall (i, j) \in \mathscr{A},\ \forall k \in \mathscr{K},\ \forall s \in \mathscr{S}_{ij}, \tag{6.10}$$

where $b_{ij}^{ks} = \min\{e_{ij}^{s}, d^{k}\}$ is an upper bound on the flow of commodity $k \in \mathscr{K}$ on arc $(i, j) \in \mathscr{A}$ associated with segment $s \in \mathscr{S}_{ij}$. It is clear that these *extended linking constraints* generalize the strong linking constraints for the *MCFND*, i.e., when $s(i, j) = 1$ for each $(i, j) \in \mathscr{A}$ (see Chap. 2). The formulation obtained by adding (6.8)–(6.10) to the basic model is called the *extended model*.

As in the *MCFND*, the LP relaxation bound resulting from the addition of the extended variables and constraints is vastly better than that of the original model (6.1)–(6.7). However, the model itself is also way larger, which means that it cannot be efficiently solved with off-the-shelf LP technology as the size of the problem, and in particular the number of segments, increase. Dealing with such huge LPs requires *decomposition methods* like Lagrangian relaxation and Dantzig-Wolfe decomposition (see Chap. 3). We present such a decomposition method in Sect. 3.1, but we first look back at the single-facility multicommodity network design problem (see Chap. 5) and cast it as a special case of the *PLCND*.

2.2 Piecewise Linear Cost Model of the Single-Facility Problem

The special case we are dealing with is that where there is only one facility type (for each arc), which means that g_{ij} is a "uniform" staircase-structured piecewise linear function where all the segments are alike. The corresponding *single-facility multicommodity network design* problem (*SFMND*) is typically formulated with general integer variables y_{ij} that represent the *number of facilities* installed on each arc $(i, j) \in \mathscr{A}$, yielding

$$\text{Minimize} \quad \textstyle\sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + \sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}} c_{ij} x_{ij}^k \tag{6.11}$$

$$\text{Subject to} \quad \textstyle\sum_{j\in\mathscr{N}_i^+} x_{ij}^k - \sum_{j\in\mathscr{N}_i^-} x_{ji}^k = w_i^k, \quad \forall\, i \in \mathscr{N}, \forall\, k \in \mathscr{K}, \tag{6.12}$$

$$\textstyle\sum_{k\in\mathscr{K}} x_{ij}^k \le u_{ij} y_{ij}, \quad \forall\,(i, j) \in \mathscr{A}, \tag{6.13}$$

$$x_{ij}^k \ge 0, \quad \forall\,(i, j) \in \mathscr{A}, \forall\, k \in \mathscr{K}, \tag{6.14}$$

$$y_{ij} \ge 0 \text{ and integer}, \quad \forall\,(i, j) \in \mathscr{A}. \tag{6.15}$$

The objective function (6.11) comprises the cost of installing facilities, where each unit of facility on arc $(i, j) \in \mathscr{A}$ incurs a cost $f_{ij} \ge 0$, and the total transportation cost, where each unit of flow on arc $(i, j) \in \mathscr{A}$ incurs a cost $c_{ij} \ge 0$. More generally, transportation costs could also depend on the commodity, but we use commodity-independent transportation costs for consistency with the definition of the *PLCND* given in Sect. 2.1.

We can easily interpret this objective function as a piecewise linear function by defining a segment $s \in \mathscr{S}_{ij}$ for each positive number s of units of the facility installed on arc $(i, j) \in \mathscr{A}$. Clearly, we then have $s(i, j) = \lceil(\sum_{k\in\mathscr{K}} d^k)/u_{ij}\rceil$. In such interpretation, we define $f_{ij}^s = sf_{ij}$, $c_{ij}^s = c_{ij}$ and $e_{ij}^s = su_{ij}$, for each $(i, j) \in \mathscr{A}$ and $s \in \mathscr{S}_{ij}$. We use the following equations to relate the general integer variables in model (6.11)–(6.15) to the 0-1 variables used to model the *PLCND*:

$$\textstyle\sum_{s\in\mathscr{S}_{ij}} s y_{ij}^s = y_{ij}, \qquad \forall\,(i, j) \in \mathscr{A}. \tag{6.16}$$

By introducing the segment-based flow variables defined by Eqs. (6.4), we then derive a reformulation of (6.11)–(6.15) that corresponds to the basic model (6.1)–(6.7) for the *PLCND*. Clearly, we can also define extended variables and constraints to derive the stronger extended model defined by adding (6.8)–(6.10) to the basic model.

We now establish a relationship between the LP relaxation of this piecewise linear cost extended model (hereafter, simply called the *extended relaxation*) for the *SFMND* and the Lagrangian relaxation of the flow conservation equations (6.12) derived from the standard model (6.11)–(6.15). Denoting as $\pi = (\pi)_{i\in\mathscr{N}}^{k\in\mathscr{K}}$ the (unrestricted) Lagrange multipliers associated with these constraints, we obtain the following Lagrangian subproblem, by adding (redundant) upper bound constraints on each variable:

$$\text{Minimize}\quad \textstyle\sum_{(i,j)\in\mathscr{A}} f_{ij}y_{ij} + \sum_{k\in\mathscr{K}}\sum_{(i,j)\in\mathscr{A}}(c_{ij}+\pi_i^k-\pi_j^k)x_{ij}^k \tag{6.17}$$

Subject to (6.13)–(6.15)

$$x_{ij}^k \le d^k, \quad \forall\,(i,j)\in\mathscr{A},\ \forall\, k\in\mathscr{K}, \tag{6.18}$$

$$y_{ij} \le s(i,j), \quad \forall\,(i,j)\in\mathscr{A}. \tag{6.19}$$

Note that in the objective (6.17) the constant term "$-\sum_{i\in\mathscr{N}}\sum_{k\in\mathscr{K}}\pi_i^k w_i^k$" is omitted for better readability. This Lagrangian subproblem decomposes by arc. For each $(i,j)\in\mathscr{A}$, we obtain the following MILP model with a single general integer variable y_{ij}:

$$\text{Minimize}\quad f_{ij}y_{ij} + \textstyle\sum_{k\in\mathscr{K}}(c_{ij}+\pi_i^k-\pi_j^k)x_{ij}^k \tag{6.20}$$

$$\text{Subject to}\quad \textstyle\sum_{k\in\mathscr{K}} x_{ij}^k \le u_{ij}y_{ij}, \tag{6.21}$$

$$0 \le x_{ij}^k \le d^k, \qquad \forall\, k\in\mathscr{K}, \tag{6.22}$$

$$y_{ij} \in [\,0\,,\ s(i,j)\,]\ \text{and integer.} \tag{6.23}$$

We can solve this arc-based MILP model by a naïve approach that solves $s(i,j)$ continuous knapsack problems for every positive integer value $y_{ij}\in[\,0\,,\ s(i,j)\,]$, each providing an objective value $v(y_{ij})$. The optimal solution corresponds either to the value y_{ij} that achieves the minimum objective value $v(y_{ij})$, if this objective value is negative, or to $y_{ij}=0$, otherwise.

However, (6.20)–(6.23) can be solved more efficiently by observing that $v(y_{ij})$ corresponds to the value of the *Benders subproblem* obtained by partitioning the variables in the "natural" way, i.e., the single y_{ij} in the master and all the x_{ij}^k, for $k\in\mathscr{K}$, in the subproblem. It is well-known, from the theory of Benders decomposition, that $v(y_{ij})$ is a convex function of y_{ij}. This is true, in particular, if Benders decomposition is applied to the LP relaxation of the arc-based MILP model. Thus, $v(y_{ij})$ is a convex function of (*continuous*) $y_{ij}\in[\,0\,,\ s(i,j)\,]$. However, it is easy to realize that the y_{ij} component of the optimal solution of the continuous relaxation of (6.20)–(6.23) is immediately available by the closed formula $y_{ij} = (\sum_{k\in\mathscr{K}} x_{ij}^k)/u_{ij}$. This means that such continuous relaxation is in fact equivalent to

$$\text{Minimize } \left\{ \textstyle\sum_{k\in\mathscr{K}} (f_{ij}/u_{ij} + c_{ij} + \pi_i^k - \pi_j^k) x_{ij}^k \mid 0 \le x_{ij}^k \le d^k,\ k \in \mathscr{K} \right\}. \tag{6.24}$$

In turn, an optimal solution to (6.24) is immediately obtained by setting, for each $k \in \mathscr{K}$, $\overline{x}_{ij}^k = d^k$ if $(f_{ij}/u_{ij} + c_{ij} + \pi_i^k - \pi_j^k) < 0$, and $\overline{x}_{ij}^k = 0$ otherwise. Thus, the minimizer of the convex function $v(y_{ij})$ over the interval $[\,0\,,\ s(i, j)\,]$ is given by $\overline{y}_{ij} = (\sum_{k\in\mathscr{K}} \overline{x}_{ij}^k)/u_{ij}$. The convexity of the function $v(y_{ij})$ implies that an optimal *integer* solution is either $\lfloor \overline{y}_{ij} \rfloor$ or $\lceil \overline{y}_{ij} \rceil$. It suffices to solve the two corresponding continuous knapsack problems (in fact, only one if $\lfloor \overline{y}_{ij} \rfloor = 0$) to derive an optimal solution to (6.20)–(6.23). Both approaches have to solve continuous knapsack problems, but the naïve one requires a *pseudo-polynomial* number $(s(i, j) = \lceil (\sum_{k\in\mathscr{K}} d^k)/u_{ij} \rceil)$ of them, while the faster approach only requires a constant number (at most, two) of them.

Although not efficient, the naïve approach has a nice theoretical interpretation: for each arc $(i, j) \in \mathscr{A}$, the arc-based MILP model (6.20)–(6.23) decomposes for each value $y_{ij} \in \{1, \ldots, s(i, j)\}$. In other words, it corresponds to solving an optimization problem (with linear objective function) over a *finite union of bounded polyhedra* $\cup_{s\in\mathscr{S}_{ij}} \mathscr{P}_{ij}^s$, where

$$\mathscr{P}_{ij}^s = \{\, x_{ij} = (x_{ij}^k)^{k\in\mathscr{K}} \mid \textstyle\sum_{k\in\mathscr{K}} x_{ij}^k \le s u_{ij}\ ,\ 0 \le x_{ij}^k \le d^k,\ k \in \mathscr{K} \,\}.$$

There is a well-known modeling technique to formulate (linear) optimization problems over a finite union of bounded polyhedra as a *linear programming model*. It consists in associating to each polyhedron $\mathscr{P}_{ij}^s$ a continuous weight variable $\theta_{ij}^s \ge 0$, to be multiplied to the RHS of all constraints defining the polyhedron, and a "copy" ξ_{ij}^{ks} of each original continuous variable x_{ij}^k, $k \in \mathscr{K}$. Applying the technique yields the following *LP reformulation* of the arc-based MILP model (6.20)–(6.23):

$$\text{Minimize } \textstyle\sum_{s\in\mathscr{S}_{ij}} s f_{ij} \theta_{ij}^s + \sum_{k\in\mathscr{K}} \sum_{s\in\mathscr{S}_{ij}} (c_{ij} + \pi_i^k - \pi_j^k) \xi_{ij}^{ks} \tag{6.25}$$

$$\text{Subject to } \textstyle\sum_{k\in\mathscr{K}} \xi_{ij}^{ks} \le s u_{ij} \theta_{ij}^s, \quad \forall s \in \mathscr{S}_{ij}, \tag{6.26}$$

$$0 \le \xi_{ij}^{ks} \le d^k \theta_{ij}^s, \quad \forall k \in \mathscr{K}, \forall s \in \mathscr{S}_{ij}, \tag{6.27}$$

$$\textstyle\sum_{s\in\mathscr{S}_{ij}} \theta_{ij}^s \le 1, \tag{6.28}$$

$$\theta_{ij}^s \ge 0, \quad \forall s \in \mathscr{S}_{ij}. \tag{6.29}$$

Indeed, one can show that there exists an integer optimal solution, i.e., $\theta_{ij}^s \in \{0, 1\}$, $s \in \mathscr{S}_{ij}$, to this LP model. Hence, we can derive an optimal solution to the arc-based

MILP model (6.20)–(6.23) by using the equations:

$$\textstyle\sum_{s\in\mathscr{S}_{ij}} \xi_{ij}^{ks} = x_{ij}^{k}, \qquad \forall k \in \mathscr{K}, \tag{6.30}$$

$$\textstyle\sum_{s\in\mathscr{S}_{ij}} s\theta_{ij}^{s} = y_{ij}. \tag{6.31}$$

This implies that the *convex hull of feasible solutions to the arc-based MILP model* is given by the projection on the space of variables $(y_{ij}, (x_{ij}^{k})^{k\in\mathscr{K}})$ of the set of solutions that satisfy (6.26)–(6.31). The primal interpretation of Lagrangian duality (see Chap. 3) then allows us to express the Lagrangian dual as follows:

$$\text{Minimize} \quad \textstyle\sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + \sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}} c_{ij} x_{ij}^{k} \tag{6.32}$$

Subject to (6.12)

$$\textstyle\sum_{s\in\mathscr{S}_{ij}} \xi_{ij}^{ks} = x_{ij}^{k}, \quad \forall (i,j)\in\mathscr{A}, \forall k\in\mathscr{K}, \tag{6.33}$$

$$\textstyle\sum_{s\in\mathscr{S}_{ij}} s\theta_{ij}^{s} = y_{ij}, \quad \forall (i,j)\in\mathscr{A}, \tag{6.34}$$

$$\textstyle\sum_{k\in\mathscr{K}} \xi_{ij}^{ks} \le s u_{ij}\theta_{ij}^{s}, \quad \forall (i,j)\in\mathscr{A}, \forall s\in\mathscr{S}_{ij}, \tag{6.35}$$

$$0 \le \xi_{ij}^{ks} \le d^{k}\theta_{ij}^{s}, \quad \forall (i,j)\in\mathscr{A}, \forall k\in\mathscr{K}, \forall s\in\mathscr{S}_{ij}, \tag{6.36}$$

$$\textstyle\sum_{s\in\mathscr{S}_{ij}} \theta_{ij}^{s} \le 1, \quad \forall (i,j)\in\mathscr{A}, \tag{6.37}$$

$$\theta_{ij}^{s} \ge 0, \quad \forall (i,j)\in\mathscr{A}, \forall s\in\mathscr{S}_{ij}. \tag{6.38}$$

It is clear that any optimal solution to this LP model can be mapped into an optimal solution to the extended relaxation by the trivial identifications

$$\xi_{ij}^{ks} = x_{ij}^{ks}, \qquad \forall (i,j)\in\mathscr{A}, \forall k\in\mathscr{K}, \forall s\in\mathscr{S}_{ij}, \tag{6.39}$$

$$\theta_{ij}^{s} = y_{ij}^{s}, \qquad \forall (i,j)\in\mathscr{A}, \forall s\in\mathscr{S}_{ij}. \tag{6.40}$$

Indeed, the constraints are the same in both LP models, with two exceptions. First, the extended relaxation contains the additional constraints $\sum_{k\in\mathscr{K}} x_{ij}^{ks} \ge (s-1)u_{ij}y_{ij}^{s}$, $(i,j)\in\mathscr{A}, s\in\mathscr{S}_{ij}$. Since $f_{ij}\ge 0$, $(i,j)\in\mathscr{A}$, it is easy to see that an optimal solution to the extended relaxation satisfies

$$y_{ij}^{s} = \max\left\{\frac{\sum_{k\in\mathscr{K}} x_{ij}^{ks}}{s u_{ij}}, \max_{k\in\mathscr{K}}\left\{\frac{x_{ij}^{ks}}{b_{ij}^{ks}}\right\}\right\}$$

for all $(i,j)\in\mathscr{A}$ and $s\in\mathscr{S}_{ij}$, and one can then verify that the additional constraints are always satisfied. Second, the extended linking constraints (6.10), along with non-negativity constraints, imply inequalities (6.36); in turn, the latter, when combined with (6.35), imply the extended linking constraints.

To summarize, we have shown in this section that:

- the *SFMND* can be cast as a *PLCND*;
- the Lagrangian dual obtained by relaxing the flow conservation equations in the standard model for the *SFMND* (with general integer design variables) is equivalent to the extended relaxation for the *PLCND* reformulation of the problem (with 0-1 design variables).

For solving the *SFMND*, we propose to exploit its *PLCND* reformulation, more specifically to solve its extended relaxation by a column generation method where the pricing subproblem is the Lagrangian subproblem (6.20)–(6.23) obtained by relaxing the flow conservation equations. The next section presents a general framework for such a column generation method, which we call *structured Dantzig-Wolfe decomposition*, and its application both to the generic *PLCND* and to the *SFMND*.

3 Structured Dantzig-Wolfe Decomposition for Piecewise Linear Cost Network Design

We start by briefly recalling the basics of the Dantzig-Wolfe (DW) decomposition method and of its relationships with Lagrangian duality, as studied in Chap. 3. We describe the approach in the general setting

$$\min\{\, cx \mid Ax = b,\ x \in X\,\}\,.$$

The Lagrangian relaxation with respect to $Ax = b$ yields the Lagrangian subproblem

$$v(\pi) = \min\{\,(c - \pi A)x \mid x \in X\,\} + \pi b,$$

where π is the vector of Lagrange multipliers. The Lagrangian dual associated with this relaxation is $\max_\pi v(\pi)$, which can be reformulated as

$$(P) \quad \min\{\, cx \mid Ax = b,\ x \in conv(X)\,\}\,,$$

where the set $conv(X)$ is "easy" in the sense that linear optimization over $conv(X)$ is "significantly easier" than (P). For the sake of simplicity, we assume compactness, i.e., $X = \{\, x^1, \ldots x^t\,\}$ for finite (albeit possibly very large) t. The DW decomposition approach hinges on considering the *DW reformulation* of (P)

$$\text{Minimize} \quad \textstyle\sum_{h=1}^{t}(cx^h)\theta^h \tag{6.41}$$

$$\text{Subject to} \quad \textstyle\sum_{h=1}^{t}(Ax^h)\theta^h = b, \tag{6.42}$$

$$\textstyle\sum_{h=1}^{t} \theta^h = 1, \tag{6.43}$$

$$\theta^h \geq 0, \qquad \forall h = 1, \ldots, t. \tag{6.44}$$

Since the DW reformulation (6.41)–(6.44) typically has far too many variables to be solved in one blow, the DW decomposition approach constructs the *master problem* by restricting the problem to a selected small subset $\mathscr{B} \subset \{1, \ldots, t\}$ of them. The optimal dual solution π^* of constraints (6.42) in the master problem is then used to compute the *reduced* (or Lagrangian) costs $c - \pi^* A$, and linear optimization over $conv(X)$ (allegedly, an "easy" task) with these costs is performed. This yields a new point x^h; a simple test allows to establish whether adding the point to $\mathscr{B}$ may improve the value of the master problem, in which case this is done and the process iterates, or prove that the solution to the master problem was already optimal for (P), in which case the algorithm terminates. This is well-known to be equivalent to applying the cutting-plane approach to solving the Lagrangian dual with respect to the constraints $Ax = b$, as detailed in Chap. 3.

However, the standard version of the DW decomposition approach assumes that the "complicating" constraints $Ax = b$ can be written in the original x space, i.e., that they do not change when new points are added to $\mathscr{B}$. In other words, they are "few" even if the points needed to characterize X are "many"; this is, therefore, a standard *column generation* approach. However, in our application *both the columns and the rows* are "many", and need be generated incrementally. For this to be possible, X must have a specific structure that allows rows and columns to be generated simultaneously. We first present a generic treatment that describes which assumptions must be made on this structure, and then show why these assumptions are satisfied in our application.

3.1 Structured Dantzig-Wolfe Decomposition

The *structured Dantzig-Wolfe* (SDW) decomposition method requires the same underlying assumption as the DW one, i.e., that linear optimization over X is "easy." However, besides this, it requires the following three assumptions about the structure of the set X:

(i) *Existence of reformulation*: For a finite vector of variables θ and finite matrices C, Γ and γ of appropriate dimensions, $conv(X) = \{ x = C\theta \mid \Gamma\theta \leq \gamma \}$.

(ii) *Feasibility of zero padding*: Let $\mathscr{B} = (\mathscr{B}^c, \mathscr{B}^r)$, where $\mathscr{B}^c$ is a subset of the variables θ (columns of Γ and C) and $\mathscr{B}^r$ is a subset of the constraints (rows in Γ and γ) which impact *at least one* variable in $\mathscr{B}^c$, and denote by $\theta_{\mathscr{B}}$, $\Gamma_{\mathscr{B}}$, $\gamma_{\mathscr{B}}$ and $C_{\mathscr{B}}$ the corresponding restrictions of the data: if $\Gamma_{\mathscr{B}}\bar{\theta}_{\mathscr{B}} \leq \gamma_{\mathscr{B}}$ and $\theta = [\, \bar{\theta}_{\mathscr{B}} \,,\, 0 \,]$, then $\Gamma\theta \leq \gamma$.

(iii) *Easy extension of the approximation*: With

$$X_{\mathscr{B}} = \{ x = C_{\mathscr{B}}\theta_{\mathscr{B}} \mid \Gamma_{\mathscr{B}}\theta_{\mathscr{B}} \leq \gamma_{\mathscr{B}} \} ,$$

let $\bar{x}$ be a point such that $\bar{x} \in conv(X) \setminus X_{\mathscr{B}}$; then, it must be easy to update $\mathscr{B}$ and the associated $\Gamma_{\mathscr{B}}$, $\gamma_{\mathscr{B}}$ and $C_{\mathscr{B}}$ to a set $\mathscr{B}' \supset \mathscr{B}$ that satisfies (ii) such that there exists $\mathscr{B}'' \supseteq \mathscr{B}'$ with $\bar{x} \in X_{\mathscr{B}''}$.

We now comment on the assumptions. The first is inherent to the fact that, being a generalization of DW, the approach is based on a ("large") reformulation of the feasible set. Such a reformulation must be amenable to dynamic variable generation, i.e., variables fixed to 0 should not impair feasibility of a partial solution, which is what point (ii) entails. This clearly requires a specific structure on the constraints, as such a property is not true in general in a linear program, unless one requires that $\mathscr{B}^r$ contains *all* rows which impact (at least one of) the variables in $\mathscr{B}^c$, because then all relevant constraints are known. Such a requirement may be reasonable or even obvious in some cases: for instance, in the DW decomposition approach, there is only one constraint. In general, however, imposing this to hold might cause $\mathscr{B}^r$ to grow large very quickly. For instance, in the *PLCND*, each variable y_{ij}^s is involved in "many" extended linking constraints, and generating all of them—as much as generating all the corresponding flow variables x_{ij}^{ks}—could be impractical. The fundamental effect of assumption (ii) is to guarantee that $X_{\mathscr{B}} \subseteq X$. Finally, point (iii) states that, given a solution $\bar{x}$ that *cannot* be expressed by means of a given $\mathscr{B}$, it must be easy to update the latter in order to "capture a bit more of $\bar{x}$." This is purposely stated in a rather abstract form, but it is easy to see that this is possible in many cases, among which, of course, our application. Indeed, any $\bar{x} \in X$ can be represented by means of some of the variables θ; if $\bar{x} \notin X_{\mathscr{B}}$, it must be easy to reconstruct which of the necessary variables are missing in $\mathscr{B}^c$, and consequently, which rows should be added to $\mathscr{B}^r$ to ensure that (ii) is satisfied. Note that (iii) is carefully worded as not to require that enough variables are added to $\mathscr{B}^c$ so that $\bar{x}$ *immediately* becomes a point of $X_{\mathscr{B}}$; for the algorithm to work, it is only necessary to insert in $\mathscr{B}$ *at least one* of the "missing" variables. Since the set of variables is finite, iterating the process will ensure that eventually all the required variables (and constraints) will be generated.

With these assumptions, the SDW algorithm is easily described. It starts by solving the master problem

$$\min \{ cx \mid Ax = b, \; x = C_{\mathscr{B}}\theta_{\mathscr{B}}, \; \Gamma_{\mathscr{B}}\theta_{\mathscr{B}} \leq \gamma_{\mathscr{B}} \} \tag{6.45}$$

for an appropriately initialized set $\mathscr{B}$. For simplicity, we assume this meaning that (6.45) is nonempty (and therefore is has an optimal solution), although standard "phase 0" tricks can be used to relax this assumption. This provides the optimal dual solution π^* associated with $Ax = b$, which is used to compute the reduced costs $c - \pi^* A$. The Lagrangian subproblem with $\pi = \pi^*$ is then solved, i.e., the point $\bar{x}$ minimizing $(c - \pi^* A)x$ over X is computed: if $\bar{x} \in X_{\mathscr{B}}$ (a property that is easily tested by checking the corresponding optimal value against that of the master problem), then the algorithm terminates; otherwise, $\mathscr{B}$ is updated according to assumption (iii) and the process is repeated.

It is hardly necessary to mention that the original DW decomposition is a special case of this approach: indeed, in that case $X_{\mathscr{B}}$ is just $conv(\mathscr{B})$ (identifying each variable θ^h with the corresponding point x^h), which obviously satisfies assumptions (i) and (ii), and updating $\mathscr{B}$ only amounts at adding $\bar{x}$ to it, and hence assumption (iii) is satisfied as well. Directly generalizing the standard analysis of the DW decomposition method, it is easy to prove that the SDW decomposition is correct and complete as well, i.e., it finitely determines an optimal solution to (P). However, this means that SDW decomposition may also suffer from some of the issues that plague by-the-book implementations of DW decomposition, among which chiefly *instability*. As in DW decomposition, this can be tackled by *stabilizing* the master problem, adding appropriate constraints and/or terms in the objective function. Due to the "inherently richer" master problem, this may be less crucial for SDW decomposition than it is for the original DW decomposition, but it has been shown to be still quite useful. The technical details being somehow involved, we will not delve further into this topic.

3.2 *Application to Piecewise Linear Cost Network Design*

The application of SDW decomposition to the *PLCND* is based on the Lagrangian relaxation of the flow conservation equations (6.2) in the basic model (6.1)–(6.7) with Lagrange multipliers $\pi = (\pi)_{i\in\mathscr{N}}^{k\in\mathscr{K}}$. To obtain bounded Lagrangian subproblems, we add the upper bound constraints (6.18) on the flow variables: the resulting Lagrangian subproblem decomposes by arc, yielding for each arc $(i, j) \in \mathscr{A}$:

$$\text{Minimize} \quad \textstyle\sum_{s\in\mathscr{S}_{ij}} f_{ij}^s y_{ij}^s + \sum_{s\in\mathscr{S}_{ij}} c_{ij}^s x_{ij}^s + \sum_{k\in\mathscr{K}} (\pi_i^k - \pi_j^k) x_{ij}^k \tag{6.46}$$

$$\text{Subject to} \quad 0 \le x_{ij}^k \le d^k, \quad \forall k \in \mathscr{K}, \tag{6.47}$$

$$\textstyle\sum_{k\in\mathscr{K}} x_{ij}^k = \sum_{s\in\mathscr{S}_{ij}} x_{ij}^s, \tag{6.48}$$

$$e_{ij}^{s-1} y_{ij}^s \le x_{ij}^s \le e_{ij}^s y_{ij}^s, \quad \forall s \in \mathscr{S}_{ij}, \tag{6.49}$$

$$\textstyle\sum_{s\in\mathscr{S}_{ij}} y_{ij}^s \le 1, \tag{6.50}$$

$$y_{ij}^s \in \{0, 1\}, \quad \forall s \in \mathscr{S}_{ij}. \tag{6.51}$$

This subproblem can be easily solved by considering $s(i, j)$ continuous knapsack problems, each corresponding to a given segment $s \in \mathscr{S}_{ij}$:

$$\text{Minimize } v_{ij}^s = \sum_{k\in\mathcal{K}}(c_{ij}^s + \pi_i^k - \pi_j^k)x_{ij}^k \tag{6.52}$$

$$\text{Subject to } 0 \le x_{ij}^k \le b_{ij}^{ks}, \quad \forall k \in \mathcal{K}, \tag{6.53}$$

$$e_{ij}^{s-1} \le \sum_{k\in\mathcal{K}} x_{ij}^k \le e_{ij}^s. \tag{6.54}$$

If $\min_{s\in\mathcal{S}_{ij}}\{f_{ij}^s + v_{ij}^s\} \ge 0$, then the optimal solution to (6.46)–(6.51) is to set all variables equal to 0. Otherwise, let $s' = \arg\min_{s\in\mathcal{S}_{ij}}\{f_{ij}^s + v_{ij}^s\}$ and $\overline{x}_{ij} = (\overline{x}_{ij}^k)_{k\in\mathcal{K}}$ be the optimal solution to the continuous knapsack problem for s': then, the optimal solution to (6.46)–(6.51) is given by $y_{ij}^{s'} = 1$, $x_{ij}^{s'} = \sum_{k\in\mathcal{K}} \overline{x}_{ij}^k$, $x_{ij}^k = \overline{x}_{ij}^k$ for all $k \in \mathcal{K}$, and all other variables assuming the value 0.

This shows that (6.46)–(6.51) can be reformulated as a linear optimization problem over a finite union of bounded polyhedra $\cup_{s\in\mathcal{S}_{ij}} \mathcal{P}_{ij}^s$, each polyhedron $\mathcal{P}_{ij}^s$ being defined by (6.53)–(6.54). Using the same modeling technique described in Sect. 2.2, we can then reformulate (6.46)–(6.51) as

$$\text{Minimize } \sum_{s\in\mathcal{S}_{ij}} f_{ij}^s\theta_{ij}^s + \sum_{s\in\mathcal{S}_{ij}}\sum_{k\in\mathcal{K}}(c_{ij}^s + \pi_i^k - \pi_j^k)\xi_{ij}^{ks} \tag{6.55}$$

$$\text{Subject to } 0 \le \xi_{ij}^{ks} \le b_{ij}^{ks}\theta_{ij}^s, \quad \forall k \in \mathcal{K}, \tag{6.56}$$

$$e_{ij}^{s-1}\theta_{ij}^s \le \sum_{k\in\mathcal{K}} \xi_{ij}^{ks} \le e_{ij}^s\theta_{ij}^s, \quad \forall s \in \mathcal{S}_{ij}, \tag{6.57}$$

$$\sum_{s\in\mathcal{S}_{ij}} \theta_{ij}^s \le 1, \tag{6.58}$$

$$\theta_{ij}^s \ge 0, \quad \forall s \in \mathcal{S}_{ij}. \tag{6.59}$$

The variables θ_{ij}^s and ξ_{ij}^{ks} in (6.55)–(6.59) can be linked to those of the basic model by

$$\sum_{s\in\mathcal{S}_{ij}} \xi_{ij}^{ks} = x_{ij}^k, \quad \forall k \in \mathcal{K}, \tag{6.60}$$

$$\sum_{k\in\mathcal{K}} \xi_{ij}^{ks} = x_{ij}^s, \quad \forall s \in \mathcal{S}_{ij}, \tag{6.61}$$

$$\theta_{ij}^s = y_{ij}^s, \quad \forall s \in \mathcal{S}_{ij}. \tag{6.62}$$

Projecting out the x_{ij}^s variables using (6.61), this finally yields the primal form of the Lagrangian dual:

$$\text{Minimize } \sum_{(i,j)\in\mathcal{A}}\sum_{s\in\mathcal{S}_{ij}} f_{ij}^s y_{ij}^s + \sum_{(i,j)\in\mathcal{A}}\sum_{s\in\mathcal{S}_{ij}} c_{ij}^s \xi_{ij}^{ks} \tag{6.63}$$

Subject to (6.2)

$$\sum_{s\in\mathcal{S}_{ij}} \xi_{ij}^{ks} = x_{ij}^k, \quad \forall (i,j) \in \mathcal{A}, \forall k \in \mathcal{K}, \tag{6.64}$$

$$e_{ij}^{s-1}y_{ij}^s \le \sum_{k\in\mathcal{K}} \xi_{ij}^{ks} \le e_{ij}^s y_{ij}^s, \quad \forall (i,j) \in \mathcal{A}, \forall s \in \mathcal{S}_{ij}, \tag{6.65}$$

$$0 \leq \xi_{ij}^{ks} \leq b_{ij}^{ks} y_{ij}^{s}, \quad \forall (i,j) \in \mathscr{A}, \forall k \in \mathscr{K}, \forall s \in \mathscr{S}_{ij}, \tag{6.66}$$

$$\textstyle\sum_{s \in \mathscr{S}_{ij}} y_{ij}^{s} \leq 1, \quad \forall (i,j) \in \mathscr{A}, \tag{6.67}$$

$$y_{ij}^{s} \geq 0, \quad \forall (i,j) \in \mathscr{A}, \forall s \in \mathscr{S}_{ij}. \tag{6.68}$$

This is precisely the extended relaxation of the *PLCND*. Thus, we generalized the result of Sect. 2.2 for the *SFMND*: the Lagrangian dual with respect to the relaxation of the flow conservation equations is equivalent to the extended relaxation.

Exploiting this polyhedral result, we propose to solve the extended relaxation by a SDW decomposition algorithm. In this setting, the constraints $Ax = b$ correspond to the flow conservation equations (6.2), while the set X is the feasible domain of the Lagrangian subproblem. The latter separates by arc and, for each arc $(i, j) \in \mathscr{A}$, the arc-based MILP model can be solved with an LP reformulation: this means that we have a complete description of $conv(X)$, as required by assumption (i) of the SDW decomposition framework. Assumption (ii) is satisfied as well: due to (6.65) and (6.66), fixing to 0 all variables corresponding to one segment s satisfies all the constraints in which the variables are involved. Thus, one can identify $\mathscr{B}^c$ as a (possibly, empty) subset of the set of segments for each arc. If a segment s is present in $\mathscr{B}^c$ for some arc $(i, j) \in \mathscr{A}$, then the corresponding variable y_{ij}^s and *some* variables ξ_{ij}^{ks} are added to the master problem, and the corresponding constraints (6.65) and (6.66) are added to $\mathscr{B}^r$. Furthermore, whenever a variable ξ_{ij}^{ks} for some $k \in \mathscr{K}$ is present in $\mathscr{B}^c$, then also the corresponding x_{ij}^k and (6.64) are added to $\mathscr{B}^c$ and $\mathscr{B}^r$, respectively (if not present there already). Once the Lagrange multipliers π are fixed, the problem (6.46)–(6.51) can be efficiently solved as described (which is the fundamental assumption), thus identifying the crucial segment s' and the solution $\bar{x}_{ij} = (\bar{x}_{ij}^k)_{k \in \mathscr{K}}$. If s' is not in $\mathscr{B}^c$ already, it is added to it, together with the ξ_{ij}^{ks} necessary to reproduce the $\bar{x}_{ij}^k > 0$; the corresponding constraints are also added to $\mathscr{B}^r$. There can actually be variants of this approach; for instance, assumption (iii) only requires that *one* of the missing segments is added to the corresponding arc, thereby reducing the number of variables and constraints added at each iteration (but possibly at the cost of more iterations). Conversely, once a segment s' is generated, one could generate all the ξ_{ij}^{ks} corresponding to *all* $k \in \mathscr{K}$, irrespectively of the fact that $\bar{x}_{ij}^k > 0$, thereby adding more variables (but possibly reducing the number of iterations). The corresponding trade-offs between master problem size and iterations count can be optimized via computational experiments, and we do not add further details here; however, it is easy to see how assumption (iii) of the SDW decomposition framework is easily satisfied in our application.

The SDW decomposition algorithm for the *PLCND* applies directly to the *SFMND*, with the only difference that the Lagrangian subproblem can be solved even more efficiently. As we have seen in Sect. 2.2, the arc-based MILP model can be solved by considering at most *two* continuous knapsack problems, instead of $\mathscr{S}_{ij}$ in the general case. However, once this is done, the update of $\mathscr{B}^c$ and $\mathscr{B}^r$ can be performed in the same way. Note that in this case the extended formulation has a *pseudo-polynomial* size with respect to the size of the original problem, which means that its dynamic generation via the SDW decomposition approach is clearly instrumental.

4 Bibliographical Notes

Piecewise linear costs appear in a large number of applications in logistics and transportation, in particular multimodal transportation, where they can be used to model economies of scale, as in LTL transportation, or the use of multiple vehicles on a single lane, as in TL transportation; see, e.g., Balakrishnan and Graves (1989) and Croxton et al. (2003a). In the context of network flow problems, arc separable piecewise linear costs are often represented with as many parallel arcs as there are segments of the cost function. In general, standard fixed-cost network design problems cannot be obtained in this way, since it is still necessary to add constraints to ensure that no more than one parallel arc, representing the cost function for any given arc, is selected. However, when the cost function is concave, such constraints are not needed, since, in that case, the linear costs c_{ij}^s for each arc (i, j) are decreasing as s increases and the "right" parallel arc is automatically selected in an optimal solution. This observation has been made for both single-commodity (Kim and Pardalos 2000a) and multicommodity problems (Balakrishnan and Graves 1989). In particular, when the concave-cost problem is also uncapacitated, there is no need to add capacity constraints linking together parallel arcs: the problem then reduces to the *SUFND*, in the single-commodity case, or to the *MUFND*, in the multicommodity case (see Chap. 2 for definitions of these problems).

Problems with piecewise linear costs are typically modeled using MILP formulation techniques, giving rise to various models, including three "classical" ones: the convex combination (Manne and Markovitz 1957), incremental (Dantzig 1960) and multiple choice models. Croxton et al. (2003b) shows that the LP relaxations of the three formulations approximate the convex envelope of the cost function (see also Keha et al. (2004)). The multiple choice model can be derived from the model for the union of polyhedra, due to Balas (1979) and studied in Jeroslow and Lowe (1984). Using the same modeling approach, Vielma et al. (2010) extended to the non-separable case the different models for piecewise linear costs, usually restricted to the separable case. Since these different modeling techniques involve the introduction of auxiliary variables, in the same order as the total number of segments in all cost functions, a lot of effort has been dedicated in the last decade to the development of models that preserve the strength of LP relaxations, while reducing the number of auxiliary variables needed (see, e.g., Vielma et al. 2010; Vielma 2018, 2019; Huchette and Vielma 2019). In some sense, the modeling and algorithmic approaches described in this chapter take a completely different viewpoint: more auxiliary variables are added to the models to derive tighter relaxations, through the addition of constraints involving these auxiliary variables. To handle the resulting very large-scale models, column-and-row generation algorithms are then developed.

The extended model presented in Sect. 2.1 first appeared in Balakrishnan and Graves (1989), where it is used as a basis for a Lagrangian heuristic for solving a piecewise linear concave cost multicommodity flow problem. In Croxton et al. (2007), the extended model is compared to the basic one, as well as to an intermediate model (called "strong model") that does not introduce the extended variables, but rather adds to the basic model the following valid inequalities, which also generalize the strong linking constraints for the *MCFND*, but yield a weaker LP relaxation than the extended formulation:

$$x_{ij}^k \leq d^k \sum_{s \in \mathscr{S}_{ij}} y_{ij}^s, \qquad \forall (i, j) \in \mathscr{A},\ \forall k \in \mathscr{K}. \tag{6.69}$$

The reformulation of the *SFMND* as a *PLCND* was studied in Frangioni and Gendron (2009), where the equivalence between the extended relaxation and the Lagrangian dual with respect to the relaxation of flow conservation equations is shown. As a corollary of this equivalence, the extended relaxation is shown to be equivalent to the LP relaxation of the *SFMND* with the addition of the residual capacity inequalities (see Chapter 5). The proof exploits an argument used in Croxton et al. (2007) to show that the extended relaxation approximates the objective function by its convex envelope in the space of commodity flows. The proof based on the model to represent a finite union of bounded polyhedra (Balas 1979) first appeared in Khuong (2013). For the *SFMND* studied in Sect. 2.2, the observation that the Lagrangian subproblem resulting from the relaxation of flow conservation equations reduces to two continuous knapsack problems can be found in Atamtürk and Rajan (2002), while the interpretation of $v(y_{ij})$ as the value of a Benders subproblem (then, necessarily convex) appeared in Gendron (2019).

The SDW decomposition approach outlined in Sect. 3.1 has been proposed, in the context of the *SFMND*, in Frangioni and Gendron (2009). Stabilization of the approach, along some of the many possible lines that have been applied to the original DW decomposition algorithm (Frangioni 2005, 2020), has been studied in Frangioni and Gendron (2013) and shown to actually improve the computational efficiency of the method in practice. Its application to the *PLCND*, presented in Sect. 3.2, generalizes the approach applied to the *SFMND*. For the *PLCND*, the equivalence between the extended relaxation and the Lagrangian dual with respect to the flow conservation equations appeared in Croxton et al. (2007), but the proof based on the model for a finite union of bounded polyhedra (Balas 1979) is new.

The literature also contains several problems closely related to the *PLCND*, including the single-commodity piecewise linear network design problem (Kim and Pardalos 2000b); the multicommodity piecewise convex network design problem (Mahey and de Souza 2017); the unsplittable multicommodity piecewise linear network design problem (Fortz et al. 2017); the multicommodity piecewise linear network design problem with integer flows (Gendron and Gouveia 2017).

5 Conclusions and Perspectives

In this chapter, we studied the piecewise linear cost network design problem (*PLCND*). While modeling piecewise linear costs with MILP techniques is a classical topic, recent research in this area has favored approaches that reduce the number of auxiliary variables needed. The models that we have presented take a completely different viewpoint: they increase the number of auxiliary variables to generate valid inequalities that can then improve the LP relaxations. We rely on the structured Dantzig-Wolfe decomposition (SDW), which implements column-and-row generation, to solve the large-scale models that result from the introduction of auxiliary variables and their associated constraints.

Our developments in Sect. 2.2 show that the single-facility multicommodity network design problem (*SFMND*) can be cast as a *PLCND*. In particular, the extended relaxation can be shown to be as strong as the LP relaxation of the *SFMND* to which we add residual capacity inequalities (see Chap. 5). While the latter can be separated in linear time, the same can be said about the pricing subproblem for generating extended variables. The two approaches, a cutting-plane method to generate residual capacity inequalities and a SDW decomposition approach for the extended model, have been compared experimentally, showing the advantage of the latter for problems with a large number of commodities, but further research is needed. In particular, the development of exact B&P algorithms is a topic of investigation, as well as the adaptation of the other valid inequalities studied in Chap. 5 to the 0-1 reformulation of the *SFMND*. In addition, the multifacility multicommodity network design problem studied in Chap. 5 can also be cast as a *PLCND*, which opens other avenues of research.

The SDW decomposition approach is quite general, but its application has been limited so far to the extended reformulation for the *SFMND*. The algorithm is particularly well-adapted to many other network design problems. In particular, we might think of models where commodities are initially aggregated (say, by origins) and gradually disaggregated to introduce additional variables and valid inequalities. Also, in the context of service network design (see Chap. 12), where space-time networks are used, we might exploit SDW decomposition to generate time-discretized flow variables and associated constraints in a dynamic way.

References

Atamtürk, A., & Rajan, D. (2002). On splittable and unsplittable flow capacitated network design arc-set polyhedra. *Mathematical Programming A, 92*, 315–333.

Balakrishnan, A., & Graves, S. C. (1989). A composite algorithm for a concave-cost network flow problem. *Networks, 19*, 175–202.

Balas, E. (1979). Disjunctive programming. *Annals of Discrete Mathematics, 5*, 3–51.

Croxton, K. L., Gendron, B., & Magnanti, T. L. (2003a). Models and methods for merge-in-transit operations. *Transportation Science, 37*(1), 1–22.

Croxton, K. L., Gendron, B., & Magnanti, T. L. (2003b). A comparison of mixed-integer programming models for nonconvex piecewise linear cost minimization problems. *Management Science, 49*(9), 1268–1273.

Croxton, K. L., Gendron, B., & Magnanti, T. L. (2007). Variable disaggregation in network flow problems with piecewise linear costs. *Operations Research, 55*(1), 146–157.

Dantzig, G. B. (1960). On the significance of solving linear programming problems with some integer variables. *Econometrica, 28*, 30–44.

Fortz, B., Gouveia, L., & Joyce-Moniz, M. (2017). Models for the piecewise linear unsplittable multicommodity flow problems. *European Journal of Operational Research, 261*, 30–42.

Frangioni, A. (2005). About Lagrangian methods in integer optimization. *Annals of Operations Research, 139*, 163–193.

Frangioni, A. (2020). Standard bundle methods: untrusted models and duality. In A. M. Bagirov, M. Gaudioso, N. Karmitsa, M. Mäkelä, & S. Taheri (Eds.), Numerical nonsmooth optimization: state of the art algorithms (pp. 61–116). Cham: Springer.

Frangioni, A., & Gendron, B. (2009). 0-1 reformulations of the multicommodity capacitated network design problem. *Discrete Applied Mathematics, 157*(6), 1229–1241.

Frangioni, A., & Gendron, B. (2013). A stabilized structured Dantzig-Wolfe decomposition method. *Mathematical Programming B, 140*, 45–76.

Gendron, B. (2019). Revisiting Lagrangian relaxation for network design. *Discrete Applied Mathematics, 261*, 203–218.

Gendron, B., & Gouveia, L. (2017). Reformulations by discretization for piecewise linear integer multicommodity network flow problems. *Transportation Science, 51*(2), 629–649.

Huchette, J., & Vielma, J. P. (2019). A geometric way to build strong mixed-integer programming formulations. *Operations Research Letters, 47*, 601–606.

Jeroslow, R. G., & Lowe, J. K. (1984). Modeling with integer variables. *Mathematical Programming Studies, 22*, 167–184.

Keha, A. B., de Farias, I. R., & Nemhauser, G. L. (2004). Models for representing piecewise linear cost functions. *Operations Research Letters, 32*, 44–48.

Khuong, P. V. (2013). *Lagrangian-informed mixed integer programming reformulations*. PhD thesis, Département d'informatique et recherche opérationnelle, Université de Montréal

Kim, D., & Pardalos, P. (2000a). Dynamic slope scaling and trust interval techniques for solving concave piecewise linear network flow problems. *Networks, 35*(3), 216–222.

Kim, D., & Pardalos, P. (2000b). A dynamic domain contraction algorithm for nonconvex piecewise linear network flow problems. *Journal of Global Optimization, 17*, 225–234.

Mahey, P., & de Souza, M. C. (2017). Multicommodity network flows with nonconvex arc costs. *Pesquisa Operacional, 37*(3), 571–595.

Manne, A. S., & Markovitz, H. M. (1957). On the solution of discrete programming problems. *Econometrica, 25*, 84–110.

Vielma, J. P. (2018). Embedding formulations and complexity for unions of polyhedra. *Management Science, 64*(10), 4721–4734.

Vielma, J. P. (2019). Small and strong formulations for unions of convex sets from the cayley embedding. *Mathematical Programming A, 177*, 21–53.

Vielma, J. P., Ahmed, S., & Nemhauser, G. L. (2010). Mixed-integer models for nonseparable piecewise linear optimization: unifying framework and extensions. *Operations Research, 58*(2), 303–315.

Chapter 7
Topology-Constrained Network Design

Bernard Fortz

1 Introduction

Many network design problems considered in this book aim at optimizing simultaneously the decisions on opening links (with an associated fixed cost) and the capacity to allocate to these links in order to satisfy a set of demands, with a variable routing cost associated to these demands. However, in certain situations, the demand is not known in advance, or involves a lot of uncertainty, leading to an approach in two phases, where the topological design of the network (considering only fixed cost of opening links) is considered first, and the decisions on routing and capacity allocation taken in a second (later) stage. This approach is relevant when the fixed costs are high compared to routing and capacity costs, and/or when topological decisions do not affect too much capacity decisions. For example, in telecommunications, fiber optic cables have a virtually unlimited capacity, and the limitation of capacity arises from equipment placed in the nodes of the network (routing cards). While decisions to dig a trench to lay a cable are very costly and must be taken over a long term horizon, increasing capacity by adding or upgrading equipment into nodes is relatively simple and cheap.

In this chapter, we study models and techniques for long-term planning of the first phase, i.e., we only deal with topological aspects. Two main issues appear in the planning process of networks: economy and survivability. Economy refers to the construction cost, which is expressed as the sum of the edge costs, while survivability refers to the restoration of services in the event of node or link failure.

B. Fortz (✉)
Computer Science Department, Université libre de Bruxelles, Brussels, Belgium

INOCS, INRIA Lille Nord-Europe, Villeneuve-d'Ascq, France
e-mail: bernard.fortz@ulb.ac.be

The goal is then to determine a set of links connecting all nodes under some survivability criteria.

For example, in telecommunications, the network is seen as a given set of nodes and a set of possible fiber links that have to be placed between these nodes to achieve connectivity and survivability at minimum cost. Until about 25 years ago, the limited capacity of copper cables resulted in highly diverse routing. The developments in fiber-optic technology have led to components that are cheap and reliable, having an almost unlimited capacity. The introduction of such a technology has made hierarchical routing and bundling of traffic very attractive. This approach has resulted in sparse, even treelike network topologies with larger amounts of traffic carried by each link. Trees satisfy the primary goal of minimizing the total cost while connecting all nodes. However, only one node or edge breakdown causes a tree network to fail in its main objective of enabling communication between all pairs of nodes.

This means that some survivability constraints have to be considered while building the network. Losing end-to-end customer service could lead to dramatic loss of revenue for commercial service providers. Constructing network topologies that provide protection against failures has become one of the most important problems in the field of network design.

The most studied models deal with k-connectivity requirements, i.e., the ability to restore network service in the event of a failure of at most $k - 1$ components of the network. Among them, the minimum-cost two-connected spanning network problem consists in finding a network with minimal total cost for which two node-disjoint paths are available between every pair of nodes. This means that two-connected networks are able to deal with a single link or node failure. Two-connected networks have been found to provide a sufficient level of survivability in most cases, and a considerable amount of research has focused on so-called *low-connectivity constrained* network design problems, i.e., problems for which each node j is characterized by a requirement $r_j \in \{0, 1, 2\}$ and $\min\{r_i, r_j\}$ node-disjoint paths between every pair of nodes i, j are required.

Two-connectivity seems a sufficient level of survivability for most networks, since the probability of dealing with two simultaneous failures is usually very low for fiber optics technologies used in telecommunications networks. However, it turns out that the optimal solution of this problem is often very sparse (in many cases such as a Hamiltonian cycle). In such a topology, primary routing paths and re-routing paths in case of failure might become very long. This introduces another difficulty as it causes large delays in the network.

To avoid this, two kinds of solutions have been proposed in the literature. The first one imposes a constraint on the length of the paths (in terms of number of links crossed), the so-called *hop-constrained* models. The second approach consists of imposing that each edge belongs to at least one cycle whose length is bounded by a given constant, which ensures the existence of an alternate short path in case of failure.

The chapter is organized as follows. After introducing in Sect. 2 the specific notation and some fundamental definitions used in the chapter, we begin by the simplest but fundamental problem: the design of connected networks (and in particular the minimum spanning tree problem) is covered in Sect. 3. Next we turn our attention to networks requiring a higher level of survivability in Sect. 4. Sections 5 and 6 consider problems in which the length of re-routing paths in case of failure is limited, by introducing hop constraints and rings of bounded lengths, respectively. Section 7 provides links to the relevant literature that was used as basis to this chapter, as well as interesting references to dig further. We conclude in Sect. 8 with some perspectives on future trends in topological network design.

2 Notation and Definitions

Most models considered in this chapter are based on *undirected graphs* as capacity is not involved and links are usually bi-directional. Therefore, the given sets of nodes and possible connections are represented by an undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ where $\mathcal{N}$ is the set of *nodes* and $\mathcal{E}$ is the set of *edges* that represent the possible pairs of nodes between which a direct connection can be established. The graph $\mathcal{G}$ may have parallel edges but should not contain loops. Throughout this chapter, $n = |\mathcal{N}|$ and $m = |\mathcal{E}|$ will denote the number of nodes and edges of $\mathcal{G}$.

Given a subset of nodes $\mathcal{S} \subset \mathcal{N}$, the edge set

$$\delta(\mathcal{S}) = \{\{i, j\} \in \mathcal{E} \mid i \in \mathcal{S},\ j \in \mathcal{N}\backslash\mathcal{S}\}$$

is called the *cut* induced by $\mathcal{S}$. We write $\delta_{\mathcal{G}}(\mathcal{S})$ to make clear—in case of possible ambiguities—with respect to which graph the cut induced by $\mathcal{S}$ is considered. For a single node $i \in \mathcal{N}$, we denote $\delta(i) = \delta(\{i\})$. The set

$$\mathcal{E}(\mathcal{S}) = \{\{i, j\} \in \mathcal{E} \mid i \in \mathcal{S},\ j \in \mathcal{S}\}$$

is the set of edges having both end nodes in $\mathcal{S}$. We denote by $\mathcal{G}(\mathcal{S}) = (\mathcal{S}, \mathcal{E}(\mathcal{S}))$ the subgraph induced by edges having both end nodes in $\mathcal{S}$. If $\mathcal{E}(\mathcal{S})$ is empty, $\mathcal{S}$ is an *independent set*. $\mathcal{G}/\mathcal{S}$ is the graph obtained from $\mathcal{G}$ by contracting the nodes in $\mathcal{S}$ to a new node w (retaining parallel edges). Given two subsets of nodes $\mathcal{S}_1$ and $\mathcal{S}_2$, $\mathcal{S}_1 \cap \mathcal{S}_2 = \emptyset$, the subset of edges having one endpoint in each subset is denoted by

$$[\mathcal{S}_1 : \mathcal{S}_2] = \{\{i, j\} \in \mathcal{E} \mid i \in \mathcal{S}_1,\ j \in \mathcal{S}_2\}.$$

We denote by $\mathcal{N} - z = \mathcal{N}\backslash\{z\}$ and $\mathcal{E} - e = \mathcal{E}\backslash\{e\}$ the subsets obtained by removing one node or one edge from the set of nodes or edges. $\mathcal{G} - z$ denotes the graph $(\mathcal{N} - z, \mathcal{E}\backslash\delta(z))$, i.e., the graph obtained by removing a node z and its incident edges from $\mathcal{G}$. This is extended to a subset $\mathcal{Z} \subset \mathcal{N}$ of nodes by the notation $\mathcal{G} - \mathcal{Z} = (\mathcal{N}\backslash\mathcal{Z}, \mathcal{E}\backslash(\delta(\mathcal{Z}) \cup \mathcal{E}(\mathcal{Z})))$.

Each edge $e = \{i, j\} \in \mathcal{E}$, has a *fixed cost* $c_e = c_{ij}$ representing the cost of establishing the direct link connection, and a *length* $d_e = d_{ij} = d(i, j)$. It is assumed throughout this work that these edge lengths satisfy the *triangle inequality*, i.e.,

$$d(i, j) + d(j, k) \geq d(i, k) \quad \text{for all } i, j, k \in \mathcal{N}.$$

The cost of a network $(\mathcal{N}, \mathcal{F})$ where $\mathcal{F} \subseteq \mathcal{E}$ is a subset of possible edges is denoted by $c(\mathcal{F}) = \sum_{e \in \mathcal{F}} c_e$. The *distance* between two nodes i and j in this network is denoted by $d_{\mathcal{F}}(i, j)$ and is given by the length of a shortest path linking these two nodes in $\mathcal{F}$.

Without loss of generality, all costs are assumed to be nonnegative, because an edge e with a negative cost c_e will be contained in any optimum solution.

For any pair of distinct nodes $s, t \in \mathcal{N}$, an $[s, t]$-*path* $\mathcal{P}$ is a sequence of nodes and edges $(v_0, e_1, v_1, e_2, \ldots, v_{l-1}, e_l, v_l)$, where each edge e_i is incident to the nodes v_{i-1} and v_i $(i = 1, \ldots, l)$, where $v_0 = s$ and $v_l = t$, and where no node or edge appears more than once in $\mathcal{P}$. A collection $\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_k$ of $[s, t]$-paths is called *edge-disjoint* if no edge appears in more than one path, and is called *node-disjoint* if no node (other than s and t) appears in more than one path. A *cycle* (containing s and t) is a set of two node-disjoint $[s, t]$-paths. A *Hamiltonian cycle* is a cycle using each node of the network exactly once. A graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ is *k-edge-connected* (resp., *k-node-connected*) if, for each pair s, t of distinct nodes, $\mathcal{G}$ contains at least k edge-disjoint (resp., node-disjoint) $[s, t]$-paths.

When the type of connectivity is not mentioned, we assume node-connectivity. The *edge connectivity* (resp., *node-connectivity*) of a graph is the maximal k for which it is k-edge-connected (resp., k-node-connected). A 1-edge-connected network is also 1-node-connected, and we call it simply *connected.* A cycle-free graph is a *forest* and a connected forest is a *tree*. A *connected component* of a graph is a maximal connected subgraph. If $\mathcal{G} - e$ has more connected components than $\mathcal{G}$ for some edge e, we call e a *bridge*. Similarly, if $\mathcal{Z}$ is a node set and $\mathcal{G} - \mathcal{Z}$ has more connected components than $\mathcal{G}$, we call $\mathcal{Z}$ an *articulation set* of $\mathcal{G}$. If a single node forms an articulation set, the node is called *articulation point.*

Node and edge-disjoint $[s, t]$-paths are related to cuts and articulation sets by Menger's theorem:

Theorem 1 (Menger (1927))

1. *In a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, there is no cut of size $k - 1$ or less disconnecting two given nodes s and t, if and only if there exist at least k edge-disjoint $[s, t]$-paths in $\mathcal{G}$.*
2. *Let s and t be two nonadjacent nodes in $\mathcal{G}$. Then there is no articulation set $\mathcal{Z}$ of size $k - 1$ or less disconnecting s and t, if and only if there exist at least k node-disjoint $[s, t]$-paths in $\mathcal{G}$.*

In order to formulate network design problems as integer linear programs, we associate with every subset $\mathcal{F} \subseteq \mathcal{E}$ an *incidence vector* $y^{\mathcal{F}} = (y_e^{\mathcal{F}})_{e \in \mathcal{E}} \in \{0, 1\}^{|\mathcal{E}|}$ by setting

$$y_e^{\mathcal{F}} = \begin{cases} 1 & \text{if } e \in \mathcal{F}, \\ 0 & \text{otherwise} . \end{cases}$$

Conversely, each vector $y \in \{0, 1\}^{|\mathcal{E}|}$ induces a subset

$$\mathcal{F}^y = \{e \in \mathcal{E} \mid y_e = 1\}$$

of the edge set $\mathcal{E}$. For any subset of edges $\mathcal{F} \subseteq \mathcal{E}$ we define

$$y(\mathcal{F}) = \sum_{e \in \mathcal{F}} y_e .$$

3 Connected Networks

A fundamental constraint in topological network design is to ensure all nodes can communicate. This translates into the constraint that a path must exist between any pair of nodes in the graph, or in other terms, the constructed graph must be connected.

The problem of finding a minimum cost connected network is polynomially solvable: if costs are non-negative, there exists an optimal solution with the minimum number $n - 1$ of edges in a connected graph, hence the problem reduces to the well-known *Minimum Spanning Tree* problem. A simple algorithm to solve it is the greedy algorithm: start with an empty solution; order the edges of $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ by increasing costs; iteratively consider each edge in this sorted list and add it to the solution if it does not form a cycle with the edges already selected.

The property that a spanning tree is a graph with $n - 1$ edges and without cycles is the basis of the greedy algorithm described above. This property also leads to an integer programming formulation of the problem. Let y_e be a binary variable indicating whether edge $e \in \mathcal{E}$ is part of the spanning tree. Then the minimum spanning tree problem can be formulated as

$$\text{Minimize} \quad \sum_{e \in \mathcal{E}} c_e y_e \tag{7.1}$$

$$\text{Subject to} \quad y(\mathcal{E}) = n - 1, \tag{7.2}$$

$$y(\mathcal{E}(\mathcal{S})) \leq |\mathcal{S}| - 1 \quad \forall \emptyset \neq \mathcal{S} \subset \mathcal{N}, \tag{7.3}$$

$$y_e \in \{0, 1\}, \qquad \forall e \in \mathcal{E}, \tag{7.4}$$

where constraint (7.2) imposes the cardinality constraint and constraints (7.3) are subtour elimination constraints that eliminate all possible cycles from the solution.

Although there is an exponential number of subtour elimination constraints, they can be separated in polynomial time by a simple minimum cut computation.

Primal-dual arguments can be used to show that the linear programming relaxation of formulation (7.1)–(7.4) is integer, and its extreme points coincide with the incidence vectors of spanning trees. Moreover, the same arguments can be used to prove the correctness of the greedy algorithm.

Another formulation for the minimum spanning tree problem is obtained by considering a spanning tree as a connected subgraph with $n-1$ edges. Connectivity can be imposed by forcing each cut in the graph to contain at least one edge, leading to the cut-set formulation

$$\text{Minimize} \sum_{e \in \mathcal{E}} c_e y_e \tag{7.5}$$

$$\text{Subject to} \quad y(\mathcal{E}) = n-1, \tag{7.6}$$

$$y(\delta(\mathcal{S})) \geq 1 \quad \forall \emptyset \neq \mathcal{S} \subset \mathcal{N}, \tag{7.7}$$

$$y_e \in \{0,1\}, \quad \forall e \in \mathcal{E}, \tag{7.8}$$

where subtour elimination constraints have been replaced by cut inequalities (7.7).

In general, the linear relaxation of (7.5)–(7.8) has fractional extreme points and therefore its polytope strictly contains the polytope induced by (7.1)–(7.4). Cut inequalities can be generalized: Consider a partition $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_p$ of $\mathcal{N}$ into p nonempty subsets. Any spanning tree contains at least $p-1$ edges joining the sets $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_p$, leading to the valid inequality

$$\frac{1}{2}\sum_{i=1}^{p} y(\delta(\mathcal{S}_i)) \geq p-1 \tag{7.9}$$

that is usually called *partition inequality* or *multi-cut inequality*. Cut inequalities (7.7) are a special case of (7.9) with $p = 2$. Again, it is possible to show that the polyhedron of the linear relaxation of the formulation given by (7.6), (7.8), and (7.9) has integer extreme points. Hence the linear relaxations of the multi-cut formulation and of the subtour formulation are equivalent and both define the convex hull of incidence vectors of spanning trees.

The subtour and multi-cut formulations are ideal, but are not usable as such in practice, as they suffer from an exponential number of constraints. However, a cutting-plane approach can be used, as both classes of inequalities can be separated in polynomial time.

Another approach is to model the problem with extended formulations, by introducing new sets of variables but keeping the number of constraints polynomial. One of these extended formulations makes use of directed flows between an arbitrarily chosen *root node* $r \in \mathcal{N}$ and all the other nodes to impose connectivity.

Indeed, the constructed graph is connected if and only if there exists a path from r to every other node $k \in \mathcal{N} \setminus \{r\}$. To this aim, let us consider each node $k \neq r$ as a commodity, where one unit of flow originates at node r and must be delivered to node k. In order to represent these flows, we define $\mathcal{A} = \{(i, j), (j, i) | \{i, j\} \in \mathcal{E}\}$ as the directed set of arcs obtained by replacing each edge by two arcs in opposite direction. As in previous chapters, let x_{ij}^k be the flow of commodity k in arc (i, j). We can then formulate the minimum spanning tree problem as:

$$\text{Minimize} \sum_{e \in \mathcal{E}} c_e y_e \tag{7.10}$$

$$\text{Subject to} \quad \sum_{j \in \mathcal{N}_i^+} x_{ij}^k - \sum_{j \in \mathcal{N}_i^-} x_{ji}^k = w_i^k, \quad \forall i \in \mathcal{N}, \forall k \in \mathcal{N} \setminus \{r\} \tag{7.11}$$

$$x_{ij}^k + x_{ji}^{k'} \leq y_e, \quad \forall e = \{i, j\} \in \mathcal{E}, \ \forall k, k' \in \mathcal{N} \setminus \{r\}, \tag{7.12}$$

$$y(\mathcal{E}) = n - 1, \tag{7.13}$$

$$x_{ij}^k \geq 0, \quad \forall (i, j) \in \mathcal{A}, \ \forall k \in \mathcal{N} \setminus \{r\}, \tag{7.14}$$

$$y_e \in \{0, 1\}, \quad \forall e \in \mathcal{E}, \tag{7.15}$$

where, for each $i \in \mathcal{N}$, we define

$$\mathcal{N}_i^+ = \{j \in \mathcal{N} : (i, j) \in \mathcal{A}\}, \quad \mathcal{N}_i^- = \{j \in \mathcal{N} : (j, i) \in \mathcal{A}\}. \tag{7.16}$$

Flow balance constraints (7.25) define a path between r and k with

$$w_i^k = \begin{cases} 1, \text{ if } i = r, \\ -1, \text{ if } i = k, \\ 0, \text{ otherwise.} \end{cases} \tag{7.17}$$

Each edge is given a natural direction as flows are directed away from the root r of the tree. Constraints (7.12) ensure that flow is sent only on edges present in the spanning tree, and always in the same direction. Finaly (7.13) are the usual cardinality constraints.

This formulation can also be seen as an application of the max flow-min cut theorem, and again leads to a complete description of the convex hull of incidence vectors of spanning trees (or to be more precise, the projection of the formulation on the space of y-variables defines this polyhedron).

Now consider the general case where costs are not restricted to be positive. Then a minimum cost connected network is not necessarily a tree, as all negative cost edges should belong to the optimal solution, possibly creating cycles. Hence the subtour formulation is not valid anymore. However, the multi-cut and the flow formulations become valid if cardinality constraints (7.6) and (7.13) are removed.

Moreover, the obtained formulations are ideal again in the sense that their linear relaxation (or its projection) describe the convex hull of incidence vectors of connected networks.

4 Survivable Networks

The major problem with the models presented above is that the topology tends to be sparse, as costs are minimized, the most extreme case being the minimum spanning tree. However, networks are subject to failures. For example, in the context of telecommunications, a network is seen as a set of gateway nodes (routers or telephone offices) and (fiber) links that are placed between nodes. If connectivity is the only constraint imposed on the network, a single link or node failure will disconnect it, which is clearly not acceptable.

In this context, survivability refers to the restoration of services in the event of node or link failure, or, in other words, a network is survivable if there exists a prespecified number of node-disjoint or edge-disjoint paths between any two nodes. Again, the only costs considered are construction costs, like the cost of digging trenches and placing a fiber cable into service.

A considerable amount of research has focused on low-connectivity constrained network design problems. These models can be described informally as follows: a set of nodes that have to be connected by a network is given. These nodes are classified according to their importance, namely the

- *special nodes*, for which a "high" degree of survivability has to be ensured in the network to be constructed;
- *ordinary nodes*, which have to be simply connected to the network;
- *optional nodes*, which may not be part of the network at all.

The pairs of nodes between which a direct transmission link can be placed are also given, together with the cost of placing the fiber cable and putting it into service. The problem now consists in determining where to place fiber cables so that the construction cost, i.e., the sum of the fiber cable costs, is minimized and certain survivability constraints are ensured. For instance, we may require that

- the destruction of any single link may not disconnect any two special nodes, or
- the destruction of any single node may not disconnect any two special nodes.

These requirements are equivalent to ask that there exist

- at least two edge-disjoint paths, or
- at least two node-disjoint paths

between any two special nodes.

Higher survivability levels may be imposed by requiring the existence of three or more paths between certain pairs of nodes according to their importance class.

However, up to now, low-connectivity requirements have been found to provide a sufficient level of survivability for telecommunications operators.

In graph-theoretic language, the set of nodes and possible link connections can be represented again by an undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$. The survivability requirement or importance of a node is modeled by node types. In particular, each node $s \in \mathcal{N}$ has an associated nonnegative integer r_s, the *type* of s. Sometimes, we also write $r(s)$ instead of r_s. A network $(\mathcal{N}, \mathcal{F})$, where $\mathcal{F} \subseteq \mathcal{E}$ is a subset of the possible links, is said to satisfy the *node-connectivity requirements*, if, for each pair $s, t \in \mathcal{N}$ of distinct nodes, $(\mathcal{N}, \mathcal{F})$ contains at least

$$r(s,t) = \min\{r_s, r_t\}$$

node-disjoint $[s, t]$-paths.

Similarly, we say that $(\mathcal{N}, \mathcal{F})$ satisfies the *edge-connectivity requirements*, if, for each pair $s, t \in \mathcal{N}$ of distinct nodes, the network contains at least $r(s, t)$ edge-disjoint $[s, t]$-paths. If all node types have the same value k, it is equivalent to request that $(\mathcal{N}, \mathcal{F})$ is k-node-connected or k-edge-connected.

We consider here the low-connectivity requirements, i.e., node types $r_s \in \{0, 1, 2\}$. Using our previous classification of nodes,

- special nodes have type 2,
- ordinary nodes have type 1, and
- optional nodes have type 0.

To shorten notation, we extend the type function r to sets by setting

$$\begin{aligned} r(\mathcal{S}) &= \max\{r_s \mid s \in \mathcal{S}\} \text{ for all } \mathcal{S} \subseteq \mathcal{N}, \text{ and} \\ \operatorname{con}(\mathcal{S}) &= \max\{r(s,t) \mid s \in \mathcal{S}, t \in \mathcal{N}\backslash\mathcal{S}\} \\ &= \min\{r(\mathcal{S}), r(\mathcal{N}\backslash\mathcal{S})\} \qquad \forall \mathcal{S} \subset \mathcal{N}, \emptyset \neq \mathcal{S} \neq \mathcal{N}. \end{aligned}$$

We write $\operatorname{con}_G(\mathcal{S})$ to make clear with respect to which graph $\operatorname{con}(\mathcal{S})$ is considered.

We can now formulate the connectivity constrained network design problem as the following integer linear program:

$$\text{Minimize } \sum_{e \in \mathcal{E}} c_e y_e \tag{7.18}$$

$$\text{Subject to} \quad y(\delta(\mathcal{S})) \geq \operatorname{con}(\mathcal{S}) \quad \forall \mathcal{S} \subset \mathcal{N},\ \emptyset \neq \mathcal{S} \neq \mathcal{N}, \tag{7.19}$$

$$y(\delta_{G-z}(\mathcal{S})) \geq \operatorname{con}_{G-z}(\mathcal{S}) \quad \forall z \in \mathcal{N},\ \mathcal{S} \subset \mathcal{N}\backslash\{z\}, \quad \emptyset \neq \mathcal{S} \neq \mathcal{N}\backslash\{z\}, \tag{7.20}$$

$$y_e \in \{0, 1\} \quad \forall e \in \mathcal{E}. \tag{7.21}$$

It follows from Menger's Theorem that, for any feasible solution y of this program, the subgraph $(\mathcal{N}, \mathcal{F}^y)$ of $\mathcal{G}$ defines a network satisfying the node-connectivity requirements. Removing (7.20), we obtain an integer linear program for edge-connectivity requirements. Inequalities (7.19) are called *cut inequalities*, while inequalities (7.20) are called *node cut inequalities*.

The connectivity constrained network design problem is NP-hard in general. In particular:

- If $r_s \in \{0, 1\}$, $\forall s \in \mathcal{N}$, it reduces to the well-known NP-hard Steiner tree problem in networks.
- If $r_s = 2$, $\forall s \in \mathcal{N}$, it consists in determining a minimum cost two-connected network. This last problem is NP-hard even if the graph is complete and costs satisfy the triangle inequality, since with an algorithm for this problem, one could decide whether a graph has a Hamiltonian cycle by associating a cost equal to 1 to all graph edges and cost equal to 2 to all non-graph edges.

However, for some particular connectivity requirements or costs, or when the underlying graph $\mathcal{G}$ is restricted, the problem may become polynomially solvable:

- If $r_s = 1$, $\forall s \in \mathcal{N}$, the problem reduces to the minimum cost connected network problem studied in the previous section.
- If $r_s = 1$ for exactly two nodes of $\mathcal{N}$ and $r_s = 0$ for all the other nodes, the problem becomes a shortest path problem.
- If $r_s = k$, $k \geq 2$, for exactly two nodes of $\mathcal{N}$ and $r_s = 0$ for all the other nodes, the problem becomes a k-shortest paths problem.
- If $r_s \in \{0, 1\}$, $\forall s \in \mathcal{N}$, the problem reduces to the Steiner tree problem in networks. This problem is NP-hard in general, but solvable in polynomial time in the case where either the number of nodes of type 0 or the number of nodes of type 1 is restricted.

The general formulation described above and many of its special cases received a lot of attention. Polyhedral results have been obtained for different special cases of the model (see Sect. 7 for references to surveys on the subject).

An important class of valid inequalities for k-node-connected networks are obtained by a generalization of partition inequalities (7.9). First, we observe that the deletion of $k-1$ nodes from a k-node-connected network leaves a connected graph. Thus, if $\mathcal{Z} \subseteq \mathcal{N}$ is a node set with exactly $k-1$ nodes and $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_p$ $(p \geq 2)$ is a partition of $\mathcal{N} \backslash \mathcal{Z}$ into p nonempty subsets, the inequality

$$\frac{1}{2} \sum_{i=1}^{p} y(\delta_{\mathcal{G}-\mathcal{Z}}(\mathcal{S}_i)) \geq p - 1 \tag{7.22}$$

is valid for the polytope of k-node-connected networks. These inequalities are called *node-partition inequalities*, and can be seen as a generalization of node cut inequalities (7.20).

When $r(s) = 2$ for all $s \in \mathcal{N}$, partition inequalities can be generalized further: consider again a patition $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_p$ $(p \geq 2)$ of $\mathcal{N}$ and let $\mathcal{F} \subseteq \delta(\mathcal{S}_1)$ with $|\mathcal{F}|$ odd. The *$\mathcal{F}$-partition inequality* is defined as

$$\frac{1}{2}\sum_{i=1}^{p} y(\delta(\mathcal{S}_i)) - y(\mathcal{F}) \geq p - \left\lceil \frac{|\mathcal{F}|}{2} \right\rceil. \tag{7.23}$$

To show it is valid, simply add valid inequalities

$$\begin{aligned} y(\delta(\mathcal{S}_i)) &\geq 2 \quad \forall\, i = 2, \ldots, p, \\ -y_e &\geq -1 \quad \forall e \in \mathcal{F}, \\ y_e &\geq 0 \quad \forall e \in \delta(\mathcal{S}_1) \setminus \mathcal{F}, \end{aligned}$$

divide by 2 and round up to obtain (7.23).

The formulation above is in the space of design variables only, and involves an exponential number of constraints. As in Sect. 3, it is also possible to obtain valid polynomial-size models by the introduction of flow variables. We do it here only for edge-connectivity requirements. Assuming that we want to construct a network in which there are $r(s,t)$ edge-disjoint paths between nodes s and t, we can define a set $\mathcal{K}$ of commodities where we have one commodity for each node pair s, t such that $r(s,t) > 0$, with a flow requirement of $r(s,t)$ where the source and sink of commodity k are arbitrarily chosen between s and t.

$$\text{Minimize} \sum_{e \in \mathcal{E}} c_e y_e \tag{7.24}$$

$$\text{Subject to} \quad \textstyle\sum_{j \in \mathcal{N}_i^+} x_{ij}^k - \sum_{j \in \mathcal{N}_i^-} x_{ji}^k = w_i^k, \quad \forall\, i \in \mathcal{N}, \forall\, k \in \mathcal{K}, \tag{7.25}$$

$$x_{ij}^k + x_{ji}^k \leq y_e, \qquad \forall\, e = \{i,j\} \in \mathcal{E}, \ \forall\, k \in \mathcal{K}, \tag{7.26}$$

$$x_{ij}^k \geq 0, \qquad \forall\, (i,j) \in \mathcal{A}, \ \forall\, k \in \mathcal{K}, \tag{7.27}$$

$$y_e \in \{0, 1\}, \qquad \forall\, e \in \mathcal{E}. \tag{7.28}$$

Flow balance constraints (7.25) define $r(s,t)$ paths between s and t for each commodity $k \in \mathcal{K}$ corresponding to the node pair s, t, with

$$w_i^k = \begin{cases} r(s,t), \text{ if } i = s, \\ -r(s,t), \text{ if } i = t, \\ 0, \text{ otherwise}, \end{cases} \tag{7.29}$$

and (7.26) impose that these paths are edge-disjoint.

Applying simple max-flow/min-cut arguments, it is quite easy to see that the linear relaxation of the flow formulation is equivalent to the formulation involving only design variables with an exponential number of constraints.

5 Hop Constraints

The models from the previous section usually lead to designs that are very sparse. In general, the survivability constraints alone may not be sufficient to guarantee a cost effective routing with a good quality of service. The reason for this is that the routing paths may be too long, leading to unacceptable delays. Since in most of the routing technologies, delay is caused at the nodes, it is usual to measure the delay in a path in terms of its number of intermediate nodes, or equivalently, its number of arcs (or hops). Thus, to guarantee the required quality of service, one can impose a limit on the number of arcs in the routing paths.

Given a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ with nonnegative edge costs c_e, $e \in \mathcal{E}$, and a set of node pairs $\mathcal{K}$ (sometimes called commodities), we study the problem of constructing a minimum cost set of edges so that the induced subgraph contains at least K edge-disjoint paths with at most L edges between each pair in $\mathcal{K}$.

In theory, we could formulate the limit on the number of hops in a path in the space of design variables only. This approach was used by several authors (see Sect. 7) but has a major drawback: it leads to formulations with an exponential number of constraints, some of which are really hard to interpret and also difficult to handle numerically (the associated separation problem being NP-hard).

In this chapter, we prefer to use extended formulations, once again based on multi-commodity flow variables, that allow to model the limit on the path length more naturally. Moreover, these formulations imply all the valid inequalities known for formulations in the space of design variables. More precisely, the projection of the extended formulations presented here on the space of design variables strictly contains the polyhedron defined by formulations in the space of design variables presented in the literature so far.

The basic idea is to use a layered representation of graph $\mathcal{G}$ to implicitly force each path to use at most L edges. We model the subproblem associated with each commodity with a directed graph composed of $L+1$ layers as illustrated in Fig. 7.1.

Namely, from the original non-directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, we create a directed layered graph $\mathcal{G}^q = (\mathcal{N}^q, \mathcal{A}^q)$ for each commodity $q \in \mathcal{K}$, where $\mathcal{N}^q = \mathcal{N}_1^q \cup \ldots \cup \mathcal{N}_{L+1}^q$ with $\mathcal{N}_1^q = \{o(q)\}$, $\mathcal{N}_{L+1}^q = \{d(q)\}$ and $\mathcal{N}_l^q = \mathcal{N} \backslash \{o(q)\}$, $l = 2, \ldots, L$. Let v_l^q be the copy of $v \in \mathcal{N}$ in the *l-th* layer of graph $\mathcal{G}^q$. Then, the arc sets are defined by $\mathcal{A}^q = \{(i_l^q, j_{l+1}^q) \mid \{i, j\} \in \mathcal{E}, i_l^q \in \mathcal{N}_l^q, j_{l+1}^q \in \mathcal{N}_{l+1}^q, l \in \{1, \ldots, L\}\} \cup \{d(q)^l, d(q)^{l+1}, l \in \{2, \ldots, L\}\}$, see Fig. 7.1. In the sequel, an (undirected) edge in $\mathcal{E}$ with endpoints i and j is denoted $\{i, j\}$ while a (directed) arc between $i_l^q \in \mathcal{N}_l^q$ and $j_{l+1}^q \in \mathcal{N}_{l+1}^q$ is denoted by (i, j, l) (the commodity q is omitted in the notation as it is often clear from the context).

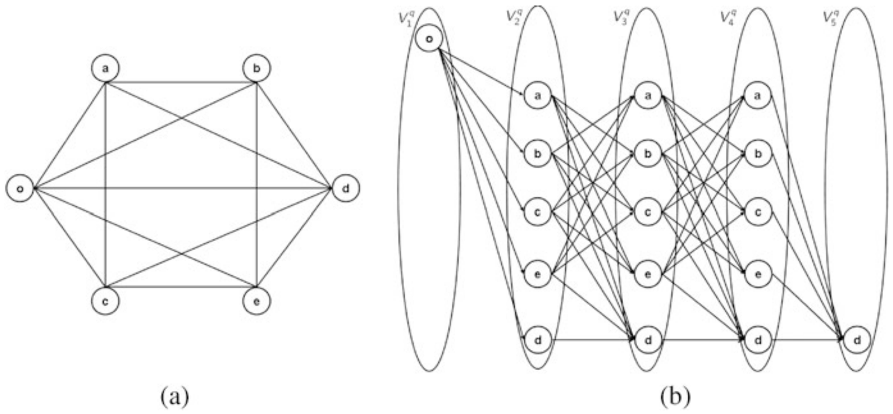


Fig. 7.1 Original network (**a**) and its alternative (or associated) layered representation (**b**) when $L = 4$

Note that each path between $o(q)$ and $d(q)$ in the layered graph G^q is composed of exactly L arcs (hops), which correspond to a maximum of L edges (hops) in the original one. In fact this is the main idea of this transformation, since in the layered graph any path is feasible with respect to the hop-constraints. The usual multi-commodity flow equations defined in this layered graph yield the following model:

$$\text{Minimize} \sum_{e \in \mathcal{E}} c_e y_e \tag{7.30}$$

Subject to

$$\textstyle\sum_{j \in \mathcal{N}_i^+} x_{ij}^{lq} - \sum_{j \in \mathcal{N}_i^-} x_{ji}^{l-1\,q} = w_i^q, \quad \forall i \in \mathcal{N}^q, \ \forall l \in \{2, \ldots, L+1\}, \ \forall q \in \mathcal{K} \tag{7.31}$$

$$\textstyle\sum_{l=1}^{L} \left(x_{ij}^{lq} + x_{ji}^{lq} \right) \leq y_e, \quad \forall e = \{i, j\} \in \mathcal{E}, \ \forall q \in \mathcal{K}, \tag{7.32}$$

$$x_{ij}^{lq} \geq 0, \ integer, \qquad \forall (i, j, l) \in \mathcal{A}^q, \ \forall q \in \mathcal{K}, \tag{7.33}$$

$$y_e \in \{0, 1\}, \qquad \forall e \in \mathcal{E}. \tag{7.34}$$

Flow balance constraints (7.31) define K paths between $o(q)$ and $d(q)$ in the layered graph $\mathcal{G}^q$ with

$$w_i^k = \begin{cases} K, \text{ if } i = o(q), \\ -K, \text{ if } i = d(q), \\ 0, \text{ otherwise.} \end{cases} \tag{7.35}$$

Constraints (7.32) guarantee that paths are edge-disjoint and only use installed edges.

This model can become very large when the number of commodities and the size of the network increase, but it can be solved efficiently using Benders decomposition. It is also interesting to mention that if there is only one commodity, and with non-negative costs, the linear relaxation of this model is always integral when $L \leq 3$.

6 Rings

As stated before, models from Sect. 4 lead to very sparse designs. In fact, it turns out that the optimal solution of the two-connected network problem is often a Hamiltonian cycle. Hence, any edge failure implies that the flow that was routed on that edge must be rerouted, using all the edges of the network, an obviously undesirable feature.

It is therefore necessary to add extra constraints to limit the region of influence of the traffic that it is necessary to reroute if a connection is broken. Hop constraints presented above are a possible way to achieve this. Another approach presented in this section is based on the technology of *self-healing rings*, quite popular in telecommunications networks. Self-healing rings are cycles in the network equipped in such a way that any link failure in the ring is automatically detected and the traffic rerouted by the alternative path in the cycle. It is natural to impose a limited length of these rings. This is equivalent to set a bound on the length of the shortest cycle including each edge.

This leads to the problem of designing a minimum cost network with the following constraints:

1. The constructed network contains at least two node-disjoint paths between every pair of nodes (*2-connectivity constraints*),
   and
2. each edge of the network belongs to at least one cycle whose length is bounded by a given constant K (*ring constraints*).

This problem is called the *Two-Connected Network with Bounded rings (2CNBR) problem*. The length of an edge can be unitary (i.e., similar to hop constraints), or can be weighted, to represent for example the physical delay for flowing through a given edge.

A useful tool to analyze feasible solutions of 2CNBR is the *restriction of a graph to bounded rings*. Given a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ and a constant $K > 0$, we define for each subset of edges $\mathcal{F} \subseteq \mathcal{E}$ its restriction to bounded rings $\mathcal{F}_K$ as

$$\mathcal{F}_K = \left\{ e \in \mathcal{F} : \begin{array}{l} e \text{ belongs to at least one cycle} \\ \text{of length less than or equal to } K \text{ in } \mathcal{F} \end{array} \right\}.$$

The subgraph $\mathcal{G}_K = (\mathcal{N}, \mathcal{E}_K)$ is the *restriction of $\mathcal{G}$ to bounded rings*. Note that an edge $e \in \mathcal{E} \backslash \mathcal{E}_K$ will never belong to a feasible solution of 2CNBR.

Further we denote by $\mathcal{Y}$ the set of incidence vectors of subsets $\mathcal{F} \subseteq \mathcal{E}$ such that

1. $\mathcal{F}$ is two-connected,
2. $\mathcal{F} = \mathcal{F}_K$.

Then, the 2CNBR problem consists in

$$\text{Minimize } \sum_{e \in \mathcal{E}} c_e y_e$$

$$\text{Subject to } \ y \in \mathcal{Y}.$$

Checking that $\mathcal{G}_K$ is two-connected, i.e., that $\mathcal{Y}$ is nonempty, can be done in polynomial time. We therefore assume in the remainder of this chapter that there always exists a feasible solution to the problem.

Since all costs $c_e,\ e \in \mathcal{E}$ are assumed to be nonnegative, there always exists an optimal solution of 2CNBR whose induced graph is minimal with respect to inclusion. More precisely, if $\mathcal{F}_K$ is two-connected, as $\mathcal{F} \supseteq \mathcal{F}_K$, $\mathcal{F}$ is also two-connected and the cost of $\mathcal{F}$ is greater than or equal to the cost of $\mathcal{F}_K$. We can thus relax the constraints and just require that $\mathcal{F}_K$ is two-connected for a set of edges $\mathcal{F}$ to be feasible. Hence, 2CNBR can be equivalently formulated as

$$\text{Minimize } \sum_{e \in \mathcal{E}} c_e y_e$$

$$\begin{aligned} \text{Subject to } \quad & \mathcal{F}_K^y \text{ is two-connected}, \\ & y_e \in \{0, 1\}, \qquad \forall e \in \mathcal{E}. \end{aligned}$$

In order to formulate the problem using only design variables y, observe that if a subset of edges $\mathcal{F} \subseteq \mathcal{E}$ is such that $(\mathcal{G} - \mathcal{F})_K$ is not two-connected, then $\mathcal{G} - \mathcal{F}$ does not contain a feasible solution. Therefore each feasible solution contains at least one edge from $\mathcal{F}$.

As we are only interested in minimal feasible solutions, this is sufficient to formulate the 2CNBR problem as the following integer linear program:

$$\text{Minimize } \sum_{e \in \mathcal{E}} c_e y_e \tag{7.36}$$

$$\text{Subject to } \quad y(\mathcal{F}) \geq 1, \ \forall \mathcal{F} \subseteq \mathcal{E}, \ (\mathcal{G} - \mathcal{F})_K \text{ is not two-connected}, \tag{7.37}$$

$$y_e \in \{0, 1\}, \ \forall e \in \mathcal{E}. \tag{7.38}$$

Constraints (7.37) are called *subset constraints*.

As feasible solutions of 2CNBR are two-connected graphs, valid inequalities for the design of 2-connected networks are also valid for 2CNBR. In particular, cut constraints (7.19) and node-partition inequalities (7.22) are very important in branch-and-cut strategies to solve the problem. However, for general 2-connected networks, cut constraints are facet-defining under very mild conditions, while in many cases they do not define facets for 2CNBR. By studying conditions for these inequalities to define facets, it is possible to strengthen them.

Given a subset of nodes $\mathcal{S} \subseteq \mathcal{N}$, $\emptyset \neq \mathcal{S} \neq \mathcal{N}$, the cut constraint imposes that there are at least two edges leaving $\mathcal{S}$, i.e.,

$$y(\delta(\mathcal{S})) \geq 2.$$

To characterize which cut constraints define facets, it is useful to know, for any pair of edges $e, f \in \delta(\mathcal{S})$, if there exists a solution whose incidence vector lies in the face $y(\delta(\mathcal{S})) = 2$, i.e., if there exists a feasible solution containing e and f but no other edge of $\delta(\mathcal{S})$). This is the case if and only if

$$C_{e,f} = \mathcal{E}(\mathcal{S}) \cup \mathcal{E}(\mathcal{N} \backslash \mathcal{S}) \cup \{e, f\}$$

is feasible, i.e., if $(C_{e,f})_K$ is two-connected.

A useful tool to represent and analyze the vectors belonging to the face defined by a cut constraint is the *ring-cut graph*: Given a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, a constant $K > 0$, and a subset of nodes $\mathcal{S} \subset \mathcal{N}$, $\emptyset \neq \mathcal{S} \neq \mathcal{N}$, the ring-cut graph $RCG_{\mathcal{S},K} = (\delta(\mathcal{S}), RCE_{\mathcal{S},K})$ induced by $\mathcal{S}$ is the graph defined by associating one node to each edge in $\delta(\mathcal{S})$ and by the set of edges

$$RCE_{\mathcal{S},K} = \left\{\{e, f\} \subseteq \delta(\mathcal{S}) : (C_{e,f})_K \text{ is two-connected}\right\}.$$

With the help of the ring-cut graph, it is possible to characterize which cut constraints are facet-defining. Moreover, the ring-cut graph can be used to derive new valid inequalities for 2CNBR. If $\mathcal{F} \subseteq \delta(\mathcal{F})$ is an independent subset in the ring-cut graph $RCG_{\mathcal{S},K}$, then

$$y(\mathcal{F}) + 2y(\delta(\mathcal{S}) \backslash \mathcal{F}) \geq 3 \tag{7.39}$$

is a valid inequality for the 2CNBR problem. Inequalities (7.39) are called *ring-cut inequalities*, and are also very useful to strengthen formulations of 2CNBR.

7 Bibliographical Notes

7.1 Connected Networks

The greedy algorithm for the Minimum Spanning Tree problem is due to Kruskal (1956). The tree polytope is a special case of the matroid polytope first described by Edmonds (1971). For a review on polyhedral results for tree related problems, see Magnanti and Wolsey (1995). Another compact formulation for the minimum spanning tree problem (not described here) was proposed by Martin (1991).

Grötschel, Monma and Stoer studied in detail network design problems with connectivity constraints. A survey of their work can be found in Grötschel et al. (1995a) and Stoer (1992). A later survey by Kerivin and Mahjoub (2005) concentrates on polyhedral aspects for some special cases and presents a general branch-and-cut algorithm. They also consider problems with bounded rings and hop-constrained problems.

7.2 Survivable Networks

In their earliest work on the subject, Grötschel and Monma (1990) introduced a general model mixing edge and node survivability requirements. They examined the dimension of the associated polytope and proved facet results for cut and node-cut inequalities.

They also described completely the polytope of the (1-)connected network problem, based on the work of Cornuéjols et al. (1985), by the introduction of partition inequalities. The first separation algorithm for these inequalities was proposed by Cunningham (1985) and requires $|\mathcal{E}|$ min-cut computations. Barahona (1992) reduced this computing time to $|\mathcal{N}|$ min-cut computations. $\mathcal{F}$-partition inequalities were first proposed by Mahjoub (1994).

Low-connectivity constrained network design problems have been introduced by Monma and Shallcross (1989) and Stoer (1992), who introduced most of the terminology and models presented in Sect. 4. For high-connectivity requirements, the reader is referred to Grötschel et al. (1995b) and Stoer (1992).

Directed multi-commodity flow formulations were studied by Magnanti and Raghavan (2005), and they introduced improved directed flow models that are much stronger than the cut formulations for some variants of the problem (in particular when there are many nodes with unitary requirements).

The complexity of the minimum cost two-connected network problem was established by Eswaran and Tarjan (1976). An in-depth survey of Steiner tree problems was made by Winter (1987). The algorithm for solving the Steiner tree problem in polynomial time when either the number of nodes of type 0 or the number of nodes of type 1 is restricted is due to Lawler (1976). At the time of writing, the most recent and efficient exact approaches for some variants of the Steiner tree problem are due to Fischetti et al. (2017).

7.3 Hop Constraints

Hop-constraints were considered by Balakrishnan and Altinkemer (1992) as a means of generating alternative base solutions for a network design problem. Later on, Gouveia (1998) presented a layered network flow reformulation whose linear programming bound proved to be quite tight. This reformulation has, then, been used in several network design problems with hop-constraints (Pirkul and Soni 2003; Gouveia and Magnanti 2003; Gouveia et al. 2003) and even some hop-constrained problems involving survivability considerations (more on this below). It is also interesting to point out that the apparently simple general network design problem with $L = 2$ already contains a complex structure (Dahl and Johannessen 2004) who also conduct a computational study of this variation of the problem.

The K-edge-disjoint L-hop-constrained network design problem was first studied by Huygens et al. (2007) who only consider $L \leq 4$ and $K = 2$. The node-disjoint variant was studied by Gouveia et al. (2006) and later by Gouveia et al. (2008) who consider a more complicated version. A summary of these results is also presented in the survey by Kerivin and Mahjoub (2005). Itaí et al. (1982) and later Bley (2003) study the complexity of this problem for the node-disjoint and the edge-disjoint cases. More recently, Bley and Neto (2010) also studied the approximability of the problem for $L = 3$ and $L = 4$.

Some authors focused on the formulation of some variants of the problem in the space of design variables. For $K = 1$, Dahl (1999) has provided such a formulation and shown that it describes the corresponding convex hull for $L \leq 3$. Later on, Dahl et al. (2004) have shown that finding such a description for $L \geq 4$ would be much more complicated. For $K \geq 2$ the results are even worse. Huygens et al. (2004) have extended Dahl's result for $K = 2$ and $L \leq 3$. For $L \geq 4$, the only interesting result for the moment is the one given by Huygens and Mahjoub (2007) for $L = 4$ and $K = 2$ where a valid formulation has been given. However, in terms of valid inequalities and with the exception of the well known L-path cut inequalities, nothing is known for larger values of L. This may also explain why the only cutting plane method for the more general problem with several sources and several destinations by Huygens et al. (2007) only considers $L \leq 3$.

The extended formulation for the general case presented in Sect. 5 was introduced by Botton et al. (2013) who also proposed an efficient Benders decomposition method to solve it. They also applied this algorithm to a generalisation of the model with so-called reliable edges (Botton et al. 2015). For the single commodity case, Botton et al. (2018) proposed valid inequalities that completely describe the polyhedron of incidence vectors of feasible solutions for $K = 2$ and $L = 3$ (hence for arbitrary costs), extending results by Bendali et al. (2010) for non-negative costs.

A version of the problem where the hop constraint limit is different in the nominal graph and in case of failures was introduced by Gouveia and Leitner (2017) and an efficient branch-and-cut approach for this problem was proposed by Gouveia et al. (2018).

Another application of extended formulations for hop constrained problems in the context of distribution networks was proposed by De Boeck and Fortz (2018), where some preprocessing techniques to reduce the size of the resulting formulations are also discussed.

Recently, Gouveia et al. (2019) published a survey of layered graph approaches for hop constrained problem.

7.4 Rings

The Two-Connected Network with Bounded Rings problem was first studied by Fortz et al. (2000). More polyhedral results can be found in Fortz (2000); Fortz and Labbé (2002). Fortz and Labbé (2004) studied the particular case of unit edge lengths. The edge-connectivity version of the problem was studied by Fortz et al. (2006).

Other network design problems involve the creation of rings of bounded lengths. Goldschmidt et al. (2003) study the problem of connecting subsets of customers to a concentrator through a self-healing ring connected to a backbone ring of concentrators. This problem, called the *SONET Ring Assignment Problem* (SRAP) has the drawback that many instances are infeasible. Carroll et al. (2013) studied a generalization of the SRAP, called the *Ring Spur Assignment Problem* (RSAP). In this problem, the objective is to design a set of bounded disjoint local rings that are interconnected by a backbone ring, like in the SRAP. Since no SRAP solution exist in some real world instances, locations that have no possible physical route due to limitations of geography can be connected to local rings by spurs off the local rings.

8 Conclusions and Perspectives

Topological network design is a very important topic, and is often used as first step of network planning, considering a long-term horizon where demand is not known in advance. Decoupling the decisions on capacity and routing from the physical design is possible because technology in telecommunications make these two sets of decisions independent, as fiber optics cables have almost infinite capacity, but capacity restrictions arise from equipments that are subsequently added to the nodes.

However, when considering topological design, the simplest model that consists of building a connected network (and that can be solved in polynomial time), is not sufficient as survivability and quality of service aspects must be ensured: one must guarantee that redundancy exists in the network to cover different cases of failures, and that paths used for connecting demand nodes are not too long. Unfortunately, as soon as such constraints are added, all problems become NP-hard and more challenging. Therefore, there is also quite a lot of literature (not covered in this chapter) dealing with the development of heuristics for the problems presented here.

Concerning exact methods, problems with survivable requirements only (as those presented in Sect. 4) have been extensively studied from the polyhedral point of view. The problems are usually modeled in the "natural" space of design variables and several classes of facet-inducing valid inequalities have been proposed in the literature, leading to powerful branch-and-cut algorithms able to solve realistic size instances very efficiently.

As soon as constraints on the length of the paths are added, formulating the problems in the space of design variables only becomes much more challenging, and most approaches rely on extended formulations. But, despite the progress of modern solvers, these formulations quickly become intractable because of their large size, and decomposition methods like Lagrangian relaxation, Benders decomposition or branch-and-price are often used.

Interesting open problems involve a better understanding of Benders cuts arising from Benders decomposition methods for these problems in order to develop more efficient separation algorithms, as currently most approaches rely on solving linear programs for the separation. Given that large size instances remain intractable, there is also a large unexplored research direction in matheuristics, i.e., methods using structural knowledge from exact methods to derive effective heuristics and provide a certificate of quality for the solutions obtained. To the best of our knowledge, the state-of-the-art heuristics are meta-heuristics that do not use any knowledge gained from exact method development.

On a more managerial side, it would also be interesting to measure more precisely, on realistic instances, the loss in solution quality due to the separation of topological design and capacity allocation / routing decisions compared to an integrated strategy.

References

Balakrishnan, A., & Altinkemer, K. (1992). Using a hop-constrained model to generate alternative communication network design. *ORSA Journal on Computing, 4*, 192–205.

Barahona, F. (1992). Separating from the dominant of the spanning tree polytope. *Operations Research Letters, 12*, 201–203.

Bendali, F., Diarrassouba, I., Mahjoub, A., & Mailfert, J. (2010). The edge-disjoint 3-hop-constrained paths polytope. *Discrete Optimization, 7*(4), 222–233.

Bley, A. (2003). On the complexity of vertex-disjoint length-restricted path problems. *Computational Complexity, 12*(3–4), 131–149.

Bley, A., & Neto, J. (2010). Approximability of 3- and 4-hop bounded disjoint paths problems. In *Proceedings of IPCO 2010, Lausanne*. *Lecture notes in computer science* (vol. 6080, pp 205–218). Berlin: Springer.

Botton, Q., Fortz, B., & Gouveia, L. (2015). On the hop-constrained survivable network design problem with reliable edges. *Computers & Operations Research, 64*, 159–167.

Botton, Q., Fortz, B., & Gouveia, L. (2018). The 2 edge-disjoint 3-paths polyhedron. *Annals of Telecommunications, 73*(1), 29–36.

Botton, Q., Fortz, B., Gouveia, L., & Poss, M. (2013). Benders decomposition for the hop-constrained survivable network design problem. *INFORMS Journal on Computing, 25*(1), 13–26.

Carroll, P., Fortz, B., Labbé, M., & McGarraghy, S. (2013). A branch-and-cut algorithm for the ring spur assignment problem. *Networks, 61*(2), 89–103.

Cornuéjols, G., Fonlupt, F., & Naddef, D. (1985). The traveling salesman problem on a graph and some related integer polyhedra. *Mathematical Programming, 33*, 1–27.

Cunningham, W. (1985). Optimal attack and reinforcement of a network. *Journal of ACM, 32*, 549–561.

Dahl, G. (1999). Notes on polyhedra associated with hop-constrained walk polytopes. *Operations Research Letters, 25*, 97–100.

Dahl, G., & Johannessen, B. (2004). The 2-path network problem. *Networks, 43*, 190–199.

Dahl, G., Foldnes, N., & Gouveia, L. (2004). A note on hop-constrained walk polytopes. *Operations Research Letters, 32*(4), 345–349.

De Boeck, J., & Fortz, B. (2018). Extended formulation for hop constrained distribution network configuration problems. *European Journal of Operational Research, 265*(2), 488–502.

Edmonds, J. (1971). Matroids and the greedy algorithm. *Mathematical Programming, 1*(1), 127–136.

Eswaran, K., & Tarjan, R. (1976). Augmentation problems. *SIAM Journal on Computing, 5*, 653–665.

Fischetti, M., Leitner, M., Ljubić, I., Luipersbeck, M., Monaci, M., Resch, M., et al. (2017). Thinning out steiner trees: a node-based model for uniform edge costs. *Mathematical Programming Computation, 9*(2), 203–229.

Fortz, B. (2000). *Design of survivable networks with bounded rings, network theory and applications* (vol 2). Dordrecht: Kluwer Academic Publishers.

Fortz, B., & Labbé, M. (2002). Polyhedral results for two-connected networks with bounded rings. *Mathematical Programming, 93*(1), 27–54.

Fortz, B., & Labbé, M. (2004). Two-connected networks with rings of bounded cardinality. *Computational Optimization and Applications, 27*(2), 123–148.

Fortz, B., Labbé, M., & Maffioli, F. (2000). Solving the two-connected network with bounded meshes problem. *Operations Research, 48*(6), 866–877.

Fortz, B., Mahjoub, A., Mc Cormick, S., & Pesneau, P. (2006). Two-edge connected subgraphs with bounded rings: polyhedral results and branch-and-cut. *Mathematical Programming, 105*, 85–111.

Goldschmidt, O., Laugier, A., & Olinick, E. V. (2003). SONET/SDH ring assignment with capacity constraints. *Discrete Applied Mathematics, 129*, 99–128.

Gouveia, L. (1998). Using variable redefinition for computing lower bounds for minimum spanning and steiner trees with hop constraints. *INFORMS Journal on Computing, 10*, 180–188.

Gouveia, L., Joyce-Moniz, M., & Leitner, M. (2018). Branch-and-cut methods for the network design problem with vulnerability constraints. *Computers & Operations Research, 91*, 190–208.

Gouveia, L., & Leitner, M. (2017). Design of survivable networks with vulnerability constraints. *European Journal of Operational Research, 258*(1), 89–103.

Gouveia, L., Leitner, M., & Ruthmair, M. (2019). Layered graph approaches for combinatorial optimization problems. *Computers & Operations Research, 102*, 22–38.

Gouveia, L., & Magnanti, T. L. (2003). Network flow models for designing diameter-constrained minimum-spanning and Steiner trees. *Networks, 41*(3), 159–173.

Gouveia, L., Patrício, P., & Sousa, A. (2006). Compact models for hop-constrained node survivable network design. In *Telecommunications planning: Innovations in pricing. Network design and management* (pp. 167–180). New York: Springer.

Gouveia, L., Patrício, P., & Sousa, A. (2008). Hop-contrained node survivable network design: An application to MPLS over WDM. *Networks and Spatial Economics, 8*(1), 3–21.

Gouveia, L. E., Patrício, P., de Sousa, A., & Valadas, R. (2003). MPLS over WDM network design with packet level QoS constraints based on ILP Models. In *Proceedings of IEEE INFOCOM* (pp. 576–586).

Grötschel, M., & Monma, C. (1990). Integer polyhedra arising from certain design problems with connectivity constraints. *SIAM Journal on Discrete Mathematics, 3*, 502–523.

Grötschel, M., Monma, C., & Stoer, M. (1995a). *Design of survivable networks*. *Handbooks in OR/MS, vol 7 on Network models* (chap 10, pp. 617–672). Amsterdam: North-Holland.

Grötschel, M., Monma, C., & Stoer, M. (1995b). Polyhedral and computational investigations for designing communication networks with high survivability requirements. *Operations Research, 43*(6), 1012–1024.

Huygens, D., Labbé, M., Mahjoub, A. R., & Pesneau, P. (2007). The two-edge connected hop-constrained network design problem: Valid inequalities and branch-and-cut. *Networks, 49*(1), 116–133.

Huygens, D., & Mahjoub, A. R. (2007). Integer programming formulations for the two 4-hop-constrained paths problem. *Networks, 49*(2), 135–144.

Huygens, D., Mahjoub, A, & Pesneau, P. (2004). Two edge-disjoint hop-constrained paths and polyhedra. *SIAM Journal on Discrete Mathematics, 18*(2), 287–312.

Itaí, A., Perl, Y., & Shiloach, Y. (1982). The complexity of finding maximum disjoint paths with length constraints. *Networks, 2*, 277–286.

Kerivin, H., & Mahjoub, A. R. (2005). Design of survivable networks: A survey. *Networks, 46*(1), 1–21.

Kruskal, J. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society, 7*, 48–50.

Lawler, E. (1976). *Combinatorial optimization: Networks and matroids*. New-York: Holt, Rinehart and Wilson.

Magnanti, T. L., & Raghavan, S. (2005). Strong formulations for network design problems with connectivity requirements. *Networks, 45*(2), 61–79.

Magnanti, T. L., & Wolsey, L. A. (1995). Optimal trees. *Handbooks in Operations Research and Management Science, 7*, 503–615.

Mahjoub, A. (1994). Two-edge connected spanning subgraphs and polyhedra. *Mathematical Programming, 64*, 199–208.

Martin, R. K. (1991). Using separation algorithms to generate mixed integer model reformulations. *Operations Research Letters, 10*(3), 119–128.

Menger, K. (1927). Zur allgemeinen kurventheorie. *Fundamenta Mathematicae, 10*, 96–115.

Monma, C., & Shallcross, D. (1989). Methods for designing communications networks with certain two-connected survivability constraints. *Operations Research, 37*(4), 531–541.

Pirkul, H., & Soni, S. (2003). New formulations and solution procedures for the hop constrained network design problem. *European Journal of Operational Research, 148*, 126–140.

Stoer, M. (1992). *Design of survivable networks*. *Lecture Notes in Mathematics* (vol. 1531). Berlin: Springer.

Winter, P. (1987). Steiner problems in networks: A survey. *Networks, 17*, 129–167.

Chapter 8
Network Design with Routing Requirements

Anantaram Balakrishnan, Thomas L. Magnanti, Prakash Mirchandani, and Richard T. Wong

In Memory of Randy Magnanti

1 Introduction

The topological design and configuration of a network determines its service capabilities to transport flows of material, energy, or information effectively. These capabilities include the network's ability to route origin-to-destination flows on paths that meet performance requirements such as maximum permitted route length, time, transshipments, or likelihood of failure. To account for the interdependence between design and routing decisions, optimization models for network design jointly decide the network configuration and flow routes. However, due to economies of scale (e.g., fixed costs) in network design, optimal solutions to a basic network design model that focuses on cost minimization, without explicitly imposing routing constraints, may not meet the service requirements. For instance, when fixed costs are very high, the optimal network configuration will be sparse, implying that the routes for origin-to-destination flows on the chosen network can be long. Similarly, since facilities with higher performance capabilities (e.g., faster transport service) are more expensive, the minimum cost network design may select

A. Balakrishnan (✉)
University of Texas at Austin, Austin, TX, USA
e-mail: anantb@mail.utexas.edu

T. L. Magnanti
Massachusetts Institute of Technology, Cambridge, MA, USA
e-mail: magnanti@mit.edu

P. Mirchandani
University of Pittsburgh, Pittsburgh, PA, USA
e-mail: pmirchan@katz.pitt.edu

R. T. Wong
e-mail: r.t.wong@att.net

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_8

cheaper arcs that have poorer routing performance. So, for application contexts in which providing good or guaranteed end-to-end service performance is important, we must augment the basic network design model by explicitly incorporating the desired performance requirements. These requirements often take the form of constraints on the routes chosen for each origin-to-destination flow. The goals of this chapter are to explore and understand the structure of network design problems with additional route performance requirements, and discuss tailored algorithms to effectively solve the problem. For this purpose, we focus on a core model that augments the uncapacitated multicommodity, fixed-charge network design model with constraints on the arc flow variables to capture the routing requirements for each origin-destination pair. We refer to this model as the *Network Design with Routing Requirements* (*NDRR*) problem. This NP-hard problem encompasses a broad spectrum of models including the well-known Budget-constrained shortest path and Hop-constrained network design problems. We discuss modeling and theoretical issues as well as algorithmic strategies for the NDRR problem (including valid inequalities and decomposition methods), and relate these issues to prior work on special cases such as constrained shortest path and hop-constrained network design problems that can also arise as subproblems of the general NDRR problem. The simpler and special structure of these problems makes them more amenable to theoretical analysis, and have led to tailored solution techniques. We discuss opportunities to extend these results and methods to the NDRR problem and its capacitated variant. Next, we outline some practical application contexts for the NDRR model.

The NDRR model and its variants apply to transportation, telecommunication, electricity distribution, and other network-based service contexts. We briefly outline selected applications particularly in the *transportation* sector where route performance requirements can take various forms for different modes of freight or passenger transport.

- The *vehicle routing* problem with delivery deadlines (e.g., Desaulniers et al. 2014; Vidal et al. 2013) requires finding minimum-distance vehicle routes to deliver products from depots to geographically dispersed customers, with each customer requiring delivery by a specified time. We can view this problem as a NDRR problem (with additional constraints to ensure that the routes are cyclic) in which each required delivery corresponds to a commodity to be dispatched from a depot to a customer location; selecting an arc from i to j corresponds to routing a truck (carrying orders for multiple customers) between these two locations at a fixed cost equal to the distance from i to j. The delivery deadline for each customer imposes an upper limit on the time that the truck assigned to this customer takes to reach the customer location after departing from the depot, including the time for intermediate deliveries. Thus, the route performance requirement for each customer is the total time for the depot-to-customer route.
- *Package* and *less-than-truckload* carriers need to decide when to dispatch trailers (loaded or empty) between various hub or transshipment locations, and how to route shipments on the chosen services (e.g., Malandraki et al. 2001; Estrada

and Robuse 2009). Chapter 12 of this volume on Service Network Design and Chap. 14 on Motor Carrier Network Design elaborate on the optimization models that arise in this context. Moving trailers between any pair of hubs incurs fixed costs that depend both on distance and the frequency of these services. More frequent movements increase cost, but reduce the waiting (at hub) and/or total travel time for shipments. Shipments may have different priorities, with some requiring time-definite deliveries and others having less stringent requirements. This service design problem can be viewed as a NDRR problem defined on a time-space network whose arcs represent trailer movements and connections (including waiting) at hubs, with the additional requirement that shipments must be routed within their guaranteed origin-to-destination transit times.

- *Airline crew scheduling* (e.g., Gopalakrishnan and Johnson 2005) entails deciding the pairing or duty cycle for each crew member to ensure that each scheduled flight has the required complement of crew members while satisfying crew work rules. The cost of each duty cycle depends on the crew member's assigned flight legs, deadheads, and layovers at intermediate locations. Federal regulations and union rules limit the total duration and possibly the number of layovers in each duty cycle. In the NDRR framework, the commodities are crew members, and the routes correspond to deciding the sequence of flight legs for each duty cycle such that the cycle duration (and number of layovers) does not exceed the permitted value.
- *Service design* for *freight railroads* (e.g., Zhu et al. 2014; see also Chap. 13) requires, as one of its components, designing an effective blocking network. This problem entails selecting a limited number of blocks (sequences or paths of train-service legs, which can be viewed as logical links between yards, on which groups of railcars travel together) for routing shipments (e.g., Barnhart et al. 2000; Ahuja et al. 2007). To limit the number of times railcars are reclassified, i.e., moved from one block to the next block at an intermediate yard, during their origin-to-destination trip, the number of arcs in each shipment's trip plan must not exceed a pre-specified upper limit (that can vary by shipment).
- *Liner container shipping* companies must decide the cyclic routes for their ships, the frequency of service on each route, and the movement of containers between origins and destinations on the chosen services (e.g., Agarwal and Ergun 2008; see also Chap. 15). The route for each container consists of a sequence of sailing legs on different services, with transshipment from one service to the other at intermediate ports. Transshipments are expensive due to cargo damage or loss, handling, and storage; they also increase the origin-to-destination transit time because containers have to wait at the intermediate port for the next scheduled service. So, shipping companies seek to design their service network so that they can transport cargo subject to restrictions on the transit time and number of transshipments (e.g., Balakrishnan and Karsten 2017; Karsten et al. 2017).

Analogous applications with route performance requirements are also pervasive in *telecommunications* network planning. The performance specifications, often referred to as *Quality of Service* (QoS) requirements, stem from the need to limit the 'latency' or end-to-end transmission delay for the many vital and time-sensitive traffic flows on telecommunication networks, including voice-over-IP, distributed game playing, transmission of financial information, and emergency communications. The latency depends on the speeds of the links on the path as well as the number and speeds of intermediate routers/switches. Moreover, we must route critical communications over 'reliable' paths having low probability of link failures or packet switching loss. For certain applications such as multicast broadcast networks, QoS considerations impose additional requirements such as limiting the number of links or hops on the paths from the root node (typically, the message source) to every other node (distribution points or destinations).

The problem of deciding the optimal network configuration (or upgrading an existing network) while ensuring adequate routing performance also arises in contexts such as energy distribution (e.g., De Boeck and Fortz 2017) and supply chain networks, and applies to various scheduling problems that we can define over virtual (versus physical) networks. For instance, we can view the parallel machine, non-preemptive scheduling problem where jobs have different release times and deadlines and require sequence-dependent change-over times (or costs) as the problem of identifying the least cost star network (with as many branches as the number of machines) that spans all the job nodes, with restrictions on the maximum time to reach each job node from the root node. Other applications of the NDRR framework include managing feature addition during a product's life-cycle (e.g., Wilhelm et al. 2003) and optimal path configuration for radar avoidance (e.g., Zabarankin et al. 2001).

Given such widespread applications of the NDRR problem, we focus on how to effectively model and solve this problem. Section 2 provides a classification of network design problems with routing requirements, and formulates the core version that we will address in the remainder of the chapter. We also discuss the challenges in solving the problem, and outline some related threads of theoretical research on the problem's difficulty. Section 3 outlines a polyhedral approach for effectively solving the general network design problem with routing requirements that combines problem reduction, model strengthening, and cutting planes. Section 4 addresses two notable special cases of the problem—constrained shortest path and hop-constrained network design problems—that can arise as subproblems of the general problem. We also describe illustrative tailored solution approaches that exploit the special structure of these problems. Section 5 discusses decomposition methods to solve the general problem, including Lagrangian relaxation, column generation, and Benders decomposition. We also discuss how the preceding methods can be extended to the capacitated variant of the problem that imposes arc capacity constraints in addition to routing requirements. Section 6 provides Bibliographical Notes on prior literature related to the discussions in the following sections. Section 7 concludes the paper with a summary of key observations and learnings about the NDRR problem, and some thoughts on future research directions.

2 Problem Classification and Model Formulation

Network design encompasses a vast array of models that differ in their features and assumptions depending on the application context. Section 2.1 briefly outlines a framework to classify network design problems based on their structure and assumptions. Section 2.2 elaborates on the types of additional flow constraints (besides demand, supply, and flow conservation constraints) that routing requirements may impose. Section 2.3 presents the integer programming formulation for the NDRR problem that we study, and Sect. 2.4 provides insight into why the problem is challenging and why even some of its simpler special cases are difficult.

2.1 Model Classification

The two core decisions for network design are: (1) which arcs, from the given set of candidate arcs, to include in the design, and (2) how to route the origin-to-destination flows on the chosen arcs so as to satisfy demand. We refer to the corresponding decision variables as *design variables* and *flow* or *routing variables*. Two types of constraints are common to all network design models: flow conservation constraints on the flow variables (including demand and supply constraints), and forcing constraints to relate the design and routing decisions, i.e., to ensure that flow is only routed on arcs that are included in the design. We can differentiate network design problems along the following four main dimensions, based on the arc and flow characteristics and requirements.

- *Directed* arcs that can carry flows only in the arc's direction versus *Undirected* edges that permit flows in both directions. Generally, network design models (without any additional valid inequalities) over undirected networks tend to have weaker LP lower bounds than those for directed networks (see, for example, Balakrishnan et al. 1989).
- *Multiple commodities*, distinguished by their origins and destinations, costs, and other characteristics, versus a *Single* homogeneous commodity that can be supplied by any source to a destination. With multiple commodities, the origin-destination demand pattern can be arbitrary or have special structure (e.g., single source, single destination, or complete demand between every pair of nodes). For network design, multicommodity formulations can be tighter (e.g., Rardin and Choe 1979; Vanderbeck and Wolsey 2010).
- *Non-bifurcated* flows that must be routed on a single path from each commodity's origin to destination versus *Bifurcated* flows that permit splitting the required flows among multiple origin-to-destination paths. Ensuring that flows are non-bifurcated requires defining binary variables to define the path for each commodity, making the problems more difficult to solve.
- *Additional constraints*: Network design applications in practice may impose additional constraints besides the flow conservation and forcing constraints of

the basic uncapacitated model. These constraints fall into two main categories—configuration constraints and flow restrictions. Configuration or design constraints involve only the design variables, and define the permissible configurations. For instance, some applications require the design to be a tree network (e.g., for multicasting) while others seek a network that is the union of cycles (e.g., container ship routes, fiber optic ring networks). Flow or routing restrictions limit the routing options by imposing constraints on the flow variables. We will discuss these latter constraints in more detail in Sect. 2.2.

Within this framework, the model can accommodate several other variants such as different objective functions (e.g., maximizing profits with the flexibility to selectively meet demands, or minimizing the number of transshipments) and incorporating node attributes (costs, waiting or processing times, other capabilities). In this chapter, we focus on directed, multicommodity problems with non-bifurcated flows together with additional constraints that we discuss next to account for service requirements.

2.2 Routing Requirements

We capture routing and service requirements by constraining the routes on which commodities can flow. We can broadly classify such constraints as inter-commodity constraints or intra-commodity constraints. *Inter-commodity* constraints enforce joint requirements on the flows of multiple commodities. The most common example is the arc capacity constraint to ensure that the total flow of all commodities on an arc does not exceed the arc's capacity. Other examples include situations in which using an arc for one (or more) commodity on an arc necessitates either not routing another commodity (or subset of commodities) or co-routing another commodity on that arc. For instance, in the rail freight industry, policy and technological restrictions prohibit transporting certain combinations of commodities on the same arc. Likewise, in crew scheduling, some organizations favor keeping crew members from different occupations (commodities) together as a team for multiple trips. *Intra-commodity* constraints refer to flow constraints that involve flow variables from a single commodity. Many of the applications discussed in Sect. 1 fall into this category since they impose performance or service requirements on each origin-to-destination flow, which in our model corresponds to an individual commodity. These applications vary in the performance *metric* they use to ensure that the solution meets service requirements. Further, the metric is often additive, i.e., the total value of the metric for a path, which the constraint seeks to limit, is the sum of the metrics of the arcs and nodes on this path. We next list some common metrics.

- *Time*: In transportation applications, the metric for each arc (node) is often the transportation (transshipment) time. Upper bounds on the transit time from origin to destination, which is the sum of traversal times of the arcs and nodes on the route, can stem from delivery deadlines, product perishability, or the need to reduce in-transit inventory.

- *Distance*: Each arc has an associated distance, and operational or service considerations may require selecting origin-to-destination routes whose distance does not exceed a pre-specified value that can vary with the origin-destination pair.
- *Cost*: Arcs and nodes may have associated costs or other financial metrics (different from those in the cost minimization objective function) for using the arcs or for processing at the nodes. Associated constraints, sometimes called *budget* constraints, impose upper limits on the total cost for each origin-to-destination route.
- *Transshipments*: Many contexts require limiting the number of intermediate transshipments on origin-to-destination routes, for instance, to avoid excessive handling and to regulate the effort and time for processing at nodes. By associating a metric whose value is one for each arc, the total value for any origin-to-destination route is the number of *hops*, i.e., number of arcs on this route; the service requirement imposes an upper bound on this value.
- *Reliability*: In contexts where arcs (or nodes) can fail, a natural service requirement is that the reliability of any chosen origin-to-destination route, defined as the likelihood that this route is operational, must exceed a pre-specified threshold value. Defining the performance metric of each arc (node) as the logarithm of the probability that the arc (node) will be operational and assuming that arc (and node) failures are independent, the service requirement imposes a lower bound on the sum of the arc (node) metrics on any route.

Additional constraints on flow variables may also arise due to operational restrictions or policies that govern routing decisions. For instance, we can add constraints to model logical conditions such as the following: if the route contains arcs from a specified subset, it must not include any arc (or must necessarily include every arc) from another subset. These constraints arise in transportation contexts (e.g., shipment routing for different materials on railroads) and also to impose special configuration requirements such as requiring multiple possible routes for each commodity (e.g., Grotschel et al. 1995; Balakrishnan et al. 2009).

In each of the above examples, the requirement (e.g., maximum transit time, maximum permitted number of hops, minimum required reliability) can vary by origin-destination pair. Further, if we classify traffic flows between an origin and destination into multiple types based on their priorities or service characteristics, we can define separate commodities for each flow type, permitting finer-grained differentiation of routing requirements. Finally, we can readily incorporate performance metrics associated with nodes by simply adding the value of each metric corresponding to a node to the metric of each arc that is incident to (or from) that node. Next, we present an integer programming formulation for the network design problem with routing constraints, and discuss some of its special cases.

2.3 Model Formulation

As noted in Sects. 2.1 and 2.2, network design problems with routing restrictions have many different variants. Our main focus in this chapter is to understand the effects on problem structure and solution strategies when we impose performance or service requirements on origin-to-destination routes in network design solutions. Accordingly, we consider the core multicommodity, fixed charge, uncapacitated network design problem, requiring non-bifurcated flows for each commodity, augmented with intra-commodity routing constraints.

We use the following notation to formulate this optimization problem. Let $\mathscr{G} = (\mathscr{N}, \mathscr{A})$ be the directed graph on which the problem is defined. Node set $\mathscr{N}$ consists of n $(= |\mathscr{N}|)$ nodes representing origin, destination, or transshipment nodes, and arc set $\mathscr{A}$ contains arcs that are available for installation and use. Let $\mathscr{K}$ denote the set of commodities. Commodity $k \in \mathscr{K}$ originates at node $O(k)$ and terminates at node $D(k)$. Commodities may be further distinguished by their routing and service requirements, i.e., we can have multiple commodities with the same origin and destination but different routing constraints. Without loss of generality (since the network is uncapacitated), we scale each commodity's demand to one but permit the routing or flow cost to vary by commodity. Define $\mathscr{N}_i^+ = \{j \in \mathscr{N} : (i, j) \in \mathscr{A}\}$ and $\mathscr{N}_i^- = \{j \in \mathscr{N} : (j, i) \in \mathscr{A}\}$ as the subsets of downstream and upstream neighbors for each node i.

The network design problem has two sets of binary decision variables: (1) *design* variable y_{ij}, for each arc $(i, j) \in \mathscr{A}$, that takes the value one if the solution includes arc (i, j) in the design, and is zero otherwise; and, (2) *routing* or flow variable x_{ij}^k, for arc $(i, j) \in \mathscr{A}$ and commodity $k \in \mathscr{K}$ that equals one if the solution routes commodity k on arc (i, j). Let f_{ij} and c_{ij}^k respectively denote the non-negative fixed cost for using arc (i, j) and flow cost for routing commodity k on this arc. We permit imposing m^k different routing constraints for each commodity k, one corresponding to each performance metric of interest (as discussed in Sect. 2.2). For each metric $m = 1, 2, \ldots, m^k$, let q_{ij}^{km} denote the non-negative coefficient or *weight* of the routing variable x_{ij}^k in the m^{th} constraint, and let Q^{km} be the *weight limit*.

Using this notation, we can formulate the *Network Design problem with Routing Requirements* (NDRR) as the following integer program, denoted as model [*NDRR*].

$$\text{Minimize} \sum_{(i,j)\in\mathscr{A}} (f_{ij} y_{ij} + \sum_{k\in\mathscr{K}} c_{ij}^k x_{ij}^k) \tag{8.1}$$

subject to:

$$\textstyle\sum_{j\in\mathscr{N}_i^+} x_{ij}^k - \sum_{j\in\mathscr{N}_i^-} x_{ji}^k = \begin{cases} 1 & \text{if } i = O(k), \\ -1 & \text{if } i = D(k), \\ 0 & \text{otherwise,} \end{cases} \quad \forall\, i \in \mathscr{N}, \tag{8.2}$$

$$x_{ij}^k \le y_{ij}, \qquad \forall\, (i, j) \in \mathscr{A}, k \in \mathscr{K}, \tag{8.3}$$

$$\textstyle\sum_{(i,j)\in\mathscr{A}} q_{ij}^{km} x_{ij}^{k} \leq Q^{km}, \qquad \forall k \in \mathscr{K}, m = 1, 2, \ldots, m^{k}, \tag{8.4}$$

$$x_{ij}^{k} = 0 \text{ or } 1, y_{ij} = 0 \text{ or } 1, \qquad \forall (i, j) \in \mathscr{A}, k \in \mathscr{K}. \tag{8.5}$$

The objective function (8.1) minimizes the total fixed and routing costs. Constraints (8.2) impose *flow conservation* at every node for each commodity. Together with the *integrality* constraints (8.5) on the routing variables x_{ij}^{k}, the flow conservation equations ensure that the flow solution selects a single origin-to-destination route for each commodity k. The *forcing* constraints (8.3) relate the design and routing decisions; they specify that we can route commodity k on an arc (i, j) only if the design includes this arc (after incurring its fixed cost). Although it is possible to represent this condition using fewer 'aggregate' forcing constraints of the form $\sum_{k\in\mathscr{K}} x_{ij}^{k} \leq |\mathscr{K}| y_{ij}$, one for each arc, the disaggregate version (8.3) yields a tighter linear programming (LP) relaxation (see Balakrishnan et al. 1989). The *routing* constraints (8.4) require the total weight of commodity k's route to be less than or equal to the weight limit, for each metric $m = 1, 2, \ldots, m^{k}$. Finally, constraints (8.5) require the design and flow variables to be binary. Note that we have assumed, for notational simplicity, that every commodity can flow on each arc in the set $\mathscr{A}$. If a commodity is prohibited from flowing on certain arcs (e.g., due to operational or technological issues), we can eliminate the corresponding flow variables from the formulation.

The [*NDRR*] model has two interesting special cases that we will study further Sect. 4. First, if the set $\mathscr{K}$ contains just one commodity, then the problem reduces to finding the shortest path that satisfies all the routing constraints for this commodity. This special case, called the *Constrained Shortest Path* (CSP) problem, is interesting both because it has direct applications in a variety of practical settings and because it often arises as a subproblem in decomposition algorithms for the NDRR problem and other models. Section 4.1 discusses its properties and solution algorithms. In the second interesting and relevant special class of problems, which we call *Hop-constrained* problems, there is only one weight metric, with weight $q_{ij}^{k1} = 1$ for every commodity k and arc (i, j), and the weight limit for commodity k is the maximum allowable number of arcs (or hops) on the commodity's origin-to-destination route. We refer to this limit as the *hop limit*. Among such hop-constrained problems, we consider path and tree versions. For the latter version, we are given a root node and seek a minimum spanning tree such that none of the nodes are more than a specified number of hops away from the root node in the chosen tree. This problem, which we call the *Hop-constrained Minimum Spanning Tree* (*HCMST*) problem, can be modeled as a special case of the NDRR problem in which $|\mathscr{K}| = n - 1$, the root node is the common origin node for all commodities, and every other node is a destination. The Hop-constrained Steiner Tree problem generalizes the HCMST problem by requiring only a subset of nodes to be connected to the root node, i.e., commodities are defined only for a subset of non-root nodes. As we noted in Sect. 2.2, hop constraints are common in many practical applications. Section 4.2 discusses properties and solution methods for hop-constrained problems.

We conclude this discussion by noting that model [*NDRR*] is an arc-flow formulation of the NDRR problem in which the flow variables model each commodity's route as a sequence of arcs. As an alternative, we might consider a path-flow formulation in which the (binary) flow variables represent the choice of origin-to-destination path for each commodity. By limiting (a priori) the available paths to those that are feasible, i.e., satisfy all the routing constraints, the model only requires path selection and forcing constraints. The path-flow formulation has the advantage of having a tighter LP relaxation, and hence higher LP lower bounds, than the arc-flow model. However, it also has the significant drawback of requiring an exponential (in the size of the network) number of path-flow variables. One approach for overcoming this drawback is to use column generation to solve the problem; this approach iteratively generates promising paths based on the dual values for the current solution. However, the subproblem to generate columns is a CSP problem, which is itself NP-hard. In Sect. 6, we discuss the column generation approach to solve NDRR problems.

2.4 Challenges in Solving the NDRR Problem

The NDRR problem is challenging to solve (compared to the basic uncapacitated fixed-charge network design problem) due to the added routing restrictions which complicate the problem structure and make it difficult to even find feasible solutions. We next discuss these issues.

Problem Complexity Adding routing requirements to even simple problems can make them computationally difficult and intractable. For instance, the Shortest Path problem can be efficiently solved, but if we add just one routing constraint, the resulting problem, often called the Budget-constrained Shortest Path (BCSP) problem, is NP-hard (see Garey and Johnson 2002). Similarly, although the Minimum Spanning Tree problem is polynomially solvable, if we add hop constraints (to limit the number of arcs on the path from a root node to every other node), we obtain the HCMST problem, which is NP-hard even if the number of hops is limited to two. The result follows using a transformation from the uncapacitated facility location problem (Dahl 1998). Since the BCSP and the HCMST problems are both special cases of network design with routing requirements, the NDRR problem is also NP-hard.

Multiple Routing Requirements If a commodity has more than one routing requirement, then even finding a feasible solution is NP-hard (Balakrishnan et al. 2020, Grandoni et al. 2014). The four-node example shown in Fig. 8.1, with two routing requirements for a commodity, illustrates this issue. In this example, the commodity originates at node 1 and terminates at node 4. The numbers next to each arc show the arc's cost and its two weights, one for each metric, in the two routing constraints. The weight limits for both metrics is five. The minimum cost path from node 1 to node 4 is 1-2-4, but this path does not satisfy either routing

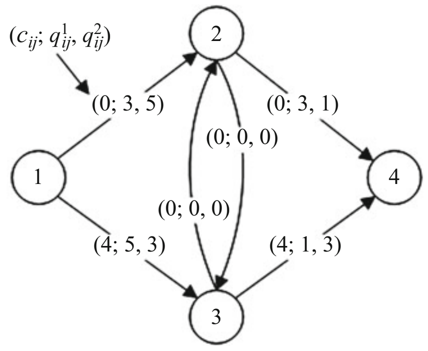


Fig. 8.1 Example with multiple routing requirements

constraint. If the problem contains only one routing constraint, say, only the first constraint, we can readily verify if the problem instance is feasible by finding the shortest "weight" path, using the arc weight for the first metric as the length of each arc. In this example, the shortest weight path for the first metric is 1-2-3-4, with a total weight of 4. Hence, the problem instance is feasible with just the first routing constraint. However, this path does not meet the second routing requirement. Similarly, the shortest weight path 1-3-2-4 using the second set of weights as arc lengths satisfies the second routing requirement, but not the first. The paths 1-2-4 and 1-3-4 satisfy neither routing constraint. So, this problem instance is infeasible if we impose both routing constraints. Observe that we had to examine every origin-to-destination path to determine that the instance is infeasible, suggesting that, with multiple routing requirements, even verifying feasibility is difficult.

Worst-Case Integrality Gap As another indicator of the added problem difficulty when we incorporate a routing constraint, consider again the BCSP problem. We know that the network flow formulation of the unconstrained Shortest Path problem has integer extreme points; so, the LP relaxation of this problem has an integer optimal solution with zero integrality gap. However, when we add a budget constraint, the integrality property no longer holds. That is, the optimal solution to the LP relaxation of the BCSP can be fractional, necessitating the explicit addition of integrality constraints. The fractional LP solution arises because the solution can satisfy the budget constraint 'on average' by routing partial flows on two different paths—one with low cost but high weight and another with higher cost but lower weight. With more than one weight constraint, the LP solution can be a convex combination of more than two paths, none of which satisfy all the weight constraints. These observations suggest that, for general fixed-charge network design problems, the gap between the optimal IP and LP values may be higher when we include routing requirements compared to the gap without these restrictions.

Finally, note that for the CSP problem (and the more general NDRR problem), if the problem imposes only one routing requirement for each commodity (as in the above example), then the original problem is feasible if and only if the LP relaxation is feasible. Moreover, we can construct a feasible solution from the LP solution by routing each commodity over its feasible route on which the LP solution routes a fractional flow, and setting the design variables on all the arcs belonging to these routes to one. However, these observations do not hold when there are two or more routing requirements for a commodity. In this case, the LP relaxation may be feasible even if the original integer program is not. So, we cannot readily construct a feasible IP solution from the LP solution.

3 Solving the NDRR Problem

As our discussions in the previous sections indicate, adding routing requirements to fixed-charge network design models makes them more difficult to solve. Therefore, effectively solving these problems requires exploiting the embedded structure of these problems to reduce problem size, raise lower bounds, and develop specialized solution methods to accelerate performance. We focus on methods that can either solve the problem optimally or provide guarantees of solution quality. We do not consider the many possible meta-heuristic approaches that solve the NDRR problem approximately but do not assure near-optimality. This section describes a cutting plane approach to solve the arc flow formulation of the general NDRR problem based on the recent paper by Balakrishnan et al. (2017), henceforth abbreviated as BLM, which is the only paper to date that addresses the general version of this problem. In Sect. 3.1, we discuss problem reduction methods to eliminate decision variables and also tighten the NDRR problem formulation. Section 3.2 discusses valid inequalities and outlines polyhedral results that underlie a cutting plane approach for the NDRR problem. Computational tests using this method, together with problem reduction and optimization-based heuristics, demonstrated that this approach can significantly reduce computational time compared to applying standard solution procedures.

3.1 Problem Reduction

Problem reduction refers to methods that we can apply a priori (before solving the problem) to fix the values for some decision variables based on feasibility requirements or properties of optimal solutions. For the NDRR problem, these techniques entail either eliminating (i.e., fixing at zero) some commodity routing or design variables, or requiring a commodity to flow through an arc (i.e., fixing the corresponding commodity flow and design variables to one). Such restrictions not only reduce the size of the model formulation (by eliminating variables and constraints), but can also tighten the model, i.e., increase the optimal value of the LP relaxation. We next discuss some intuitive approaches to reduce the NDRR problem. For this discussion, for any commodity $k \in K$ and metric $m = 1, 2, \ldots, m^k$, let L_{ab}^{km} denote the total length of a shortest weight path from node a to node b in the given graph, using the arc weight q_{ij}^{km} as the length of each arc (i, j).

Eliminating Flows To determine if we can eliminate the flow of commodity k on an arc (i, j), we check if, for each metric m, the shortest weight path from $O(k)$ to $D(k)$ containing this arc, consisting of the shortest weight path from $O(k)$ to node i, arc (i, j), and the shortest weight path from node j to $D(k)$ has total weight not exceeding the weight limit Q^{km}. If not, i.e., if:

$$L_{O(k)i}^{km} + q_{ij}^{km} + L_{jD(k)}^{km} > Q^{km} \tag{8.6}$$

for at least one metric $m = 1, 2, \ldots, m^k$, then arc (i, j) does not belong to any feasible origin-to-destination path for commodity k, and so we can eliminate variable x_{ij}^k and the associated forcing constraint (8.3) from the model formulation. Further, if the above test eliminates the flow of commodity k on all arcs incident to (or from) a node i, we can eliminate the flow conservation constraint in (8.2) for this commodity at node i. Finally, if we find that only one commodity k' can flow on an arc (i, j), then we can omit the design variable y_{ij} from the formulation, simply add the fixed cost f_{ij} to the routing cost $c_{ij}^{k'}$ for this commodity, and eliminate the corresponding forcing constraint in (8.3).

Fixing Flows For any arc (i, j) and commodity k, let $L_{ab}^{km}(\mathscr{A}\backslash(i, j))$ denote the total weight (or length) of a shortest weight path from node a to node b after deleting arc (i, j) from the given graph, and using the arc weights for metric m as arc lengths. Then, if

$$L_{O(k)D(k)}^{km}(\mathscr{A}\backslash(i, j)) > Q^{km} \tag{8.7}$$

for at least one metric $m = 1, 2, \ldots, m^k$, i.e., the shortest weight path that does not include arc (i, j) is not feasible for some metric, then commodity k must necessarily flow on arc (i, j). In this case, we can set $x_{ij}^k = y_{ij} = 1$ in the formulation, drop these two variables, and omit all the forcing constraints (8.3) for arc (i, j). Further, since commodity k can only be routed on an elementary path (without cycles) in any optimal solution, we impose the restriction that this commodity cannot flow on any other arc that is incident from node i or to node j. So, we can drop the variables $x_{i'j'}^k$ for all arcs (i', j') with either $i' = i$ or $j' = j$ but not both, and eliminate the forcing constraints for these variables.

Fixing Routes For any commodity k and any feasible (satisfying routing requirements) origin-to-destination path p for this commodity, let $C(p) = \sum_{(i,j)\in p} c_{ij}^k$ and $TC(p) = \sum_{(i,j)\in p}(f_{ij} + c_{ij}^k)$ respectively denote the routing cost and the *total* (fixed plus routing) cost of this path. The total cost essentially assigns the fixed cost of every arc on path p to commodity k. Suppose the problem instance has a unique feasible $O(k)$-to-$D(k)$ path p^* whose total cost $TC(p^*)$ does not exceed the routing $C(p)$ for any other feasible path p for commodity k. Then, the NDRR problem has an optimal solution that routes commodity k on path p^*, implying that we can fix the route for commodity k. To apply this test, we use a CSP algorithm, with commodity k's routing requirements as constraints, to identify path p^* and the other paths. Specifically, p^* is the constrained shortest path using the total cost $(f_{ij} + c_{ij}^k)$ as the length of each arc (i, j). To find the other paths p, we apply the following procedure. For every arc (i, j) in p^*, we delete (i, j) from the network, find the constrained shortest path using routing costs as arc lengths, and choose p' as the least (routing) cost path among these paths. If $TC(p^*) \leq C(p')$, then we can eliminate commodity k (and all its associated variables and constraints) from the problem formulation since this commodity must flow on path p^*; in this case, we fix to one the values of the design variables y_{ij} for all arcs (i, j) on p^*, and omit the forcing constraints for the underlying flow variables.

The above discussion illustrates the two broad approaches to reduce the problem size by eliminating or fixing variables—based on weight feasibility, such as the first two conditions for eliminating or fixing flows, or on cost (optimality) such as the third test for fixing routes. These approaches can greatly improve solution performance for the overall NDRR problem both by reducing model size (variables and constraints) and by raising the LP lower bounds. The effectiveness of these tests depends on the characteristics of the problem instance. The feasibility tests (for eliminating or fixing flows) are likely to be more effective if the weight limits in the routing requirements are somewhat stringent (tight), the network is sparse, and the arcs vary widely in their weights. For instance, for the hop-constrained network design problem (that requires each commodity to flow on a route that uses no more than a prespecified number of arcs or hops), if the hop limits are small and the network is sparse, the test to eliminate flows can be very effective since many arcs may not belong to any low-hop path from the commodity's origin to destination. If, in addition to the hop constraint, the route is also subject to a more general weight constraint and the distribution of arc weights has wide dispersion, then the problem instance may permit further reduction. We conclude this discussion by noting that the first two methods, for eliminating and fixing flow variables, also apply to the CSP problem.

3.2 Valid Inequalities and Composite Algorithm for the NDRR Problem

Polyhedral approaches have proven to be very effective to solve several notoriously difficult integer programming problems, including many variants of network design. The success of these methods rests on using strong problem formulations (i.e., formulations with small gaps between the optimal value of the integer program and its LP relaxation) and on developing tight valid inequalities, preferably facets, that can be added to cutoff fractional solutions. Often, tailored valid inequalities, that exploit insights about the polyhedral structure of the underlying problems, are most effective and yield significant improvements in the LP lower bounds. Previously, researchers have studied the polyhedral structure and developed tighter formulations for certain special cases of the NDRR problem such as hop-constrained network design (see Sect. 4). For the general NDRR problem, we next summarize the work of Balakrishnan et al. (2017) (*BLM*) who developed and successfully applied several classes of valid inequalities to solve the problem's arc flow formulation [*NDRR*].

The LP relaxation of model [*NDRR*] may achieve a much smaller optimal value than the integer program by splitting the flow of a commodity across multiple origin-to-destination paths. This flow splitting occurs for two reasons: (1) by splitting flows, the LP relaxation can select fractional values for the design (y_{ij}) variables, thus only partially absorbing the arc fixed costs; and (2) the LP solution can reduce the routing cost component of its objective value by partially routing a commodity's flow on paths that have low routing costs but high weights since it is only required to meet the route restrictions 'on average' (as illustrated in Sect. 2.3). The first reason applies more generally to other fixed-charge problems, whereas the second reason is specific to the NDRR problem. BLM propose three broad families of inequalities, called *Route Composition*, *Contingent Routing*, and *Multicommodity Design* inequalities, to reduce flow splitting, and prove that some specific versions of these inequalities are facet-defining. The first two inequality classes focus on reducing (partial) flows on infeasible paths, thus addressing the second reason above, whereas the last class addresses both reasons. To motivate these inequalities, provide intuition, and highlight the underlying principles, we discuss a few illustrative versions of these inequalities; BLM provide a more comprehensive treatment.

Route Composition inequalities These commodity-specific inequalities impose the condition that the route for a given commodity must not contain more than a certain number of arcs from a carefully chosen subset of arcs. Since each routing requirement resembles the capacity constraint in a knapsack problem, constraints analogous to the knapsack cover inequality (e.g., Nemhauser and Wolsey 1988) are a natural starting point for NDRR valid inequalities. Specifically, if $\mathscr{A}'$ is a subset of arcs such that $\sum_{(i,j)\in\mathscr{A}'} q_{ij}^{km} > Q^{km}$ for some metric $m = 1, 2, \ldots, m^k$, then no feasible path for commodity k can contain all the arcs in $\mathscr{A}'$, and so the inequality $\sum_{(i,j)\in\mathscr{A}'} x_{ij}^{km} \leq |\mathscr{A}'| - 1$ is valid. However, this inequality is based solely on the arc weights and does not take into account a key requirement that governs the choice of commodity k's routing variables in the NDRR problem, namely, that the set of arcs on which commodity k flows must constitute an elementary path from $O(k)$ to $D(k)$. By exploiting this requirement, we can formulate a cut that is significantly stronger than the basic cover inequality. To illustrate this opportunity, consider the flow of commodity k on two arcs (i_1, j_1) and (i_2, j_2), and suppose the sum of the weights of these two arcs is less than the weight limit for every metric m. So, based solely on their individual weights, we cannot impose the requirement that at most one of these arcs must be selected for commodity k. However, if we can determine that the two arcs cannot simultaneously belong to any feasible $O(k)$-to-$D(k)$ path, then the inequality $x^k_{i_1j_1} + x^k_{i_2j_2} \le 1$ is valid. For this purpose, we note that any path that contains both arcs must either traverse arc (i_1, j_1) first before arc (i_2, j_2), or vice versa. For any metric m, the total weight of the shortest weight origin-to-destination path that contains arc (i_1, j_1) before arc (i_2, j_2) is $L1^{km} = L^{km}_{O(k)i_1} + q^{km}_{i_1j_1} + L^{km}_{j_1i_2} + q^{km}_{i_2j_2} + L^{km}_{j_2D(k)}$, whereas the smallest total weight if the order is reversed is $L2^{km} = L^{km}_{O(k)i_2} + q^{km}_{i_2j_2} + L^{km}_{j_2i_1} + q^{km}_{i_1j_1} + L^{km}_{j_1D(k)}$. The smaller of these two total weights is the weight of the shortest weight $O(k)$-to-$D(k)$ path that contains both arcs. Hence, if $Min\{L1^{km}, L2^{km}\} > Q^{km}$ for some metric m, we cannot route commodity k on any path that contains both arcs. In this situation, we say that $x^k_{i_1j_1}$ and $x^k_{i_2j_2}$ are *incompatible* flows, and the inequality $x^k_{i_1j_1} + x^k_{i_2j_2} \le 1$ is valid. We can further strengthen this inequality by ‘lifting’ it, i.e., by adding some other flow variables to the left-hand side. For instance, if $\mathscr{A}' \in \mathscr{A}$ is a set of arcs such that the flow variables $x^k_{i'_1j'_1}$ and $x^k_{i'_2j'_2}$ for all $(i'_1, j'_1), (i'_2, j'_2) \in \mathscr{A}'$ are pair-wise incompatible with each other, then the inequality $\sum_{(i',j')\in\mathscr{A}'} x^k_{i'j'} \le 1$ is valid.

We can also generalize this inequality. Let $r > 1$ be an integer, and arc set $\mathscr{A}' \subseteq \mathscr{A}$ with $|\mathscr{A}'| \ge r$. If no feasible solution to formulation [*NDRR*] permits commodity k to flow over more that $(r-1)$ arcs of $\mathscr{A}'$, then we say that $\mathscr{A}'$ is r-arc incompatible. Given an r-arc incompatible set $\mathscr{A}'$, let $\mathscr{A}'' \subseteq \mathscr{A} \setminus \mathscr{A}'$. If every arc $(i, j) \in \mathscr{A}''$ is incompatible for commodity k with every arc in $\mathscr{A}' \cup \mathscr{A}''$, then the inequality $\sum_{(i,j)\in\mathscr{A}'} x^k_{ij} + (r-1)\sum_{(i,j)\in\mathscr{A}''} x^k_{ij} \le r-1$ is valid. This discussion illustrates how we can jointly exploit the weight constraints and the origin-to-destination routing requirement to develop tight valid inequalities for the NDRR problem.

Contingent Routing Inequalities This second class of inequalities further extends this principle of combining the weight constraints and the origin-to-destination routing requirement. A version of these inequalities, call *Lifted Turn* constraints, expresses the requirement that if a commodity flows on any arc in one subset, it must also flow on an arc of another subset. Consider a node v, where v is neither the origin nor the destination of commodity k. Let $Out(v) \subseteq \mathscr{N}_v^+$ be a subset of the outgoing arcs at node v. If $In(v) \subseteq \mathscr{N}_v^-$ denotes the maximal subset of incoming arcs (i, v) into node v such that every arc in $In(v)$ is pair-wise incompatible with every arc in $Out(v)$, then the inequality $\sum_{(i,v)\in In(v)} x^k_{iv} \le \sum_{(v,j)\in\mathscr{N}_v^+\setminus Out(v)} x^k_{vj}$ is valid. Note that the pair-wise incompatibility between arcs in $In(v)$ and arcs in $Out(v)$ arises because, for some m, $L1^{km} = L^{km}_{O(k)i} + q^{km}_{iv} + q^{km}_{vj} + L^{km}_{jD(k)} > Q^{km}$, for all $(i, v) \in In(v)$ and $(v, j) \in Out(v)$. Given the arc set $Out(v)$, we can identify $In(v)$ in the following way. Let $L^{km}_{min}(Out(v)) = \min_{(v,j)\in Out(v)}(q^{km}_{vj} + L^{km}_{jD(k)})$ and $L^{km}_{max}(\mathscr{N}_v^+ \setminus Out(v)) = \max_{(v,j)\in\mathscr{N}_v^+\setminus Out(v)}(q^{km}_{vj} + L^{km}_{jD(k)})$. An arc (i, v) belongs to $In(v)$ if and only if $L^{km}_{O(k)i} + q^{km}_{iv} + L^{km}_{max}(\mathscr{N}_v^+ \setminus Out(v)) \le Q^{km}$ and $L^{km}_{O(k)i} + q^{km}_{iv} + L^{km}_{min}(Out(v)) > Q^{km}$. BLM discuss an alternate way of identifying the sets $In(v)$ and $Out(v)$, and show that the Lifted Turn inequality is facet defining under mild conditions. They also discuss a generalization of this inequality obtained by considering arcs that are incident to and from a subgraph spanning multiple nodes instead of the single node v.

Multicommodity Design Inequalities Both the previous two classes of inequalities restrict the flows of a single commodity and do not involve the design variables. The Multicommodity Design (*MCD*) inequalities impose variable upper bounds (that depend on the design variables) on the sum of flows of multiple commodities on various arcs. This very general class of inequalities is particularly effective in eliminating fractional LP solutions to [*NDRR*] because it relates the design and flow variables across multiple commodities and arcs. We first define two underlying commodity-specific relationships, called OR and IF relationships, that stem respectively from the Route Composition and Contingent Routing inequalities (both of which focus on a single commodity):

- an $OR(k, \mathscr{A}', \lambda)$ relationship specifies that no feasible path for commodity k can use more than λ arcs from a set $\mathscr{A}'$; and,
- an $IF(k, \mathscr{A}', \mathscr{A}'')$ relationship specifies that if commodity k flows on an arc of $\mathscr{A}' \subset \mathscr{A}$, then it must also flow on an arc of $\mathscr{A}'' \subseteq \mathscr{A} \setminus \mathscr{A}'$.

These two types of inequalities permit us to develop the following broad class of MCD inequalities. Let $\Omega = R_1, R_2, \ldots, R_Q$ denote Q relationships such that relationship q is either an $OR(k, \mathscr{A}', \lambda)$ relationship or an $IF(k, \mathscr{A}', \mathscr{A}'')$ relationship. Let I_{OR} and I_{IF} denote the subsets of indices q corresponding to the OR and IF relationships in the set Ω, and suppose δ_{ij} is an even number of relationships in Ω that involve arc $(i, j) \in \mathscr{A}$. Adding the inequalities in Ω to the forcing constraints $x_{ij}^{k_q} \leq y_{ij}$, and rounding down the resulting right hand side gives the following inequality for model [*NDRR*]:

$$\sum_{q \in I_{OR}} \sum_{(i,j) \in \mathscr{A}_q'} x_{ij}^{k_q} + \sum_{q \in I_{IF}} \sum_{(i,j) \in \mathscr{A}_q'} x_{ij}^{k_q} \leq \sum_{(i,j) \in \mathscr{A}} \frac{\delta_{ij} y_{ij}}{2} + \lfloor \sum_{q \in I_{OR}} \frac{\lambda_q}{2} \rfloor. \tag{8.8}$$

This MCD inequality tightens the [*NDRR*] if $\sum_{q \in I_{OR}} \lambda_q$ is odd. When the subset I_{IF} is empty, the MCD inequality has only OR relationships on the left hand side, and is facet defining under relatively mild conditions.

Composite Algorithm We can solve the NDRR problem by developing a tailored approach that uses the inequalities discussed above in a cutting plane approach, blended with an optimization-based heuristic. Since the number of inequalities in each of the classes is exponential in the input size, BLM use heuristics to identify inequalities violated by the LP solution. After solving the LP relaxation of this strong model, they use a LP-based heuristic to identify a feasible solution that fixes and releases variable values, an approach that is fast and effective for generating near-optimal solutions for large-scale instances. This method yields solutions that are within 1% of optimality, significantly outperforming (both in terms of solution time and solution quality at termination) a standard branch-and-bound procedure (with built-in general cutting planes) that attempts to solve the base NDRR model without model strengthening and problem reduction.

3.3 Extension to Capacitated Network Design with Routing Restrictions

The approach discussed thus far can be extended to the Capacitated Network Design problem with Routing Restrictions (CNDRR) that not only imposes the routing constraints with non-bifurcated flow but also incorporates the following arc capacity constraints. If d^k denotes the demand for commodity k and u_{ij} is the capacity of arc (i, j), then we simply add the constraints $\sum_{k \in \mathscr{K}} d^k x^k_{ij} \leq u_{ij} y_{ij}$ for all arcs (i, j), to the [*NDRR*] formulation to model the CNDRR problem. Observe that, in this constraint although it suffices to use just the capacity u_{ij} as the right-hand side value, multiplying this value with the design variable y_{ij} strengthens the formulation. The CNDRR problem is more difficult because its LP relaxation can have fractional flows even when all the design variables are integer-valued and all the fractional flow paths for each commodity satisfy its routing requirements.

Researchers have developed various families of valid inequalities for the Capacitated Network Design (CND) problem (without routing restrictions) or its variant, the network loading problem (with discrete and modular capacities), to tighten the model. Such inequalities include the cutset, flow-cutset, partition, residual capacity, and c-strong inequalities. These inequalities remain valid even for the CNDRR problem, and so we can add them to the CNDRR formulation to strengthen it. Conversely, we can add the weight constraints and our NDRR valid inequalities to the formulation of the capacitated network design problem without routing constraints.

Interestingly, our commodity routing constraint and the arc capacity constraint are analogous but ‘orthogonal’ in the following sense. The routing constraint imposes an upper limit on the total weight of all the arcs on which a given commodity flows, whereas the arc capacity constraint limits the total flow of all the commodities that use a given arc. Conceptually, we can think of the routing constraint as a ‘longitudinal’ requirement along a commodity’s path, whereas the capacity constraint is a ‘lateral’ requirement across commodities for an arc. We can exploit this complementary nature of the two requirements to tighten the valid inequalities for the routing requirements (or to eliminate variables) based on the capacity constraints and vice versa. We provide below some examples of such ‘integration’ of the two requirements to strengthen the CNDRR model.

- Suppose the demand d^k for a commodity k exceeds the capacity of an arc (i, j). In this case, not only can we omit the variable x_{ij}^k (since commodity k cannot flow on arc (i, j)), but also delete this arc from the network when computing the shortest weight paths needed to eliminate or fix flows of commodity k (see NDRR problem reduction methods in Sect. 3.1). Moreover, eliminating the arc flow variable x_{ij}^k can tighten both the Route Composition and Contingent Routing inequalities discussed in Sect. 3.2. For instance, for the Lifted Turn inequalities, although arc (i, j) may be compatible (from the perspective of the routing constraints) with one or more arcs in the set $In(v)$ incident to node $v = i$, we can omit this variable from the right-hand side of the inequality, thereby strengthening it. Conversely, if during problem reduction based on routing constraints, we discover that commodity k cannot flow an arc (i, j) (since the length of the shortest weight path through this arc exceeds a weight limit), then omitting this arc flow from the arc capacity constraint can help tighten any related CND valid inequalities.
- Consider two commodities $k1$ and $k2$ that can individually flow on an arc (i', j'), but cannot simultaneously on this arc due to the arc's capacity constraint, i.e., because $d^{k1} + d^{k2} > u_{i'j'}$. Further, suppose there is an arc (i_1, j_1) (and arc (i_2, j_2)) such that, if $k1$ (respectively, $k2$) flows on this arc it must necessarily flow on arc (i', j') to meet the weight limits, i.e., the length of the shortest weight path that includes arc (i_1, j_1) (respectively, (i_2, j_2)) but excludes arc (i', j') exceeds the weight limit for one or more metrics for commodity k_1 (respectively, k_2). In this case, either commodity $k1$ can flow on arc (i_1, j_1) or $k2$ can flow on (i_2, j_2), but not both, implying that the inequality $x_{i_1,j_1}^{k1} + x_{i_2,j_2}^{k2} \leq 1$ is valid. We can extend this inequality to subsets of three or more commodities. There are other such opportunities to develop 'integrated' inequalities that are based on jointly considering the arc capacity and routing requirements.
- We obtained the Multicommodity Design inequalities by aggregating judiciously chosen Route Composition and Contingent Routing inequalities, and applying rounding. For the CNDRR problem, we now have additional inequalities based on arc capacity constraints that we can consider for aggregation. For instance, based on the demand for different commodities and the capacity of an arc (i, j), we can impose cover inequalities of the form $\sum_{k \in \mathscr{K}'} x_{ij}^k \leq \lambda$, for an appropriate subset of commodities $\mathscr{K}'$. We can now consider combinations of these inequalities with those obtained using the routing requirements to develop an even richer set of Multicommodity Design inequalities.

In summary, for the CNDRR problem, we can strengthen the basic model by directly adding both our NDRR valid inequalities and cuts developed for capacitated network design problems. However, there are many opportunities to further reduce problem size and develop integrated inequalities based on the joint consideration of routing and capacity constraints.

4 NDRR Special Cases: Constrained Shortest Paths and Hop-Constrained Problems

Unlike the general NDRR problem, two special cases—the Constrained Shortest Path (CSP) and Hop-constrained Tree problems—have been well-studied in the literature. This section briefly reviews salient results and methods for these two special cases since the proposed modeling and solution strategies for these problems may prove useful for solving the broader NDRR problem. For instance, the CSP problem arises as a subproblem when solving the NDRR problem using column generation. The discussion also serves to illustrate approaches to develop and analyze approximation algorithms for the special cases, possibly pointing to principles that may extend to the general NDRR problem (for which no such analysis currently exists). Finally, for certain special cases (e.g., some hop-constrained problems with low hop limits), researchers have fully characterized the convex hull of feasible solutions. These results together with a hop-constrained problem formulation based on layered networks may provide the foundation to develop tighter (extended) NDRR problem formulations. Section 6 on Bibliographical Notes outlines the literature related to the topics discussed in this section and the next.

4.1 Constrained Shortest Path (CSP) Problem

The CSP problem is a single-commodity version of the NDRR problem that requires identifying the least expensive path from a given origin node s to a destination t whose total weight for each metric m does not exceed the corresponding weight limit. The literature sometimes refers to the problem containing only one routing constraint (one metric) as the *Budget-constrained Shortest Path* (BCSP) problem. To distinguish this problem from the more general version, we refer to the problem with a single commodity but multiple metrics and constraints as the *Weight-constrained Shortest Path* or *WCSP problem*. For these special cases, we can simplify the NDRR problem formulation as follows. Since there is only one commodity, we can omit the commodity index on the flow variables and weight limits. For the BCSP problem, since there is only one metric, we also omit the index m. Moreover, with positive costs, $y_{ij} = 1$ if and only if $x_{ij} = 1$. So, we can omit the design variables y_{ij} and forcing constraints (8.3), and use $(f_{ij} + c_{ij})$ as the flow cost of each arc (i, j) in the objective function. As noted previously, the CSP problem is NP-hard. We next discuss some theoretical results on approximation algorithms for the CSP problem, and later outline two interesting solution approaches that are effective in practice.

4.1.1 Approximation Schemes for the CSP Problem

For NP-hard problems such as the CSP problem, there are approximation (heuristic) algorithms that have provable bounds on solution quality. To facilitate the analysis of their performance, these algorithms are often simple and run in polynomial time. (For more complicated schemes such as neighborhood search, it is often not possible to characterize worst-case performance or even computational complexity.) Given any input or problem instance, an approximation scheme generates a feasible solution whose value is guaranteed (a priori) to be within a pre-specified (worst-case) factor of the optimal solution value. For minimization problems, this guarantee is expressed in terms of the maximum possible ratio of the cost of the approximate solution to the optimal value. Common techniques for obtaining these bounds include methods based on LP relaxation, Lagrangian relaxation, iterative rounding, randomized rounding, primal-dual methods, greedy heuristics, and scaling and rounding. The approach used depends on the problem's underlying structure and solution characteristics. For some problems, researchers have been able to develop desirable bounds that are either a constant factor (e.g., for network design special cases such as the Steiner tree, Traveling Salesman, and Facility Location problems) or depend on the problem dimensions. For instance, Balakrishnan et al. (1996) propose a efficient overlay heuristic for the uncapacitated network design problem and showed that this method yields a solution that is guaranteed to be within a factor of $|\mathscr{K}|$ of the optimal value (this is the first known bound for this problem). In other situations, the running time depends on the desired (maximum) approximation error $\epsilon > 0$. A fully polynomial-time approximation scheme, abbreviated as *FPTAS*, for a minimization problem generates a solution that is guaranteed to be within a factor of $(1 + \epsilon)$ of the optimal solution in running time that is polynomial in $1/\epsilon$ and the size of the input.

Since the specialized approximation algorithms rely on a problem's underlying structure to characterize worst-case performance, even seemingly minor changes to the problem affect the methods' applicability, analysis, and bounds. For instance, for the Knapsack problem, a slight variation of the greedy algorithm that selects items in decreasing order of value-to-weight is easy to analyze; it produces a solution value that is within a factor of $\frac{1}{2}$ of the optimal value. Although the BCSP problem has a knapsack-type constraint, the previous greedy approach is not applicable since the chosen arcs must also form an origin-to-destination path. The predominant method used to develop approximation schemes for the CSP problem is scaling-and-rounding. This approach entails reducing the weights or costs, by scaling and rounding, to low enough values so that the scaled problem can be solved efficiently. Although the scaled problem only yields an approximate solution to the original problem, the method has better time complexity than exact algorithms. The larger the scaling factor, the quicker the method runs but the solutions may be further from optimality. By judiciously selecting the scaling method and using other algorithmic steps (e.g., to determine tight bounds), the algorithm can yield an ϵ-optimal solution in polynomial time.

We next outline a FPTAS for the BCSP problem defined over acyclic graphs to illustrate these ideas. The method is based on a dynamic programming algorithm to solve the BCSP problem. When applied to the original problem (without any cost or weight scaling), this algorithm finds the optimal solution in $O(|\mathscr{A}|Z^*)$ time, where Z^* is the (unknown) optimal value of the problem. We can readily determine a priori upper bounds on Z^* (e.g., $Z^* \leq \sum_{(i,j)\in\mathscr{A}} c_{ij}$ or, better yet, the sum of the $(n-1)$ highest arc costs since no simple path can contain more than $(n-1)$ arcs). But since these bounds depend on the data (e.g., arc costs), the dynamic programming method, applied using the original parameters, is pseudo-polynomial. Now, suppose we can develop lower and upper bounds, LB and UB, on the optimal value such that $UB/LB \leq 2$. Then, for a specified approximation error ϵ, when we apply the dynamic program after scaling and rounding the arc cost coefficients to $d_{ij} = \lfloor c_{ij}/(LB\epsilon/(n-1))\rfloor$, the method runs in polynomial time ($O(|\mathscr{A}|n/\epsilon)$) and generates a solution whose approximation error is at most $\epsilon LB \leq \epsilon Z^*$, i.e., the solution is ϵ-optimal. To achieve the appropriate bounds needed for this approach, we start with $LB = 1$ and $UB =$ sum of the $(n-1)$ highest arc costs, and iteratively apply (in polynomial time) the scaling-and-rounding method to reduce the UB and raise the LB until $UB/LB \leq 2$. The method also extends to BCSP problems over general graphs. As this discussion illustrates, developing a FPTAS requires innovative approaches and insights about the problem structure and how to exploit its properties, with a focus on both characterizing the approximation error and reducing computational effort.

Unlike the BCSP problem, fewer approximation results are known for the more general WCSP problem with two or more weight constraints. Approximation algorithms are also available for a variant of the WCSP problem that allows bounded violation of the weight constraints. That is, in addition to approximating the objective function value to within a factor of $(1+\epsilon)$ (for minimization problems) these methods also permit relaxing (approximating) the weight constraints. When the weight limits can be exceeded by the same factor $(1+\epsilon)$, the approximation algorithm for the WCSP problem essentially seeks an appropriate solution(s) to a multiobjective shortest path problem having costs and routing metrics as different criteria.

4.1.2 CSP Solution Algorithms

We next discuss two interesting solution methods for the CSP problem (with non-negative arc costs) that exploit its special structure. These methods, although not polynomial, are effective in practice.

4.1.3 Handler and Zang's Algorithm

For the BCSP problem, Handler and Zang (1980), abbreviated as HZ, consider the Lagrangian relaxation obtained by dualizing the single weight constraint with multiplier u, and proposed a novel solution approach to solve the Lagrangian dual and close the optimality gap. For this scheme, the Lagrangian subproblem:

$$L(u) = \min \sum_{(i,j)\in\mathscr{A}} (c_{ij} + uq_{ij})x_{ij} - uQ, \qquad \text{subject to (8.2) and (8.5),}$$

is a shortest path problem using arc lengths $(c_{ij} + uq_{ij})$; this path's length, when reduced by uQ, represents the optimal value $L(u)$ of the Lagrangian subproblem. For any $u \geq 0$, $L(u)$ is a lower bound on the optimal value of the original problem. We can solve the Lagrangian dual problem, maximize $\{L(u){:}u \geq 0\}$, by iteratively adjusting u using, for instance, a general technique such as sub-gradient optimization or a more specialized approach. HZ propose a tailored method that exploits the BCSP problem's special structure (and the fact that we need to optimize just one dual multiplier) to solve the Lagrangian dual and reduce any remaining duality gap. The Lagrangian value $L(u)$ is a piecewise linear, concave function of u, with each segment of the piecewise function corresponding to the Lagrangian value for one origin-to-destination path. Starting with two paths (one which minimizes cost without the budget constraint and one which minimizes budget usage), the dual solution method iteratively refines an upper (piecewise linear) approximation for the $L(u)$ function by sequentially generating s-to-t paths and updating the multiplier u. The method monotonically increases the Lagrangian lower bound, denoted as LB. If at any iteration, the Lagrangian solution is feasible (i.e., satisfies the budget constraint), we can also update the upper bound UB if the cost of this new path is lower than the current best upper bound. When the method terminates, we may still have a duality gap, i.e., LB may be less than UB. The following approach closes this gap. For the final value of the dual multiplier u, instead of finding just the shortest path (as we do to solve the Lagrangian subproblem), suppose we sequentially identify the rth shortest path (using the Lagrangian costs), for increasing values of r. Let $L_r(u)$ be the (Lagrangian) cost of the rth shortest path. We update LB as $L_r(u)$, and can possibly update UB if the rth shortest path is feasible for the BCSP problem. We increment r and repeat the process until LB equals or is sufficiently close to UB. Since the network contains only a finite number of (elementary) origin-to-destination paths, this gap reduction procedure will terminate in a finite number of iterations.

Node Labeling Approach

Another approach to solve the BCSP problem is by generalizing Dijkstra's label-setting shortest path algorithm. The generalization entails associating multiple labels with each node i, one for each sub-path from node s to i that can potentially belong to the optimal solution. With one routing constraint, each label contains two elements, the cost and weight, associated with a path from s to i. We only maintain labels for paths that are undominated, i.e., if p and p' are two different paths from s to i, and path p' has higher cost than path p, then this path is undominated only if it has strictly lower weight than path p. We can also omit some labels based on feasibility requirements: if the path corresponding to a label cannot be extended to reach node t within the weight limit (i.e., the path is not a sub-path of a feasible route for the commodity), then we can ignore this path. The method initializes the problem by assigning the label (0, 0) to node s, and then iteratively chooses the lowest cost label among all labels that have not been previously chosen. Since each node can have up to Q (the weight limit) labels (assuming nonnegative integer weights), the node labeling approach may not be effective when Q is large. Preprocessing techniques can significantly improve the empirical computational performance of the node labeling algorithm. These techniques consist of feasibility tests to identify and delete nodes and arcs that an optimal solution will not use. Using information from a Lagrangian relaxation of the weight constraint, e.g., to prune node labels, yields further improvements. In extensive computational tests, the node labeling algorithm with preprocessing is more effective than scaling techniques (Sect. 4.1.1). The node labeling approach can be extended to the WCSP problem with multiple weight constraints.

To conclude, in the context of the NDRR problem, the CSP problem is interesting and relevant because: (1) it captures the core NDRR feature of finding an origin-to-destination path for each commodity subject to routing constraints; (2) the CSP problem has received significant attention in the literature on approximation algorithms since the single commodity structure makes it more tractable; and (3) the CSP problem arises as a subproblem when we consider decomposition algorithms such as Lagrangian relaxation and column generation for solving the general NDRR problem (see Sect. 5). The CSP approximation algorithms and analysis may provide leads for analyzing the worst-case performance of approximation methods for the NDRR problem, although the presence of shared fixed costs, across commodities, in the NDRR problem may significantly complicate the analysis (possibly accounting for the lack of analogous results on network design problems, in general). The CSP algorithms, particularly the node labeling approach, can serve to solve subproblems quickly in NDRR decomposition approaches. We note that the problem reduction methods and two classes of valid inequalities—the Route Composition and Route Coordination inequalities—that BLM developed for the general NDRR problem also apply to the WCSP problem. With these inequalities, solving the strengthened WCSP model using state-of-the-art integer programming solvers may also be competitive. To our knowledge, this approach has not been tested.

4.2 *Hop-Constrained Routing and Design Problems*

We now consider the special version of the commodity routing requirement in which all arc weights are equal to one. If H^k denotes commodity k's weight limit in this constraint, the routing requirement states that each commodity k must use a path that contains no more than H^k arcs (hops). Therefore, we refer to this restriction as a *hop constraint* and to the corresponding weight limit as the hop-limit. (More generally, if all arcs have the same weight, not necessarily one, we can scale the weights and scale and round down the weight limit to convert the constraint to a hop constraint.) We refer to this special case of the NDRR problem with only hop restrictions as the *Hop-constrained Network Design* (HCND) problem. Balakrishnan and Altinkemer (1992) were the first to study the HCND problem; they develop and test a solution procedure based on Lagrangian relaxation. The literature has largely focused on a restricted version that we call the Hop-constrained Tree (HCT) problem in which there is a single source or root node that needs to be connected to other specified nodes, called terminal nodes, via a *tree* network, and all routing costs are zero. The problem is typically defined over an undirected network, and assumes the same hop-limit H for all commodities. If all nodes of the network, except the root node, are terminal nodes, then the required configuration is a spanning tree, i.e., the problem is a Hop-constrained Minimum Spanning Tree (HCMST). Otherwise, the design is a tree that spans the root node and all terminal nodes, and optionally includes non-terminal, i.e., Steiner, nodes. We refer to this latter problem as the Hop-constrained Steiner Tree problem. This section discusses approximation schemes for the HCMST problem, a layered network representation of hop-constrained path and tree problems that yields extended (tighter) model formulations, and some polyhedral results for these problems.

4.2.1 Approximation Algorithms for the HCMST Problem

Approximation algorithms for constrained tree problems largely focus on the Diameter Constrained Minimum Spanning Tree (DCMST) problem which, as we discuss next, is related to the HCMST problem. Given a maximum permitted diameter D, the DCMST problem seeks a minimum cost spanning tree such that the number of edges (hops) between any two pairs of nodes is at most D. If the diameter is even, say, $D = 2H$, we can solve n HCMST problems, each with a different node as the root node and hop-limit equal to H, and pick the lowest cost solution among these n problems as the optimal configuration for the DCMST problem. If D is odd and equals $(2H + 1)$, then any feasible DCMST solution must have an edge (i, j) such that every other node is connected to either node i or node j via a path containing no more than H edges. Thus, if we merge (or contract) nodes i and j, the solution is a DCMST with even diameter $(D - 1) = 2H$ (with the merged node as its center). So, we can solve the original DCMST problem by solving $|\mathscr{A}|$ HCMST problems, each obtained by contracting one edge of the original network. Conversely, we can also transform a HCMST problem into an equivalent DCMST problem as follows. Given the root node s and hop-limit H of the HCMST problem, we augment the network by adding two strings of H nodes, incident from node s, connected by zero cost arcs. Then, solving a DCMST problem, with diameter limit $D = 2H$, over the augmented network yields the HCMST solution rooted at node

s and satisfying the hop-limit. So, given an approximation algorithm with known performance guarantee for the DCMST problem, we can obtain an approximate solution with the same guarantee for the HCMST problem. The DCMST problem is NP-Hard even with $D = 4$ and with edge costs that are all either one or two (Garey and Johnson 2002), motivating the exploration of approximation methods. For instance, for the Diameter Constrained Steiner Tree problem (a generalization of DCMST in which the solution is only required to span a subset of nodes called terminal nodes and can optionally span other nodes called Steiner nodes), an approximation algorithm combining greedy selection and exhaustive search has a worst-case ratio of $O(\log(|T|))$, where T is the set of terminal nodes.

Interestingly, the HCMST special case with $H = 2$, which we call the two-hop HCMST problem, is equivalent to the uncapacitated facility location (UFL) problem. Given an instance of the UFL problem (with a dummy source node which is connected to all facility nodes), we can construct an equivalent two-hop HCMST instance by adding zero cost arcs between the facility nodes. Conversely, we obtain a UFL instance (with the root node as the dummy source node) corresponding to any two-hop HCMST instance by defining both a facility and a customer for each original non-root node, and assigning the cost for each original arc (i, j) to the arc from plant i to customer j (this cost is zero when $i = j$). These transformations imply that the approximation results for the UFL problem, such as the constant worst-case bounds based on greedy and cost scaling methods, also apply to the two-hop HCMST problem with metric costs. Developing a constant bound algorithm for the general HCMST problem remains an open problem.

4.2.2 Polyhedral Results for Hop-Constrained Path Problems

In Sect. 3, we discussed some polyhedral results for the general NDRR problem. For NDRR special cases when the underlying flow problem is a hop-constrained path or tree problem, there are specialized valid inequalities and polyhedral results for the underlying flow problems (assuming that all origin-to-destination paths must be elementary).

For the Hop-constrained Shortest Path (HCSP) problem, it is possible to characterize the underlying polytope when the hop-limit H is small and fixed. Since the HCSP problem has only one commodity, we omit the commodity index k in the following discussions. For $H = 2$, the solution can only contain arcs of the type (s, i) or (i, t) for some node $i \in \mathcal{N}$. Therefore, we can set the flow $x_{ij} = 0$ on all other arcs (i, j) that are not incident at either node s or t. These equalities, together with the flow conservation constraints for nodes s and t, and nonnegativity requirements on the x_{ij} variables, completely describe the convex hull of feasible solutions to the HCSP problem with $H = 2$. For $H = 3$, the flow conservation constraints at all nodes, the nonnegativity requirements on the full set of x_{ij} variables, and the inequalities

$$x_{si} - \sum_{j \in \mathscr{N} \setminus \{s,t\}} x_{ij} \geq 0 \qquad \forall i \in \mathscr{N} \setminus \{s,t\} \tag{8.9}$$

together give a complete description of the HCSP polytope.

Constraint (8.9) is a special version of a broad class of inequalities called *jump* inequalities. The basic jump inequality has the following structure. Let $V_1, V_2, \ldots, V_{H+2}$ be pairwise node-disjoint sets that partition the node set $\mathscr{N}$, with $V_1 = \{s\}$ and $V_{H+2} = \{t\}$. Define jump $J = \cup_{1 \leq i \leq j-2}\{V_i, V_j\}$, where $\{V_i, V_j\}$ is the set of arcs (a, b) such that $a \in V_i$ and $b \in V_j$. If $J(s\text{-}t, H)$ denotes the set of all jumps, then the jump inequality is

$$\sum_{(a,b) \in J} y_{ab} \geq 1 \qquad \forall J \in J(s\text{-}t, H) \tag{8.10}$$

By definition of the jump J, if an s-to-t path does not use any of the arcs in J, then the path must have at least $(H + 1)$ arcs, and so is not a feasible path for the hop-constrained problem. Lifted versions of these jump inequalities can define facets for HCSP problems with higher hop-limits ($H > 3$).

4.2.3 Layered Networks and Extended Formulations for Hop-Constrained Problems

When the routing requirement is a hop constraint, defining the commodity flows over a *layered* (expanded) network provides a convenient and intuitive representation of the network design (or shortest path) problem and also yields tighter formulations. We first discuss the structure and properties of the layered network for a single commodity (e.g., for the HCSP problem), and then address extensions to problems with multiple commodities, including hop-constrained tree problems. An important by-product of these layered network representations is that, using projection techniques on the associated extended formulations, we can obtain strong valid inequalities in the original space of design variables.

Layered Network Representation for Hop-Constrained Paths

Suppose a commodity from s to t must be routed on a path containing no more than H arcs (in the following discussion, we omit this commodity's index). The original NDRR problem formulation [*NDRR*] defines (binary) commodity flow variables x_{ij} on the original network, and imposes the hop constraint as $\sum_{(i,j) \in \mathscr{A}} x_{ij} \leq H$. Instead, suppose we define the flow variables over the following expanded network containing $(H + 1)$ layers, indexed from $h = 1$ to $h = (H + 1)$. Layer 1 contains only the source node s, and layer $(H + 1)$ only the sink node t. Each intermediate layer, $h = 2, 3, \ldots, H$, contains a copy of every node $i \neq s$, labeled as node $< i, h >$. The source node has label $< s, 1 >$, and sink node in the last layer has label $< t, H + 1 >$. If the original graph contains an arc (i, j), in the layered graph we connect node $< i, h >$ to node $< j, h + 1 >$, except when h is H we only consider

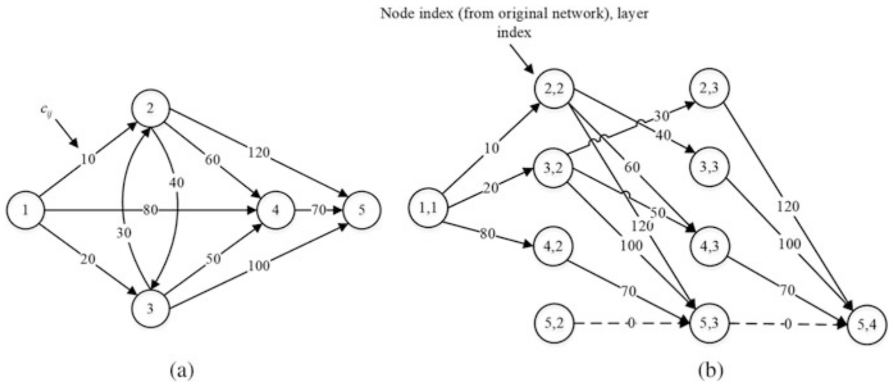


Fig. 8.2 Example of layered network for one commodity. (**a**) Original network. (**b**) Layered network with $H = 3$

$j = t$. We also add 'dummy' arcs, with zero cost, from $< t, h >$ to $< t, h+1 >$ for $h = 2, 3, \ldots, H$. Since the hop limit H must be less than the number of nodes n (assuming that we only permit elementary paths, which holds if all costs are non-negative), the size (number of nodes and arcs) of the layered network is no more than n times the size of the original network. Figure 8.2 illustrates this construction. Figure 8.2a shows the original network with source node $s = 1$ and sink node $t = 5$. Figure 8.2b shows the layered network with hop-limit $H = 3$.

From this construction, we can readily see that any path from $< s, 1 >$ to $< t, H+1 >$ in the layered network satisfies the hop constraint, and conversely every feasible path in the original network has a corresponding path from $< s, 1 >$ to $< t, H+1 >$ in the layered network. Therefore, if we define the routing variables as arc flows over the layered network (instead of the original network), then we do not need to explicitly impose the hop constraints on the routing variables. Specifically, instead of using the flow variables x_{ij} defined over the original graph, we now define disaggregated *hop-indexed* flow variables x_{ij}^h, for $h = 1, 2, \ldots, H$. The variable x_{ij}^h takes the value one if arc (i, j) is the h^{th} hop on the commodity's route from origin s to destination t, and is zero otherwise. Equivalently, x_{ij}^h is the flow from node $< i, h >$ to node $< j, h+1 >$ in the layered graph. The flow conservation constraints are:

$$\sum_{j \in \mathcal{N}_i^+} x_{ij}^h - \sum_{j \in \mathcal{N}_i^-} x_{ji}^{h+1} = \begin{cases} 1 \text{ if } i = s, \\ -1 \text{ if } i = t, \\ 0 \text{ otherwise.} \end{cases} \quad \forall h = 1, 2, \ldots, H-1. \tag{8.11}$$

For the HCSP problem, since any elementary s-to-t path cannot contain an arc (i, j) on more than one of the H hops, the 'routing' cost associated with each disaggregated flow variable x_{ij}^h is the same as the routing cost c_{ij} on the original arc (i, j). So, the HCSP problem's layer-indexed formulation, denoted as [*L-HCSP*], minimizes $\sum_{(i,j)\in\mathscr{A}}\sum_{h=1}^{H} c_{ij}x_{ij}^{h}$ subject to the flow conservation constraints (8.11) and integrality requirements $x_{ij}^{h} = 0$ or 1 for all $(i, j) \in \mathscr{A}, h = 1, 2, \ldots, H$. This model is the same as the formulation for the (unconstrained) shortest path problem over the layered network. Indeed, we can relate the HCSP problem's interpretation as the shortest path in the layered network to the well-known dynamic programming recursion: $d(j,h) = \min\{d(j,h-1), \min_{i:(i,j)\in\mathscr{A}}\{d(i,h-1)+c_{ij}\}\}$, where $d(j,h)$ denotes the shortest distance from node s to node j using h or fewer hops, for solving the hop-constrained shortest path problem (e.g., Lawler 1976). These observations imply that formulation [*L-HCSP*] is exact, i.e., it completely describes the HCSP polytope. In contrast, the LP relaxation of the original NDRR formulation, specialized to the HCSP problem, can have fractional solutions with non-zero integrality gap. Thus, although the layered formulation contains H (which is $O(n)$) times as many variables as the original formulation [*NDRR*] applied to the HCSP problem, the use of hop-indexed flow variables serves to strengthen the LP relaxation and close the integrality gap.

Layered Network and Extended Formulation for Hop-Constrained Trees
As noted earlier, the HCT problem requires designing a minimum cost tree that connects a root node s to a specified set T of terminal nodes, with a hop-limit of H on each root-to-terminal path. When T includes all nodes except the root node, the design is a spanning tree; otherwise, it is a Steiner tree. We can view this problem as a tree-constrained multicommodity HCND problem containing one commodity k for each terminal node that originates at the root node s and has node $k \in T$ as its destination. Applying the previous hop-indexed disaggregation to each commodity's flow variables, we obtain a formulation that is stronger than the [*NDRR*] formulation with added tree constraints.

4.2.4 Extended Formulations for General NDRR Problems

The models based on layered networks and the hop-indexed variables discussed in the previous section add to the rich history of using disaggregate variables to improve the effectiveness of formulations. Another related development is the use of extended formulations, obtained by adding new variables, for various combinatorial optimization problems. Research on this topic of extended formulations started mainly as a theoretical tool, but in recent years researchers have also examined their value for strengthening the LP relaxation when solving difficult integer programs (see Wolsey 2011). We can interpret the formulations based on the layered network representation of hop-constrained problems as extended formulations for the underlying problem.

The layered network concepts and modeling enhancements can also extend to more general NDRR problems, e.g., with multiple sources and destinations, without explicit tree configuration requirements, and with general (and multiple) routing constraints (vs. hop limits). For instance, we can represent problems with a general weight constraint (with arbitrary positive integer coefficients) in the following way.

Assume, for notational simplicity, that the integer arc weights and weight limits are the same for all commodities. For a routing constraint with a limit of Q for commodity k, we have $(Q + 1)$ instead of $(H + 1)$ layers in the layered network representation. Arcs between layers are not always between adjacent layers as is the case for the hop-constrained version. Instead, for each arc (i, j) with weight q_{ij} in the original network, the layered network contains an arc from node $< i, q >$ to node $< j, q + q_{ij} >$ for $1 \le q \le (Q - q_{ij} + 1)$. Corresponding to each such arc, we define a disaggregated flow variable x_{ij}^{kq}, and use flow conservation constraints analogous to (8.11) and the forcing constraints $\sum_{q=1}^{Q-q_{ij}+1} x_{ij}^{kq} \le y_{ij}$. Further, if the solution is required to be a tree, then we can also disaggregate the design variables as long as the arc weights and weight limits are the same for all commodities. Note that the size of the layered network, and hence the number of variables in the disaggregate formulation, is pseudo-polynomial since it depends on the weight limit Q. When this limit is very large or if there are multiple routing constraints, the extended formulation will be too large to solve directly using general purpose integer programming solvers. To mitigate this difficulty, we can apply an iterative approach that starts with a small layered network, and increases its size until we obtain a feasible solution that is sufficiently close to optimal.

If the problem imposes more than one routing constraint for each commodity, we can develop a 'multi-dimensional' layered network that implicitly captures multiple routing requirements. For instance, suppose there are two routing constraints with weight limits Q^1 and Q^2. Then, the multi-dimensional layered network contains $Q^1 Q^2$ layers containing nodes of the form $< i, q^1, q^2 >$, with $1 \le q^m \le (Q^m + 1), m = 1, 2$, and arcs from this node to node $< j, q^1 + q_{ij}^1, q^2 + q_{ij}^2 >$ for every arc (i, j) of the original network. Using the flow variables (and design variables, for tree sub-networks) on this layered network, we can develop an extended formulation for the NDRR problem.

In summary, for network design problems with hop limits or general weight constraints, layered network representations permit incorporating these routing restrictions implicitly by defining appropriate disaggregated decision variables instead of explicitly adding routing restrictions in the basic NDRR problem. This approach yields tighter formulations, but at the expense of requiring many more decision variables. Note that these extended formulations also apply to the capacitated version of the problem, e.g., the CNDRR problem with hop limits as the routing restrictions. For this problem, in each arc capacity constraint, we replace the original arc flow variables with the sum of the corresponding layer-indexed variables. Moreover, we can also add cuts developed for capacitated network design (using the sum of disaggregate flow variables in place of original arc flow variables) to this model to further strengthen the model. Since the layered formulation is tighter than the previous CNDRR model, it can improve the performance of the decomposition schemes discussed next.

5 Decomposition Strategies for the NDRR Problem

Section 3 discussed BLM's cutting plane approach to solve the general NDRR problem, and Sect. 4 outlined an alternate approach, using extended formulations, to strengthen the problem's LP relaxation and accelerate integer programming solvers. This section outlines other possible solution strategies based on decomposition methods, including Lagrangian relaxation, column generation, and Benders decomposition.

5.1 Lagrangian Relaxation

Lagrangian relaxation techniques are attractive when the problem contains embedded special structures that can be solved more easily. The NDRR problem has two such structures, corresponding respectively to uncapacitated network design and constrained shortest paths. If we dualize the routing constraints (8.4), the resulting subproblem is an uncapacitated fixed charge network design problem. Although this problem is NP-hard, Balakrishnan et al. (1989) describe a dual ascent procedure that is very effective in solving problems of reasonable size. When used as the procedure to solve Lagrangian subproblems, it may be possible to further accelerate the procedure by warm-starting it using dual values from the previous iteration. However, identifying a solution to the original NDRR problem that satisfies all the routing constraints may be difficult (recall from Sect. 2 that, even if we are given the network design, finding feasible solutions can be NP-hard when there are multiple routing constraints).

An alternative Lagrangian relaxation scheme consists of dualizing the forcing constraints (8.3), resulting in $|\mathscr{K}|$ CSP subproblems, one for each commodity k. CSP solution methods such as the node labeling algorithm (Sect. 4.1.2) are quite effective and quick. Potentially, when repeatedly solving CSP problems for a commodity, each of which differ only in the arc cost coefficients (which depend on the Lagrangian multiplier values), we can modify these methods to improve their performance (e.g., previous feasible solutions yield upper bounds for the current problem). Of course, since the computational effort for these pseudo-polynomial algorithms depends on the weight limits, solving each subproblem can be time consuming when these limits are large. Also, with a large number of arcs and commodities, the number of forcing constraints, and hence Lagrangian dual variables, is very large; so, the convergence of the Lagrangian dual problem may be slow.

We note that neither subproblem in the above two Lagrangian relaxation schemes satisfies the integrality property (i.e., the optimal solution to the LP relaxation of the subproblem may have fractional values). So, the best Lagrangian lower bound can exceed the LP lower bound, obtained by solving the LP relaxation of the original NDRR problem formulation (8.1)–(8.5). So, for problem instances where using either of the above subproblem solution procedures is practical, using Lagrangian relaxation at intermediate nodes of a branch-and-bound procedure can outperform standard LP-based branch-and-bound algorithms.

For the CNDRR problem, we have additional choices for the Lagrangian scheme. For instance, dualizing the arc capacity constraints results in NDRR subproblems, whereas relaxing the routing restrictions yields capacitated network design subproblems. To further simplify the subproblems, we can dualize additional constraints such as the forcing constraints or the flow conservation equations to obtain CSP or knapsack subproblems. In general, capacitated design problems tend to be more difficult to solve using this technique unless the model is further strengthened with valid inequalities (such as design inequalities based on arc capacity constraints).

5.2 Column Generation (Dantzig-Wolfe Decomposition)

As noted in Sect. 2, instead of formulating the NDRR problem using arc flow variables, we can also consider a path selection formulation that uses path flow variables corresponding to feasible origin-to-destination paths (satisfying the routing restrictions) for each commodity. We can view this reformulation (from the arc to path representation) as a change of variables, just as the layered network representation replaces the original arc flow variables with layer-indexed arc flow variables. Since the path flow model only considers origin-to-destination paths that meet the routing constraints, it has a stronger LP relaxation. However, since the number of such paths is exponential in the network size, we cannot explicitly solve the full model (except when the number of candidate paths is limited due to highly restrictive weight or hop limits). Instead, we can apply a column generation technique that iteratively generates promising paths based on the LP dual solution, embedded in a branch-and-price procedure to close the integrality gap. Applying column generation to the general NDRR problem requires iteratively solving a CSP subproblem for each commodity k in order to find a feasible $O(k)$-to-$D(k)$ path with negative reduced cost (not surprisingly, these pricing problems are the same as the Lagrangian subproblems that we need to solve when we dualize the forcing constraints of the arc flow model). The restricted master problem (RMP) is a linear program that chooses the path to be used for each commodity k from among the paths generated so far. The optimal dual prices of the RMP determine the arc costs in the CSP subproblems. Branching is needed to reduce the optimality gap between the LP value and the current best upper bound.

For the NDRR path flow model, the column generation procedure can require excessive number of iterations to converge because of the huge number of candidate paths for each commodity. One cause of slow convergence is that, during the initial iterations, the RMP (with only few columns) may not approximate the full problem very well and so its solution may not be close to optimality for the full problem. Consequently, the dual prices at these iterations may not adequately approximate the optimal dual prices of the full problem, causing the procedure to generate new columns that are not very useful. These columns can cause the RMP solutions to vary widely from iteration to iteration. Stabilization techniques can improve the convergence of column generation procedure by mitigating these difficulties. One successful approach has been to use a pre-chosen stabilization point, and constrain the RMP solutions to remain “close” to the stabilization point through the use of penalty functions.

Another technique is to improve the LP bounds of column generation model by adding cuts to the master problem. This technique, known as branch-and-cut-and-price, can be useful but care is necessary since the added cuts introduce new dual variables that change the structure of the pricing operation. If the pricing subproblem becomes difficult or intractable, then the column generation approach is impractical. However, for specific types of added cuts known as robust cuts, the basic structure of the pricing operation is unchanged. For example, in the arc-path approach to column generation, cuts using only design variables are robust.

The column generation approach also applies to the CNDRR problem, except that the RMP now also includes the arc capacity constraints (expressed in terms of the path flow variables). Again, it is important to strengthen the master problem by adding valid inequalities, preferably using robust cuts so as not to complicate the subproblems. Column generation has proven to be among the most successful methods for related capacitated problems such as vehicle routing and Capacitated Minimum Spanning Tree (CMST) problems. The CMST problem requires find the minimum cost spanning tree such that the total demand in each subtree of a designated root node does not exceed the capacity C of the arcs incident from the root node. For this problem, instead of paths, the columns represent feasible subtrees (that satisfy the capacity constraint) with degree one at the root node. Unfortunately, the pricing subproblem is strongly NP-hard and not practical computationally, necessitating some improvements to the branch-and-price approach such as modifying the column space. Specifically, instead of subtrees, using a set of arborescence-like structures permits a pseudo-polynomial algorithm (with respect to C) for the pricing subproblem. We can build upon these methods to solve CNDRR variants such as CMST with hop or more general routing restrictions. For instance, we can use a layer-indexed model for the pricing subproblem to capture both the hop limits and capacity limit for arcs incident to the root node. Such extensions provide fertile ground for further work.

5.3 Benders Decomposition

Benders decomposition entails fixing a subset of variables in a master problem, and solving LP subproblems whose dual values induce so-called Benders cuts in the master problem. Modern implementations of this approach, known as Benders branch-and-cut, embed the procedure into a branch-and-bound framework, add a Benders cut at each node in the search tree, and solve the master problem only once.

For the NDRR problem, if we fix the design variables at values $y_{ij} = \overline{y}_{ij}$ in the master problem, the resulting subproblems are CSP problems for each commodity k over the candidate network formed by $\overline{\mathscr{A}} = \{a \in \mathscr{A} | \overline{y}_{ij} = 1\}$. Unfortunately, the resulting subproblems are integer programs, and so Benders decomposition is not directly applicable. However, for the special case when all the routing constraints (8.4) are hop-limits, by using the disaggregated (hop-indexed) flow variables, the Benders subproblems are linear programs since they are simply (unconstrained) shortest path problems over the layered network for each commodity (but only including arcs chosen by the master problem). The Benders subproblems can, however, be infeasible because, in the current design chosen by the Benders master problem, the destination $D(k)$ may not be reachable from the origin $O(k)$ within the desired hop-limit (i.e., $O(k)$ and $D(k)$ are not connected in the layered network). In this case, we must generate and add a Benders feasibility cut to the master problem. In general, it is difficult to generate effective Benders feasibility cuts; this topic is currently an active area of research. Another possible strategy is to skip adding an feasibility cut and continue the branching process. This approach trades off the additional effort required in the search tree with the benefit of not generating feasibility cuts.

As we noted in Sect. 4.2.4, for NDRR problems with one routing constraint, having non-unitary coefficients, we can use the pseudo-polynomial layered network to model the disaggregate flow variables. In this case too, the subproblem of selecting a feasible path for a given set of design variables is a linear program, permitting the application of Benders decomposition.

Finally, the Benders approach also extends to CNDRR problems, except that feasibility of subproblems now depends on both whether the design contains at least one feasible origin-to-destination path (satisfying routing restrictions) for every commodity, and also meets arc capacity constraints (across all commodities). We can potentially improve Benders performance by tightening the LP relaxation of the design problem formulation in the master problem by adding cuts that only involve the design variables, such as those obtained using projection techniques of the capacitated network design problem, or reformulating the problem using disaggregated variables. Note that adding cuts to the Benders master problem is analogous to adding robust cuts for column generation. Master problem cuts (robust cuts) do not complicate the solution of the Benders subproblems (column generation pricing operation). Adding more general cuts can complicate the efficient solution of the subproblems just as non-robust cuts complicate the column generation pricing operation.

6 Bibliographical Notes

Valid Inequalities and Cutting Plane Methods for the General NDRR Problem Although network design problems have been studied extensively (e.g., Magnanti and Wong 1984; Balakrishnan et al. 1997; Crainic 2000), little research has been done on the NDRR problem. Barnhart and Schneur (1996), Armacost et al. (2002), and recently, Yildiz and Savelsbergh (2019) study optimal design for express delivery using ground and air transportation in order to determine multimodal time-sensitive origin-destination routes. Other related network design problems that also have a flavor of network design and/or flow routing with restrictions include reliable path routing, reliable network design, and survivable network design (e.g., Balakrishnan et al. 2009). Balakrishnan et al. (2017) (BLM) is the first paper to address the general NDRR problem, and develop a tailored cutting plane-based approach for this problem. The discussion in Sect. 3 is largely based on this paper. BLM provides a more general framework and treatment of the three classes of valid inequalities discussed in Sect. 3. Moreover, they describe some sophisticated lifting procedures, and prove that some versions of these inequalities are facets of the NDRR polyhedron. They report extensive computational results from applying the cutting plane procedure (using heuristic separation procedures to iteratively find violated inequalities) at the root node of a branch-and-bound algorithm, combined with an optimization-based heuristic method, for a variety of NDRR test problems containing up to 80 nodes, 320 arcs, and 240 commodities.

Extension to Capacitated NDRR Problems Magnanti et al. (1993, 1995) were among the first to study the capacitated network design problem where multiple modular facilities can be installed on the network arcs. They developed the cutset, residual capacity and 3-partition inequalities, and studied their theoretical and computational effectiveness. Bienstock and Gunluk (1996) developed several facet-defining inequalities that extend the cutset and the 3-partition inequalities. Atamturk and Rajan (2002) study single-arc set relaxations of the problem and show that the separation problem of the residual capacity inequalities (for the splittable case) can be solved in linear time while the separation problem for the c-strong inequalities (developed by Brockmuller et al. (2004) for the unsplittable case) is NP-hard. They extend the c-strong inequalities and conduct computational experiments to test the effectiveness of these inequalities. Benhamiche et al. (2016) study the polyhedral structure of a model where a commodity flow cannot split even across two different facilities on the same arc. Gendron et al. (1999) provide a survey of multicommodity capacitated network design models. They also summarize the theoretical strengths and present a computational comparison of several different relaxations of an arc-based formulation of the problem.

Approximation Schemes for the Budget-Constrained Shortest Path (BCSP) Problem Vazirani (2013) and Williamson and Shmoys (2011) provide comprehensive discussions of approximation schemes for various problem settings. For the BCSP problem over acyclic graphs, Warburton (1987) was the first to develop a FPTAS. The complexity of his approximation scheme is $O(n^3\epsilon^{-1}\log(n)\lceil\log(UB)\rceil)$, where ϵ is the performance guarantee (i.e., the heuristic solution value is within a factor of $(1+\epsilon)$ of the optimal solution value) and UB is an upper bound on the optimal solution value. Hassin (1992) employs the principles underlying this method, but uses a constant bound on the ratio of UB to LB to develop an approximation algorithm with complexity $O(|\mathscr{A}|n^2\epsilon^{-1}\log(n\epsilon^{-1}))$. Section 4.1.1

summarizes this method. Lorenz and Raz (2001) further improve the approximation scheme, achieving a n-fold reduction in time complexity, by simplifying the method for obtaining upper and lower bounds, and permitting a larger approximation error in the first stage. This method runs in $O(|\mathscr{A}|n(\log\log n + 1/\epsilon))$. Ergun et al. (2002) also improve Hassin's algorithm but by making the scaling factor adaptive, starting with a large scaling factor. As the difference between current upper and lower bounds decreases, the method reduces the scaling factor. This strategy improves solution quality without adversely affecting the running time, resulting in a FPTAS with time complexity of $O(|\mathscr{A}|n\epsilon^{-1})$.

CSP Solution Algorithms One possible drawback to the Handler-Zang (*HZ*) approach for solving BCSP problems is that reducing the gap may require generating a large number of rth shortest paths. Desrochers and Soumis (1988) propose solving the BCSP problem by generalizing Dijkstra's shortest path algorithm. Dumitrescu and Boland (2003) use preprocessing techniques to accelerate the node-labeling algorithm, and demonstrate computationally that these methods can improve performance by an order of magnitude. Feng and Korkmaz (2015) and Pugliese and Gueriero (2013) provide some suggestions to the reduce the number of r^{th} shortest paths needed to close the gap. Feng and Korkmaz also discuss an extension of the *HZ* method to solve weight-constrained shortest path (WCSP) problems. Pugliese and Gueriero (2013) review the methodological literature for WCSP problems with multiple metrics, and even with negative arc costs.

Approximation Algorithms for the Diameter-Constrained and Hop-Constrained Minimum Spanning Tree (DCMST and HCMST) Problems Kortsarz and Peleg (1999) analyze the heuristic worst-case performance of a DCMST problem with maximum diameter of five. Marathe et al. (1998) develop approximation algorithms for a generalization of the Diameter-constrained Minimum Spanning Tree (DCMST) problem where the weights associated with arcs are not necessarily one. This generalization requires the total weight of the path connecting any pair of nodes in the tree to be less than or equal a specified value. The authors' approach starts with clusters consisting of single nodes, and sequentially merges these clusters until just one cluster remains, which is the heuristic solution. Hassin and Levin (2003) study a problem in which the diameter is not fixed, but rather pairs of nodes have hop limits that belong to $\{1, 2, \infty\}$. Assuming metric edge costs, they develop a constant ratio algorithm. They also consider cases where the graph induced by node-pairs with hop-limit of one or two is a Hamiltonian graph or a 2-vertex connected graph. Althaus et al. (2005) develop a randomized algorithm with approximation ratio of $O(log(n))$ for the HCMST problem with metric costs.

Polyhedral Results for Hop-Constrained Design Problems For the Hop-constrained Shortest Path (HCSP) problem, Dahl and Gouveia (2004) provide a complete characterization of the underlying polytope when the hop limit H is small. Dahl (1998) originally introduced the jump constraints while Grotschel and Stephan (2014) propose a systematic way of generating jump constraints as well as other inequalities via projection of a HCSP problem formulation that uses hop-indexed variables (see Sect. 4.2.3). Although the basic jump inequalities do not necessarily define facets of the HCSP problem with general hop-limits, Reidl (2017) provides necessary and sufficient conditions for lifted jump inequalities to be facets of the HCSP polytope. Stephan (2009) identifies other facets by studying the polyhedral structure of related combinatorial problems.

Extended Formulations for Hop-Constrained Problems Gouveia (1998) is among the first researchers to study models with hop-indexed variables for hop-constrained problems. The variable disaggregation approach also extends to Hop-constrained Network Design (HCND) problems by defining hop-indexed flow variables for each commodity. This hop-indexed model is tighter than representing the hop constraints as routing requirements in formulation [*NDRR*], but the hop-indexed model does not fully close the integrality gap for HCND problems (unlike the situation for the HCSP problem). Researchers have used layered network representations and successfully applied the associated formulations with disaggregated (hop-indexed) flow variables to several special cases and variants of hop-constrained network design including minimum spanning tree problems with diameter constraints (e.g., Gouveia and Magnanti 2003; Gouveia et al. 2004, 2006), hub location with hop constraints (Camargo et al. 2017), and container shipping service selection with limited transshipments (Balakrishnan and Karsten 2017). The higher LP lower bounds of the hop-indexed model significantly accelerate branch-and-bound solution procedures for these problems.

Layered Network and Extended Formulations for Hop-Constrained Tree (HCT) Problems Gouveia et al. (2011), henceforth abbreviated as GSU, further strengthen this model by exploiting the problem's tree configuration requirement. Specifically, they show how to represent the HCT problem as a Steiner tree problem defined over a (single) layered network (instead of defining a separate layered network for each commodity). Effectively, this approach permits disaggregating the design variables (by hop index). For their equivalent Steiner tree problem, GSU consider a directed cut formulation (Maculan 1987) that contains only design variables (no flow variables since they do not consider routing costs), and uses cutset constraints to ensure connectivity from the root to every terminal node. GSU show that this model is tighter than the previous HCT formulation with hop-indexed flow variables defined over separate layered networks for each commodity. The authors also extend this approach to the DCMST problem. Their computational results demonstrate that using the stronger model reduces computational time by about two orders of magnitude compared to earlier methods (e.g., Gouveia and Magnanti 2003; Gouveia et al. 2004) that solve the model with only disaggregated flow variables. In Sect. 4.2.4, we discuss an extension of this technique of disaggregating the design variables to more general types of NDRR problems.

GSU's idea of disaggregating the design variables also relates to other interesting work. Ruthmair and Raidl (2011) apply disaggregation to Steiner tree problems with a single weight constraint for each commodity. To avoid solving the full extended formulation, they start with a small layered network obtained by deleting some nodes, removing the outgoing arcs for each deleted node, and redirecting its incoming arcs to a corresponding node in an earlier (later) layer. Solving the reduced network provides a valid lower (upper) bound. The approach iteratively increases the size of the layered network until the upper and lower bounds are sufficiently close. Boland et al. (2017) developed and applied a similar strategy in the context of time-space networks (which are related to layered networks).

Another way of reducing the size of the layered network is to apply scaling-and-rounding (sometimes known as discretization) to the weight constraint to reduce the size of its coefficients. In the context of time-space networks, the approach of scaling-and-rounding the time unit does not seem to be as effective computationally as the methodology described above (Boland et al. 2017, 2019).

Extended Formulations for General NDRR Problems Researchers have recognized the benefits of reformulating various network design problems (e.g., facility location, Steiner trees, and uncapacitated fixed charge network design) using disaggregated variables and forcing constraints. For example, for fixed-charge network design, instead of using one commodity to represent all the flow originating from a source node, replacing the single commodity with multiple commodities, each corresponding to one destination served by that source, yields tighter model formulations and improved solution techniques (see Magnanti and Wong 1984). For research related to extended formulations, see, for example, Vanderbeck and Wolsey (2010), Conforti et al. (2010), Conforti et al. (2014), and Fiorini and Pashkovich (2015). Applying projection techniques (e.g., Conforti et al. 2014) to the extended formulations (with disaggregated flow or design variables) can yield valid inequalities and facets for the original problem formulation [*NDRR*]. For related work on projection techniques applied to the CSP polyhedron, see Coulard et al. (1994), and Grotschel and Stephan (2014). Mirchandani (2000) uses projection for the capacitated network loading problem, and Rardin and Wolsey (1993) use projection for uncapacitated network design models with multiple sources and sinks for each commodity. Gouveia et al. (2011) apply projection techniques to the single layered network model for HCT problems to obtain valid inequalities for the base model.

Column Generation Branch-and-price, which embeds the column generation technique within a branch-and-bound tree search framework (Barnhart et al. 1998; Desrosiers et al. 1995), has proved successful for various types of integer optimization problems. Column generation has been very successfully used in crew scheduling and routing problems, with routing requirements that reflect, for instance, time or distance limits of vehicle routes and work shifts (see Lubbecke and Desrosiers 2005 for an extensive list of references on applications of column generation).

Stabilization and other techniques can be useful in improving the convergence of column generation. Lubbecke and Desrosiers (2005) and Lemaréchal et al. (1995) discuss the use of a pre-chosen stabilization point. These general ideas have proven very successful in the special context of delay-constrained minimum spanning trees and Steiner trees. Computational tests by Leitner et al. (2012) show that the stabilized version of column generation is about one order of magnitude faster than the usual column generation procedure for larger problems. The approach can solve difficult problems for networks with up to 999 nodes and about 10,000 arcs. Stabilized column generation is also at least competitive with state-of-the-art techniques using branch-and-cut applied to the single layered network model formulation (Gouveia et al. 2011).

Another convergence difficulty arises when the RMP primal solution is degenerate, implying multiple optimal dual prices in the master problem. Holloway (1973) suggests choosing an optimal dual solution that would benefit the overall convergence of column generation (i.e., choose a set of dual prices that would accurately reflect the optimal dual prices of the full problem). Such an approach would be similar in spirit to Magnanti and Wong (1981) (see also Rahmaniani et al. 2017) who propose exploiting degeneracy in Benders subproblems (whereas Holloway's proposal concerns degeneracy in the column generation master problem).

Uchoa et al. (2008) and Costa et al. (2019) discuss innovative branch-and-cut-and-price approaches to the Capacitated Minimum Spanning Tree (CMST) and the vehicle routing problems, respectively. Uchoa et al. use a new column space representation as well as robust cuts (see Poggi de Arago and Uchoa 2003) based on a new expanded representation of the arc flow variables based on capacity-indexed arc flows (similar to a layered network representation of arc flows) proposed by Gouveia and Martins (1999). Costa et al. (2019) give a comprehensive survey of techniques for improving the performance of column generation for vehicle routing including robust and non-robust cuts as well various other strategies. Some research has shown that the careful addition of non-robust cuts can improve the overall performance of the column generation approach. Thus, the cuts proposed in Sect. 3 (which are non-robust) might be of interest in the context of a column generation approach.

Benders Decomposition For an introduction to Benders decomposition and a general treatment of this approach, see, for example, Conforti et al. (2014). Over the years, researchers have suggested various techniques to improve Benders' classic approach. See Rahmaniani et al. (2017) for a comprehensive overview. As mentioned in Sect. 5.3, an active area of research is generating effective Benders feasibility cuts. See, for instance, Camargo et al. (2017).

Botton et al. (2013) apply Benders decomposition to a Hop-constrained Survivable Network Design problem. That is, in addition to hop constraints on the commodity routes, the problem also requires ensuring that each commodity has as least γ arc-disjoint origin-to-destination paths (which must all satisfy the hop limit restriction) in the chosen design. The objective is to minimize the total fixed cost of the chosen design arcs. For the design problem under consideration, at least one commodity's subproblem will always be infeasible until the relaxed master problem generates an optimal solution to the original design problem. The authors avoid this difficult issue by modifying the Benders branch-and-cut procedure by only occasionally generating Benders cuts (i.e. solving the subproblems). Thus, they trade off having a larger search tree for reducing the subproblem computation time. This modified approach is an order of magnitude faster than the usual Benders decomposition approach. The decomposition approach can solve medium to large sized problems with up to 41 nodes and 820 edges and is significantly faster than solving the original formulation with CPLEX.

As mentioned in Sect. 5, strengthening the LP relaxation of the model formulation can improve the performance of Benders decomposition. Section 6.2 of Rahmaniani et al. (2017) discusses various research work on adding valid inequalities to the Benders master problem to strengthen the overall formulation. Magnanti and Wong (1981) discuss a framework for evaluating different model formulations (having different sets of subproblem variables) in the context of Benders decomposition based on their LP relaxation strength when the master problem variables have fixed values. Certain formulations can offer a richer (better) set of Benders cuts than other ones.

7 Concluding Remarks

In this chapter, we have reviewed various applications of network design with routing requirements, identified the challenges of solving this problem, and outlined modeling and solution approaches for the problem. We next summarize the key observations and learnings, and identify some opportunities for future work.

Applications Route constrained network design problems arise in a broad spectrum of industries. These include the transportation industry, where the NDRR problem applications comprise networks on the land, sea, and air. There are also many applications in telecommunications and other areas such as electricity distribution and machine scheduling.

Polyhedral Approach Using preprocessing techniques combined with insights about the problem structure, Balakrishnan et al. (2017) derive and implement computationally effective facets and valid inequalities for the NDRR problem. For the CNDRR problem, the added capacity constraints make solving the problem more challenging. So, using tight formulations, with added valid inequalities to strengthen the formulation, will be key for effective CNDRR solution performance. The ideas outlined in Sect. 3 to develop integrated inequalities that jointly consider the routing and capacity restrictions provide interesting and useful research directions to pursue.

Constrained Shortest Path heuristics There is a wide spectrum of creative approaches including Lagrangian relaxation with path enumeration, generalized Dijkstra algorithm with preprocessing, and scaling. Different goals (e.g., improving computational efficiency vs. improving worst-case bounds) result in different types of heuristic improvements. Can we obtain heuristic worst-case bounds for the weight-constrained shortest path problem with multiple routing requirements?

Worst-Case Analysis Sections 4.1.1 and 4.2.1 discuss the worst-case analysis of approximation algorithms for some special cases of the NDRR problem. These theoretical studies are challenging but facilitate our understanding about which particular problem characteristics make the NDRR problem easier or harder to solve. Can we build upon these previous studies to develop and analyze the worst-case performance of heuristics for more general NDRR problems?

Layered Networks The layered network approach uses variable disaggregation to obtain a tighter LP relaxation for hop-constrained spanning trees. The tighter formulation improves computational performance. Importantly, removing disaggregated variables in this model via projection constitutes a systematic method for obtaining polyhedral results in the original problem space (see previous discussion on extended formulations for general NDRR problems). Adopting and extending this approach for other problems appears to be promising.

The layered network variable disaggregation technique is different from previous approaches. Instead of disaggregating a commodity into a finer set of commodities (as researchers have previously done for facility location and uncapacitated fixed charge network design), it disaggregates a flow variable into a series of hop-indexed flow variables (or a design variable into a series of hop-indexed design variables). Could there be other new variable disaggregation schemes for different types of network design problems?

Decomposition Techniques Leveraging advances (over the past several decades) in decomposition techniques (e.g., stabilization, improved column pricing methods), embedding within a tree search procedure (e.g., branch-and-price or branch-and-cut) and exploiting the structural properties of the NDRR problem solution results in useful algorithms for some of its special cases. Further exploitation of these advances appears to be a promising area for future research. Moreover, decomposition techniques are more flexible and can address problem variants such as stochastic or prize-collecting variants more easily than other types of solution techniques.

Capacitated Network Design with Routing Restrictions Network design problems become more challenging to solve if we just add routing restrictions or arc capacity constraints. Even the simplest versions of the combined CNDRR model, which has both types of constraints are NP-hard, and are likely to be quite difficult to solve. Our discussion has highlighted the longitudinal (single commodity, multiple arc) structure versus lateral (single arc, multiple commodity) structure of the routing and capacity constraints. Developing effective solution methods will require leveraging and integrating the principles and approaches developed for the NDRR and capacitated network design models. For each of the modeling and methodological approaches (polyhedral methods, extended formulations, Lagrangian relaxation, column generation, and Benders decomposition) presented in this paper, we have also discussed possible ways to extend them to the CNDRR problem. Perhaps, research to solve the CNDRR problem can begin by first addressing its special cases such as the capacitated minimum spanning tree with hop limits, capacitated hop-constrained network design, or two-commodity CNDRR before considering more general routing restrictions. These special cases have the advantage of providing a wider range of improved modeling options such as the layered network representation. Since there is little or no literature on the CNDRR problem and since this problem has considerable practical relevance, investigating and developing effective solution approaches for this problem is a promising and fruitful avenue for research.

References

Agarwal, R., & Ergun, O. (2008). Ship scheduling and network design for cargo routing in linear shipping. *Transportation Science, 42*, 175–196.

Ahuja, R. K., Jha, K. C., & Liu, J. (2007). Solving real-life railroad blocking problems. *Interfaces, 37*, 404–419.

Althaus, E., Funke, S., Har-Peled, S., Konemann, J., Ramos, E. A., & Skutella, M. (2005). Approximating k-hop minimum-spanning trees. *Operations Research Letters, 33*, 115–120.

Armacost, A. P., Barnhart, C., Ware, K. A. (2002). Composite variable formulations for express shipment service network design. *Transportation Science, 36*, 1–20.

Atamtürk, A., & Rajan, D. (2002). On splittable and unsplittable flow capacitated network design arc-set polyhedra . *Mathematical Programming, 92*, 315–333.

Balakrishnan A., & Altinkemer, K. (1992). Using a hop-constrained model to generate alternative communication network designs. *INFORMS Journal on Computing, 4*, 192–205.

Balakrishnan, A., & Karsten, C. V. (2017). Container shipping service selection and cargo routing with transshipment limits. *European Journal of Operational Research, 263*, 652–663.

Balakrishnan, A., Li, G., & Mirchandani, P. (2017). Optimal network design with end-to-end service requirements. *Operations Research, 65*, 729–750.

Balakrishnan, A., Magnanti, T. L., & Mirchandani, P. (1996) Heuristics, LPs, and trees on trees: Network design analyses. *Operations Research, 44*, 478–496.

Balakrishnan, A., Magnanti, T. L., & Mirchandani, P. (1997). Network design. In M. Dell'Amico, F. Maffioli, & S. Martello (Eds.), *Annotated bibliographies in combinatorial optimization* (pp. 311–334). New York: John Wiley and Sons.

Balakrishnan, A., Magnanti, T. L., & Wong, R. T. (1989). A dual-ascent procedure for large-scale uncapacitated network design. *Operations Research, 37*, 716–740.

Balakrishnan, A., Mirchandani, P., & Natarajan, H. P. (2009). Connectivity upgrade models for survivable network design. *Operations Research, 57*, 170–186.

Balakrishnan, A., Mirchandani, P., & Wong, R. T. (2020). On multi-constrained path, tree, and network design problems. Working paper

Barnhart, C., Jin, H., & Vance, P. (2000). Railroad blocking: A network design applications. *Operations Research, 48*, 603–614.

Barnhart, C., Johnson, E., Nemhauser, G., Savelsbergh, M., & Vance, P. (1998) Branch-and-price: Column generation for solving huge integer programs. *Operations Research, 46*, 316–329.

Barnhart, C., & Schneur, R. (1996). Air network design for express shipment service . *Operations Research, 44*, 852–863.

Benhamiche, A., Mahjoub, A. R., Perrot, N., & Uchoa, E. (2016). Unsplittable non-additive capacitated network design using set functions polyhedra. *Computers and Operations Research, 66*, 105–115.

Bienstock, D., Günlük, O. (1996). Capacitated network design – Polyhedral structure and computation. *INFORMS Journal on Computing, 8*, 243–259.

Boland, N., Hewitt, M., Marshall, L., & Savelsbergh, M. (2017). The continuous-time service network design problem. *Operations Research, 65*, 1303–1321.

Boland N, Hewitt M, Marshall, L., & Savelsbergh, M. (2019). The price of discretizing time: a study in service network design. *Euro Journal on Transportation and Logistics, 8*, 195–216.

Botton, Q., Fortz, B., Gouveia, L., & Poss, M. (2013). Benders decomposition for the hop-constrained survivable network design problem. *INFORMS Journal on Computing, 25*, 13–26.

Brockmüller, B., Günlük, O., & Wolsey, L. A. (2004). Designing private line networks: polyhedral analysis and computation. *Transactions on Operational Research, 16*, 7–24.

Camargo, R., de Miranda, G., Jr., O'Kelly, M., & Campbell, J. (2017). Formulations and decomposition methods for the incomplete hub location problem with and without hop-constraints. *Applied Mathematical Modelling, 51*, 274–301.

Conforti, M., Cornuejols, G., & Zambelli, G. (2010). Extended formulations in combinatorial optimization. *4OR: A Quarterly Journal of Operations Research, 8*, 1–48.

Conforti, M., Cornuejols, G., & Zambelli, G. (2014). *Integer programming*. Heidelberg, Springer.

Costa, L., Contardo, C., & Desaulniers, G. (2019). Exact branch-price-and-cut algorithms for vehicle routing. *Transportation Science, 53*, 946–985.

Coulard, C., Gamble, B., & Liu, J. (1994). The K-walk polyhedron. In D.-Z. Du & J. Sen (Eds.), *Advances in optimization and approximation*. Dordrecht: Kluwer Academic Publishers.

Crainic, T. G. (2000). Service network design in freight transportation. *European Journal of Operational Research, 122*, 272–288.

Dahl, G. (1998). The 2-hop spanning tree problem. *Operations Research Letters, 23*, 21–26.

Dahl, G., & Gouveia, L. (2004). On the directed hop-constrained shortest path problem. *Operations Research Letters, 32*, 15–22.

De Boeck, J., & Fortz, B. (2017). Extended formulation for hop constrained distribution network configuration problems. *European Journal of Operational Research, 265*, 488–502.

Desaulniers, G., Madsen, O. B., & Ropke, S. (2014). The vehicle routing problem with time windows. In P. Toth, & D. Vigo (Eds.), *Vehicle routing: Problems, methods, and applications, MOS-SIAM series on optimization* (Vol. 18, pp. 119–159). Philadelphia: SIAM.

Desrochers, M., & Soumis, F. (1988). A generalized permanent labelling algorithm for the shortest path problem with time windows. *INFOR: Information Systems and Operational Research, 26*, 191–212.

Desrosiers, J., Dumas, Y., Solomon, M., & Soumis, F. (1995). Time constrained routing and scheduling. In M. Ball, T. L. Magnanti, C. Monma, & G. L. Nemhauser (Eds.), *Handbooks in operations research and management science* (Vol. 8, pp. 35–139). Amsterdam: Elsevier.

Dumitrescu, I., & Boland, N. (2003). Improved preprocessing, labeling, and scaling algorithms for the weight-constrained shortest path problem. *Networks, 42*, 135–153.

Ergun, F., Sinha, R., & Zhang, L. (2002). An improved FPTAS for restricted shortest path. *Information Processing Letters, 83*, 287–291.

Estrada, M., & Robuste, F. (2009). Long-Haul shipment optimization for less-than-truckload carriers. *Transportation Research Record: Journal of the Transportation Research Board, 2091*, 12–20.

Feng, G., & Korkmaz, T. (2015). Finding multi-constrained multiple shortest paths. *IEEE Transactions on Computers, 64*, 2559–2572.

Fiorini, S., & Pashkovich, K. (2015). Uncapacitated flow-based extended formulations. *Mathematical Programming, 153*, 117–131.

Garey, M. R., & Johnson, D. S. (2002). *Computers and intractability: A guide to the theory of NP completeness*. San Francisco: W. H. Freeman.

Gendron, B., Crainic, T.G., & Frangioni, A. (1999). Multicommodity capacitated network design. In B. Sansò & P. Soriano (Eds.), *Telecommunications network planning* (pp. 1–19). Centre for Research on Transportation. Boston: Springer.

Gopalakrishnan, B., & Johnson, E. L. (2005). Airline crew scheduling: State-of-the-art. *Annals of Operations Research 140*, 305–337.

Gouveia, L. (1998). Using variable redefinition for computing lower bounds for minimum spanning tree and Steiner tree with hop constraints. *INFORMS Journal on Computing, 10*, 180–188.

Gouveia, L., & Magnanti, T. L. (2003). Network flow models for designing diameter-constrained spanning and Steiner trees. *Networks, 41*, 159–173.

Gouveia, L., Magnanti, T. L., & Requejo, C. (2004). A 2-path approach for odd diameter-constrained minimum spanning and Steiner trees. *Networks, 44*, 254–265.

Gouveia, L., Magnanti, T. L., & Requejo, C. (2006) An intersecting tree model for odd-iameter-constrained minimum spanning and Steiner trees. *Annals of Operations Research, 146*, 19–39.

Gouveia, L., & Martins, P. (1999). The capacitated minimal spanning tree problem: An experiment with a hop-indexed model. *Annals of Operations Research 86*, 271–294.

Gouveia, L., Simonetti, L., & Uchoa, E. (2011). Modeling hop-constrained and diameter-constrained minimum spanning tree problems as Steiner tree problems over layered graphs. *Mathematical Programming, 128*, 123–148.

Grandoni, F., Ravi, R., Singh, M., & Zenklusen, R. (2014). New approaches to multi-objective optimization. *Mathematical Programming, 146*, 525–554.

Grötschel, M., Monma, C. L., & Stoer, M. (1995). Design of survivable networks. In M. Ball, T. L. Magnanti, C. Monma, G. L. Nemhauser (Eds.), *Handbooks in operations research and management science* (Vol. 7, pp. 617–672). Amsterdam: Elsevier.

Grötschel, M., & Stephan, R. (2014). Characterization of facets of the hop-constrained chain polytope via dynamic programming. *Discrete Applied Mathematics, 162*, 229–246.

Handler, G., & Zang, I. (1980). A dual algorithm for the constrained shortest path problem. *Networks, 10*, 293–309.

Hassin, R. (1992). Approximation schemes for the restricted shortest path problem. *Mathematics of Operations Research, 17*, 36–42.

Hassin, R., & Levin, A. (2003). Minimum spanning tree with hop restrictions. *Journal of Algorithms, 48*, 220–238.

Holloway, C. (1973). A generalized approach to Dantzig-Wolfe decomposition for concave programs. *Operations Research, 21*, 210–220.

Karsten, C. V., Brouer, B. D., Desaulniers, G., & Pisinger, D. (2017). Time constrained liner shipping network design. *Transportation Research Part E, 105*, 152–162.

Kortsarz, G., & Peleg, D. (1999). Approximating the weight of shallow Steiner trees. *Discrete Applied Mathematics, 93*, 265–285.

Lawler, E. L. (1976). *Combinatorial optimization: Networks and matroids*. New York: Courier Corporation.

Leitner, M., Ruthmair, M., & Raidl, G. (2012). Stabilizing branch-and-price for constrained tree problems. *Networks, 61*, 150–170.

Lemaréchal, C., Nemirovskii, A., & Nesterov, Y. (1995). New variants of bundle methods. *Mathematical Programming, 69*, 111–147.

Lorenz, D. H., & Raz, D. (2001). A simple efficient approximation scheme for the restricted shortest path problem. *Operations Research Letters, 28*, 213–219.

Lübbecke, M. E., & Desrosiers, J. (2005). Selected topics in column generation. *Operations Research, 53*, 1007–1023.

Maculan, N. (1987). The Steiner problem in graphs. *Annals of Discrete Mathematics, 31*, 185–212.

Magnanti, T. L., Mirchandani, P., & Vachani, R. (1993). The convex hull of two core capacitated network design problems. *Mathematical Programming, 60*, 233–250.

Magnanti, T. L., Mirchandani, P., & Vachani, R. (1995). Modeling and solving the two-facility capacitated network loading problem. *Operations Research, 43*, 142–157.

Magnanti, T. L., & Wong, R. T. (1981). Accelerating Benders decomposition: Algorithmic enhancement and model selection criteria. *Operations Research, 29*, 464–484.

Magnanti, T. L., & Wong, R. T. (1984). Network design and transportation planning: Models and algorithms. *Transportation Science, 18*, 1–55.

Malandraki, C., Zaret, D., Perez, J., & Holland, C. (2001). Industrial engineering applications in transportation. In G. Salvendy (Eds.), *Handbook of industrial engineering* (3rd ed., pp. 787–824). New York: John Wiley and Sons.

Marathe, M. V., Ravi, R., Sundaram, R., Ravi, S. S., Rosenkrantz, D. J., & Hunt, H. B. (1998). Bicriteria network design problems. *Journal of Algorithms, 28*, 142–171.

Mirchandani, P. (2000). Projections of the capacitated network loading problem. *European Journal of Operational Research, 122*, 534–560.

Nemhauser, G. L., & Wolsey, L. A. (1988). *Integer and combinatorial optimization*. New York: Wiley.

Poggi de Arago, M., & Uchoa, E. (2003). Integer program reformulation for robust branch-and-cut-and-price. In L. Wolsey (Ed.), *Annals of Mathematical Programming in Rio* (pp. 59–61)

Pugliese, L. D. P., & Guerriero, F. (2013). A survey of resource constrained shortest path problems: Exact solution approaches. *Networks, 62*, 183–200.

Rahmaniani, R., Crainic, T. G., Gendreau, M., & Rei, W. (2017). The Benders decomposition algorithm: A literature review. *European Journal of Operational Research, 259*, 801–817.

Rardin, R. L., & Choe, U. (1979). *Tighter relaxations of fixed charge network flow problems*. Industrial and Systems Engineering Report J-79-18, Georgia Institute of Technology

Rardin, R. L., & Wolsey, L. A. (1993). Valid inequalities and projecting the multicommodity extended formulation for uncapacitated fixed charge network flow problems. *European Journal of Operational Research, 71*, 95–109.

Reidl, W. (2017). A complete characterization of jump inequalities for the hop-constrained shortest path problem. *Discrete Applied Mathematics, 225*, 85–113.

Ruthmair, M., & Raidl, G. (2011). Layered graph model and an adaptive layers framework to solve delay–constrained minimum tree problems. In O. Gunluk & G. Woeginger (Eds.), *IPCO 2011* (pp. 276–288). Berlin Heidelberg, Springer-Verlag.

Stephan, R. (2009). Facets of the (s,t)-path polytope. *Discrete Applied Mathematics, 157*, 3119–3132.

Uchoa, E., Fukasawa, F., Lysgaard, J., Pessoa, A., Poggi de Arago, M., & Andrade, D. (2008). Robust branch-cut-and-price for the capacitated minimum spanning tree problem over a large extended formulation. *Mathematical Programming, 112*, 443–472.

Vanderbeck, F., & Wolsey, L. A. (2010). Reformulation and decomposition of integer programs. In M. Jünger, T. M. Liebling, D. Naddef, G. L. Nemhauser, W. R. Pulleyblank, G. Reinelt, G. Rinaldi, & L. A. Wolsey (Eds.), *50 Years of integer programming 1958–2008* (pp. 431–502). Berlin Heidelberg: Springer-Verlag.

Vazirani, V. V. (2013). *Approximation algorithms*. New York: Springer Science & Business Media.

Vidal, T., Crainic, T. G., Gendreau, M., & Prins, C. (2013). Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. *European Journal of Operational Research, 231*:1–21.

Warburton, A. (1987). Approximation of pareto optima in multiple–objective, shortest-path problems. *Operations Research, 35*, 70–79.

Wilhelm, W. E., Damodaran, P., & Li, J. (2003). Prescribing the content and timing of product upgrades. *IIE Transactions, 35*, 647–663.

Williamson, D. P., & Shmoys, D. B. (2011). *The design of approximation algorithms*. New York: Cambridge University Press.

Wolsey, L. (2011). Using extended formulations in practice. *Optima, 85*, 7–9.

Yildiz, B., & Savelsbergh, M. (2019). Optimizing package express operations in China. Optimization Online 6799.

Zabarankin, M., Uryasev, S., & Pardalos, P. (2001). Optimal risk path algorithms. In R. Murphey & P. Pardalos (Eds.), *Cooperative control and optimization* (pp. 271–303). Dordrecht: Kluwer.

Zhu, E., Crainic, T. G., & Gendreau, M. (2014). Scheduled service network design for freight rail transportation. *Operations Research, 62*, 383–400.

Chapter 9
Bilevel Network Design

Martine Labbé and Patrice Marcotte

1 Introduction

The framework of bilevel programming allows the modelling of hierarchical situations where a leader anticipates the rational reaction of a non-cooperating follower whose objective and/or constraints are influenced by the leader's decisions. In the context of network design, this paradigm is especially relevant when the designer of a network does not have a direct control of user flows, who are assigned according to their own logic. In this chapter, we illustrate, through four distinct applications, the modelling and algorithmic issues that characterize bilevel network design problems. Throughout, we assign a broad sense to the term 'network design', meaning any program that involves the determination of variables that impact the structure of a graph or a network.

2 A Primer on Bilevel Programming

Bilevel programs, of which the well-known Stackelberg game is a particular instance, involve a leader and a follower acting in a hierarchical fashion. In our framework, the leader makes decisions embodied into a vector $x^1 \in R^{n_1}$,

M. Labbé
GOM, Université Libre de Bruxelles, Brussels, Belgium

INRIA, Lille, France
e-mail: mlabbe@ulb.ac.be

P. Marcotte (✉)
CIRRELT and DIRO, Université de Montréal, Montréal, QC, Canada
e-mail: marcotte@iro.umontreal.ca

anticipating the reaction $x^2 \in R^{n_2}$ of the follower to its design x^1. The general formulation of a bilevel program is as follows:

$$\begin{aligned} \max_{x^1,x^2} \quad & f_1(x^1, x^2) \\ \text{subject to} \quad & (x^1, x^2) \in X_1 \\ & x^2 \in S_2(x^1)\ , \end{aligned} \tag{9.1}$$

where, for a given design vector x^1, $S_2(x^1)$ is the set of optimal solutions of the follower's problem, i.e.,

$$S_2(x^1) = \arg \min_{y^2 \in X_2(x^1)} f_2(x^1, y^2)\ . \tag{9.2}$$

Under standard assumptions such as compactness or continuity, one can prove the existence of at least one solution to a bilevel program. What makes the problem difficult is that, whenever the lower level problem is not trivial, no closed form solution is available for x_2 as a function of x_1.

If the set $S_2(x^1)$ contains more than one element, the above formulation implies that the leader is free to select the element that maximizes its objective. This is the 'optimistic' case, in contrast with the 'pessimistic' case where the leader assumes that the follower will select the element of $S_2(x^1)$ that minimizes the leader's objective. Throughout this chapter, we assume that the optimistic case prevails.

Now, for notational convenience, we set

$$X_2(x^1) = \{x^2 : (x^1, x^2) \in X_2\}$$

for some set $X_2 \subseteq R^{n_2}$ and record a bilevel program in the following 'vertical' format, which contains all relevant information:

$$\begin{aligned} \max_{x^1} \quad & f_1(x^1, x^2) \\ \text{subject to} \quad & (x^1, x^2) \in X_1 \\ & \\ \min_{x^2} \quad & f_2(x^1, x^2) \\ \text{subject to} \quad & (x^1, x^2) \in X_2\ . \end{aligned} \tag{9.3}$$

It is important to understand that the follower is oblivious to the 'upper level' constraint $(x^1, x^2) \in X_1$. Its enforcement is the sole responsibility of the leader, which must make sure that it is satisfied by the rational reaction of the follower.

We now turn our attention to generic algorithmic frameworks. These are mostly based on reformulations as single-level programs, which can be achieved in a variety of ways. We introduce the three most common ones.

Value Function
In this formulation, lower level optimality is enforced by specifying that, for a given design vector x^1, x^2 must be lower-level feasible, and outperform any alternative solution. This yields the nonconvex semi-infinite program

$$\begin{aligned} \max_{x^1,x^2} \quad & f_1(x^1,x^2) \\ \text{subject to} \quad & (x^1,x^2) \in X_1 \cap X_2 \\ & f_2(x^1,x^2) \le f_2(x^1,y^2) \quad \forall y_2 \in X_2(x^1)\ . \end{aligned} \tag{9.4}$$

First-Order Conditions
If the function f_2 is differentiable and convex with respect to x^2, and if the sets $X_2(x_1)$ are convex for every feasible x_1, one can characterize the set $S_2(x^1)$ via the first-order necessary and sufficient optimality conditions of the lower level programs. This yields the single-level formulation

$$\begin{aligned} \max_{x^1,x^2} \quad & f_1(x^1,x^2) \\ \text{subject to} \quad & (x^1,x^2) \in X_1 \cap X_2 \\ & \langle \nabla_2 f_2(x^1,x^2), x^2 - y^2 \rangle \le 0 \quad \forall y^2 \in X_2(x^1)\ , \end{aligned} \tag{9.5}$$

where ∇_2 represents the gradient of f_2 with respect to its second argument x^2.

Kuhn-Tucker-Based
If the set X_2 is expressed by a set of inequalities

$$X_2 = \{(x^1,x^2) : g_{2i}(x^1,x^2) \le 0,\ i = 1,\ldots,m_2\},$$

for some convex functions g_{2i}, the follower's objective f_2 is convex with respect to x^1, *and* a regularity condition (constraint qualification) holds, then $S_2(x^1)$ can be replaced by the necessary and sufficient Kuhn-Tucker optimality conditions of the lower level program. This yields the single-level formulation

BILKT :

$$
\begin{aligned}
\max_{x^1,x^2} \quad & f_1(x^1,x^2) & \\
\text{subject to} \quad & (x^1,x^2) \in X_1 \cap X_2 & \\
& \nabla_2 f_2(x^1,x^2) + \sum_{i=1}^{m_2} \lambda_i \nabla g_{2i}(x^1,x^2) = 0 & \\
& \lambda_i g_{2i}(x^1,x^2) = 0 & i = 1,\ldots,m_2 \\
& \lambda_i \geq 0 & i = 1,\ldots,m_2\ ,
\end{aligned}
\tag{9.6}
$$

that, in contrast with the preceding two formulations, involves but a finite number of constraints. Note that BILKT, which might be the most frequently used in practice, exhibits the combinatorial nature of the problem through the complementarity between the vector of Lagrange multipliers λ and the inequality constraints defined by the convex functions g_{2i}'s. A convenient shortcut for the complementarity system formed by the last two constraints of BILKT is

$$\lambda \geq 0 \quad \perp \quad g_2(x^1,x^2) \leq 0,$$

which stresses the relationship between bilevel programming and the class of problems known as Mathematical Programs with Equilibrium Constraints, or MPEC's.

Common to all three formulations is that their constraints do not satisfy any constraint qualification generically, hence standard techniques of nonlinear (nonconvex) programming may fail to produce even stationary points. It follows that efficient algorithms must explicitly deal with the twin continuous-combinatorial nature of the problem, and frequently adapt to the specific nature of the application under consideration.

Equilibrium Constraints

In several applications of interest, a population of players reacts to the leader's decision by achieving an equilibrium. This occurs for instance in n-player games, where a Nash equilibrium is reached when no player can increase its objective by acting alone. Another important application involves selfish users that strive for minimum travel time in a congested network. A Wardrop equilibrium of this nonatomic game corresponds to a multicommodity flow vector x such that, for every origin-destination pair, flow is only assigned to paths that are shortest with respect to the delays induced by the leader's decision x^1.

In general, an equilibrium is not naturally the solution of an optimization problem. Rather, the set of lower-level equilibria corresponding to an upper-level vector x^1 can be characterized as the solution of a variational inequality involving a mapping $F_2 : R^{n_2} \to R^{n_2}$, i.e.,

$$S_2(x^1) = \{x^2 \in X_2(x^1) : \langle F_2(x^1, x^2), x^2 - y^2 \rangle \le 0 \quad \forall y^2 \in X_2(x^1)\}.$$

If F_2 happens to be the gradient of some convex function f_2, then the solutions of the above variational inequality are simply the vectors that satisfy the necessary and sufficient first-order optimality conditions of the associated lower level program. The added generality occurs when this is not the case, i.e., for a continuously differentiable mapping F_2, when its Jacobian matrix fails to be symmetric. The equivalent of the Kuhn-Tucker formulation is then obtained by substituting F_2 to ∇_2 in BILKT.

Solution Algorithms

In their generality, bilevel programs are akin to unstructured global optimization problems, and their feasible space may even be disconnected in the presence of upper level constraints involving the vector x^2. Actually, their strong NP-hardness has been proved in the 'simple' case where all functions and constraints are linear.

Most algorithms rely on a single-level reformulation and, for tractability, assume that the lower level problem is relatively easy to solve. Initiated with a relaxed problem where the leader controls both x^1 and x^2, formulations (9.4) and (9.5) suggest a cutting-plane approach that only generates 'optimality' or 'variational' constraints as required. At each iteration, and for a given upper level vector x^1, the most violated constraint is appended to the relaxed problem. For the value function formulation, this amounts to solving the lower level problem

$$\min_{y^2 \in X_2(x^1)} f_2(x^1, y^2)$$

while, for the 'first-order' formulation, one solves

$$\min_{y^2 \in X_2(x^1)} \langle \nabla_2 f_2(x^1, x^2), y^2 \rangle,$$

which reduces to a standard linear program if the lower level constraints are linear. Note that, in both cases, the relaxed problems are highly nonlinear and nonconvex.

Formulation BILKT might be the most interesting from an algorithmic point of view, the main issue being the treatment of its complementarity constraints. Indeed, their presence makes the problem 'irregular', i.e., constraint qualifications that are the basis of convergence analysis are not generically satisfied. This difficulty can be sidestepped by appending a multiple of the complementarity term into the objective. Under weak assumptions, the resulting penalty is 'exact', i.e., a global solution of the penalized problem involving a large but finite multiplier is a global solution of the original bilevel program. Unfortunately, this result might fail to hold for local solutions. Alternatively, one may smooth the complementarity term, for instance

$$\lambda_i g_{2i}(x^1, x^2) \leq \epsilon$$

for some small scalar ϵ. A third approach, which highlights the combinatorial nature of the complementarity constraints, consists in replacing the complementarity constraint $\lambda_i g_{2i}(x^1, x^2) = 0$ by the more tractable triple

$$\begin{aligned} \lambda_i &\leq M u_i \\ g_{2i}(x^1, x^2) &\leq M(1 - u_i) \\ u_i &\in \{0, 1\}, \end{aligned}$$

for some 'large' constant M. When the constraints are linear and the objectives are linear (respectively quadratic), the resulting mixed integer program can be solved to global optimality by a dedicated software. Also noteworthy is that, when the vector of binary variables is fixed, the resulting program possesses a structure (a network structure for instance) that can be exploited.

Another approach, fruitful in nonlinear programming, is to approximate the original program by a related model. Inspired by the trust region framework, one can rely on an approximation involving linear or quadratic functions, which can be solved for a global optimum. Alternatively, one can design an approximation that exploits the problem's structure, as we will observe in the next sections.

Finally, various heuristics can be applied. These can be either of a generic nature, or based on the key features of the problem under investigation. On the generic front, meta-heuristics that have been proposed for combinatorial problems (simulated annealing, tabu search, genetic algorithms, etc.) can exploit the presence of the binary variables u_i's, as well as the relative ease with which the program associated with fixed u-values can be solved.

3 The Continuous Network Design Problem

Consider the problem that consists in improving a subset of links of an urban transportation network, with the aim of minimizing the weighted sum of transportation and investment costs. Since users minimize their individual cost, their objective is not aligned with that of the network builder. More precisely, for a given link improvement vector $z \in Z$, the arc flow vector x achieves a Wardrop equilibrium that satisfies the variational inequality

$$\langle F(z, x), x - y \rangle \leq 0 \quad \forall y \in X,$$

where x is the link-flow vector, F the associated cost (delay) mapping, and X represents the set of feasible arc flows. In this realm, the designer of the network faces the bilevel problem (an MPEC to be precise)

$$\begin{aligned}
\min_{z,x} \quad & \langle F(z,x), x\rangle + c(z) \\
\text{subject to} \quad & z \in Z \\
& x \in X \\
& \langle F(z,x), x-y\rangle \leq 0 \quad \forall y \in X,
\end{aligned} \tag{9.7}$$

where $c(z)$ denotes the cost of implementing the design z.

We now focus on the case where z is only constrained to be nonnegative, and where the delay and investment functions are link-separable, thus are gradient mappings. Letting $\mathscr{A}$ denote the index set of arcs, this yields the bilevel program

$$\begin{aligned}
\min_{z} \quad & \sum_{a\in\mathscr{A}} [F_a(z_a, x_a)x_a + c_a(z_a)] \\
\text{subject to} \quad & z_a \geq 0 \qquad a \in \mathscr{A} \\
\min_{x} \quad & \sum_{a\in\mathscr{A}} \int_0^{x_a} F_a(z_a, t)\, dt \\
\text{subject to} \quad & x \in X\ .
\end{aligned} \tag{9.8}$$

Furthermore, we assume that the link delay functions F_a are of the form

$$F_a(z_a, x_a) = F_a(x_a/z_a),$$

for some nonnegative, convex and increasing functions F_a.

From the theoretical point of view, the problem is NP-hard. However, in the view that the leader and follower's objectives are not at odds (the leader's objective involves total transportation costs, versus marginal costs for the follower), it is tempting to let the leader control both the design and flow variables, yielding a design vector $\bar{z}$ and the corresponding equilibrium flow $\bar{x}$. More specifically, let us consider the system-optimal problem

$$\min_{z\geq 0, x\in X} \sum_{a\in\mathscr{A}} [F_a(x_a/z_a)x_a + c_a(z_a)]. \tag{9.9}$$

For a given flow vector x, the problem becomes separable in z, with $z_a(x_a)$ being the unique solution of the parameterized equation

$$-\,(x_a/z_a)^2 F_a'(x_a/z_a) + c_a'(z_a) = 0, \tag{9.10}$$

obtained by setting to zero the gradient of the leader's objective. One then solves the mathematical program

$$\min_{x \in X} \sum_{a \in \mathscr{A}} [F_a(x_a/z_a(x_a))x_a + c_a(z_a(x_a))], \tag{9.11}$$

to obtain a system-optimal flow $\tilde{x}$, together with $\bar{z} = z(\tilde{x})$. A feasible solution is then easily recovered by setting the flow vector to the equilibrium $\bar{x}$ corresponding to the vector $\bar{z}$. This simple heuristic procedure will be denoted H1.

We now restrict our attention to polynomial delay and investment functions, i.e.,

$$F_a(x_a/z_a) = \alpha_a + \beta_a(x_a/z_a)^p$$

and

$$c_a(z_a) = l_a z_a^m,$$

for some scalars $p \geq 1$, $m \geq 0$ and $l_a \geq 0$. According to these assumptions, the root of Eq. (9.10) can be obtained in closed form. Furthermore, if the cost functions c_a are convex (respectively concave), then problem (9.11) is convex (respectively concave). The concave situation occurs when $m < 1$, and yields an extremal flow solution. In the particular case where functions c_a's are linear, (9.11) reduces to a linear multicommodity flow problem that can be easily solved by shortest path computations.

Another heuristic approach to the problem, denoted H2, consists in iteratively solving Eq. (9.11) for fixed z-vector, and then performing an equilibrium assignment, for fixed design vector z. Whenever this cobweb process converges, it will do so at a couple $(\overline{x}, \overline{z})$ where Eq. (9.10) is satisfied, while $\overline{x}$ is in equilibrium with respect to $\overline{z}$.

A third heuristic strategy (H3) consists in finding a capacity vector that is consistent with the system-optimal flows $\tilde{x}$ provided by Heuristic H1. Algorithms H2 and H3 are actually subsumed by a more general scheme H4 where one solves the single-level program

$$\min_{z \geq 0, x \in X} \sum_{a \in \mathscr{A}} \left[\int_0^{x_a} F_a(t, z_a)\, dt + \xi_a c_a(z_a) \right] \tag{9.12}$$

for some set of nonnegative weights ξ_a's. It is clear that any solution to that program yields equilibrium flows with respect to z. Moreover, it can be shown that there exists a set of weights ξ_a such that an optimal solution of (9.12) is also optimal for the original bilevel program. While determining such weights is theoretically as hard as solving the continuous network design problem in the first place, educated guesses, such as setting ξ to the vector of ones, may prove of interest. Actually, in the view that (9.12) can be solved quickly, one might solve (9.12) over a range of weight vectors with identical values $\bar{\xi}$ for instance. Setting all parameters ξ_a's to $1/(p+1)$ yields H2, while setting them to $p+1$ yields H3.

An interesting feature of the model is that information about worst-case bounds of the heuristics can be derived with respect to key parameters of the problem, namely the respective degrees p and m of the delay and cost polynomials. Let us denote, for a given heuristic H (H1, H2, H3 or H4), by $\rho^{\mathrm{H}}(m, p)$ the worst-case ratio of the cost of the heuristic solution value over that of the unknown optimal value, i.e.,

$$\rho^{\mathrm{H}}(m, p) = \sup \frac{\text{cost of heuristic solution}}{\text{cost of optimal solution}}, \tag{9.13}$$

where the supremum is taken over all possible values of the remaining parameters and network configurations. We have the following results:

- $\lim_{p\to\infty} \rho^{\mathrm{H1}}(p, 1) \geq 2$
- $\rho^{\mathrm{H2}}(p, 1) = p + 1$
- $\rho^{\mathrm{H3}}(p, m) = \dfrac{m(p+1)}{m+p} + \dfrac{p}{(m+p)(p+1)^{m/p}}$
- $1 + \dfrac{p}{\xi(p+1)} \leq \rho^{\mathrm{H4}}(p, m) \leq \dfrac{\xi^{p/p+1}}{(p+1)^{1/p+1}}\left[1 + \dfrac{p}{\xi(p+1)}\right]^2.$

It is worth noting that, for $p = m = 1$, the worst-case bound of heuristic H3 is equal to 5/4, which is less than the price of anarchy (the worst-case ratio between the true delay associated with an equilibrium and the system-optimal delay) for affine latency functions, whose value is 4/3. This bound can also be improved to $49/41 \approx 1.195$ by computing the best solution provided by H3 and a closely related method, a result that was extended to a larger class of delay functions than polynomials. Worst-case results indicate that Heuristic H3 outperforms both H1 and H2. However, computational results tend to show that H1 and H2 perform much better in practice.

A straightforward extension is concerned with the improvement of existing networks. Let us denote by z^0 the vector of initial link capacities. This more general model is subsumed by the 'standard' model if one sets the investment cost to the nondifferentiable function $c_a \max\{0, z_a - z_a^0\}$. While the latter can be approximated as closely as desired by a differentiable function, allowing the implementation of previously described heuristics, the worst-case results do not hold any more.

4 A Competitive Location-Queuing Model

Besides its intrinsic interest, this topic provides an opportunity to analyze algorithmic techniques for addressing complex mathematical programs involving both discrete and continuous variables. In the model considered, a firm makes decisions concerning the location and service levels of facilities, with the aim of attracting the maximum number of customers. At the lower level, customers minimize their individual disutility.

Before providing a mathematical description of the model, we broadly describe the supply-demand setting. Facilities fall into two categories, namely those owned by the leader and its competitors, respectively. Each open facility is characterized by its location vertex and service level, which are the decision variables of the leader firm. Service is exponentially distributed, and queue length limited to a predetermined capacity. Whenever the capacity is exceeded, balking occurs, i.e., customers are denied service. A consequence of this phenomenon is that the model makes sense even if arrival rates exceed service rates. Throughout, we assume that the location and service rates of the competition are fixed and known.

The demand side is characterized by Poisson processes associated with nodes of the network. For given facility locations and service levels, users are assigned to the facilities according to a discrete choice (logit) model where random utilities are linear combinations of travel time to facilities, queueing delays, probability of *not* accessing service, and a random term that makes for unobserved features. In this framework, the objective of the leader is to maximize the number of customers served at its facilities, subject to a budget constraint.

Let us now focus on the assignment part. We denote by d_i the demand rate originating from node i, by λ_j the arrival rate at facility j, and by K_j the capacity of facility j. We assume that users are rational and minimize a disutility expressed as a positive linear combination

$$u_{ij} = t_{ij} + \alpha w_j + \beta p_{K_j} + \varepsilon$$

of travel time t_{ij}, waiting time w_j, probability p_{K_j} of not accessing facility j, plus a random Gumbel term ϵ with cumulative distribution function $\exp(-\exp(x/\theta))$. This yields a closed form expression for the assignment of users to facilities:

$$x_{ij} = d_i \frac{e^{-\theta u_{ij}}}{\sum_{l \in J^*} e^{-\theta u_{il}}}, \tag{9.14}$$

where J^* represents the set of open facilities. Classical queueing theory results then lead to the closed form expressions

$$w_j = \frac{1}{\mu_j}\left(K_j + \frac{K_j}{(\lambda_j/\mu_j)^{K_j} - 1} - \frac{1}{(\lambda_j/\mu_j) - 1}\right)$$

and

$$p_{K_j} = \frac{\lambda_j}{\mu_j} \cdot \frac{1 - (\lambda_j/\mu_j)}{1 - (\lambda_j/\mu_j)^{K_j+1}}.$$

If the process intensity λ_j/μ_j is equal to 1, the above expressions are replaced by their limits, which are finite.

Let J_1 denote the index set of the leader's facilities. The aim of the leader is to maximize the number of customers that access its facilities, i.e.,

$$\sum_{j\in J_1} \lambda_j(1 - p_{K_j}),$$

through the control of the service rates μ_j ($j \in J_1$) and location decisions, represented by binary variables y_j set to 1 if a facility is located at node j of the network, and to 0 otherwise. Denoting by c_j^f the fixed cost of opening a facility at node j, by c_j the cost of providing a unit of service level at node j, and by B the total budget constraint, the problem can be formulated as the mathematical program

$$\begin{array}{lll}
\text{LEADER:} & \max\limits_{y,\mu} & \sum\limits_{j\in J_1} \lambda_j(1 - p_{K_j}) & \\
& \text{subject to} & \sum\limits_{j\in J_1} c_j^f y_j + \sum\limits_{j\in J_1} c_j \mu_j \le B & \\
& & \mu_j \le M y_j & j \in J_1 \\
& & y_j \in \{0,1\} & j \in J_1 \\
\text{USERS:} & & x_{ij} = d_i y_j \dfrac{e^{-\theta\left(t_{ij}+\alpha w_j+\beta p_{K_j}\right)}}{\sum\limits_{l\in J^*} e^{-\theta\left(t_{il}+\alpha w_l+\beta p_{K_l}\right)}} & i \in I, j \in J \\
& & \lambda_j = \sum\limits_{i\in I} x_{ij} & j \in J \\
& & w_j = \dfrac{1}{\mu_j}\left(K_j + \dfrac{K_j}{(\lambda_j/\mu_j)^{K_j} - 1} - \dfrac{1}{(\lambda_j/\mu_j) - 1}\right) & j \in J \\
& & p_{K_j} = (\lambda_j/\mu_j)^{K_j} \dfrac{1 - (\lambda_j/\mu_j)}{1 - (\lambda_j/\mu_j)^{K_j+1}} & j \in J
\end{array}$$

where the main decision variables y and μ have been emphasized under the 'max' operator, and M is a suitably large 'big-M' constant. In the limiting case $\theta = \infty$, the first user equation reduces to the complementarity system (Wardrop principle)

$$t_{ij} + \alpha w_j(x,\mu) + \beta p_{Kj}(x,\mu) \begin{cases} = \xi_i, \text{ if } x_{ij} > 0 \\ \\ \ge \xi_i, \text{ if } x_{ij} = 0. \end{cases} \tag{9.15}$$

While the above looks like a standard optimization program, its main difficulty resides in the highly nonlinear constraints that define the user flows. Indeed, since the quantities w_j and p_{K_j} both depend on x_{ij}, it follows that the first user constraint is actually a fixed point equation that is not trivial to solve in its own right. To cast the problem into a more convenient framework, we first note that any solution of the following optimization program, which is convex in the user variables, yields an equilibrium solution:

$$\begin{aligned}
\min_x \quad & \sum_{i\in I}\sum_{j\in J^*}\left(\frac{1}{\theta}x_{ij}\ln x_{ij} + t_{ij}x_{ij}\right) + \alpha\sum_{j\in J^*}\int_0^{\lambda_j} w_j(\lambda,\mu_j)d\lambda \\
& \qquad\qquad + \beta\sum_{j\in J^*}\int_0^{\lambda_j} p_{Kj}(\lambda,\mu_j)d\lambda \qquad (9.16)\\
\text{subject to} \quad & \sum_{j\in J^*} x_{ij} = d_i \qquad i\in I\\
& x_{ij}\ge 0 \qquad i\in I,\ j\in J^*\\
& \lambda_j = \sum_{i\in I} x_{ij} \qquad j\in J^*.
\end{aligned}$$

When $\theta = \infty$, the first term simply disappears, and users are assigned to shortest paths.

Upon replacing the fixed point equation by the above program, one obtains a bilevel program involving a structure that can be exploited algorithmically. For instance, the first term of the lower level objective is convex, while the second is jointly pseudo-convex in λ and μ. Furthermore, in the important case when no balking occurs ($K_j = \infty$ for all leader facilities), the third term is jointly convex in λ and μ.

Let us first consider the no-balking situation. In this case, $p_{K_j} = 0$ and, in order that the problem be meaningful, the total service rate must exceed total demand. This can be already satisfied by the competing facilities, or enforced through additional bound constraints on service levels. This suggests the following strategy for solving the bilevel location problem:

- Approximate the lower level objective by a (convex) piecewise linear function.
- Express the approximate lower level program as a linear program.
- Replace the linear program by its primal-dual optimality conditions. This yields linear constraints, with the exception of primal-dual complementarity.
- With the help of binary variables, linearize the complementarity constraints to obtain a mixed integer program (MIP).
- Let y^* and μ^* be the partial solution of the MIP.
- Compute the equilibrium flows x^* compatible with y^* and μ^*.

One can show that, as the mesh of the piecewise linear approximation decreases, the solution of the MIP converges to a solution of the original problem. More important, it was observed that a coarse mesh was actually sufficient to yield high quality solutions. If the parameters K_j are finite, the situation becomes more complex, and one has to introduce additional binary variables to linearize both the upper and lower level objectives. Nevertheless, the same algorithmic strategy can still be implemented.

Since the curse of dimensionality hits very early for such complex bilevel programs, it warrants looking for alternative algorithmic strategies. One such approach consists, as has been proposed for the continuous version of the network design problem considered in the previous section, to replace the bilevel program by a single level proxy. In this regard, an equivalent to H1 would be to let the leader control all decision variables. However, this yields a poor approximation, whose trivial solution is to first build a single facility (the one with least fixed cost), to spend the residual budget on its service level, and to direct the maximal amount of customers to this facility, without regards for the competition. A better alternative is, similar to H2, to minimize a proxy program that yields equilibrium flow patterns, for instance

$$
\begin{aligned}
\min_{y,\mu,x} \quad & \sum_{i\in I}\sum_{j\in J^*}\left(\frac{1}{\theta}x_{ij}\ln x_{ij}+t_{ij}x_{ij}\right)+\alpha\sum_{j\in J^*}\int_0^{\lambda_j} w_j(\lambda,\mu_j)d\lambda \\
& \qquad +\beta\sum_{j\in J^*}\int_0^{\lambda_j} p_{Kj}(\lambda,\mu_j)d\lambda \\
\text{s.t.} \quad & \sum_{j\in J} x_{ij}=d_i && i\in I \\
& \sum_{j\in J_1} c_j^f y_j+\sum_{j\in J_1} c_j\mu_j\le B \\
& \lambda_j=\sum_{i\in I} x_{ij} && j\in J \\
& y_j\in\{0,1\} && j\in J \\
& x_{ij}\ge 0 && i\in I, j\in J.
\end{aligned}
\tag{9.17}
$$

In the spirit of H4 introduced in the previous section, the objective can be generalized by adding a term that depends on the service level vector μ, yielding:

$$
\begin{aligned}
\min_{y,\mu,x} \quad & \sum_{i\in I}\sum_{j\in J^*}\left(\frac{1}{\theta}x_{ij}\ln(x_{ij})+t_{ij}x_{ij}\right)+\alpha\sum_{j\in J^*}\int_0^{\lambda_j} w_j(\lambda,\mu_j)d\lambda \\
& \qquad +\beta\sum_{j\in J^*}\int_0^{\lambda_j} p_{Kj}(\lambda,\mu_j)d\lambda+\sum_{j\in J_1}\xi_j\mu_j,
\end{aligned}
\tag{9.18}
$$

subject to the same constraints. Note that, in the important situation where $K=\infty$ (no balking), this yields a simple convex program. Otherwise, a nonconvex piecewise linear approximation of its objective results in a mixed integer formulation that can be solved to global optimality.

While it can be proved that there exists an assignment of the ξ_j variables that yields an optimal solution of the original problem, finding such assignment is theoretically as difficult as solving the original problem. Nevertheless, based on the intuition that it makes sense for the leader to favor facilities located close to high demand nodes, the educated guess

$$\xi_j = d_i / t_{ij}$$

has been proposed. If a demand node coincides with a facility location, a small number can be added to the denominator, to avoid dividing by zero. The formula can also be enhanced to take into account the fixed costs c_j^f, for instance,

$$\xi_j = -d_i / (c_j^f t_{ij}).$$

5 Network Pricing

Pricing offers a rich area for applications of bilevel programming. This section discusses such problems involving an underlying network structure. More precisely, we consider the problem that consists in setting tolls on a subset of arcs of a multicommodity transportation network, with the aim of maximizing profit. In this context, the toll manager anticipates the flows that result from its toll policy, i.e., users are assigned to paths that minimize their respective disutility, taking into account constant transportation costs, as well as out-of-pocket costs.

As an example, consider the network depicted in Fig. 9.1, that involves two toll arcs (dotted) and a single user that travels from node 1 to node 5. The cost (equal to 22) of the toll free path from node 1 to node 5 provides an upper bound on the maximum amount the user agrees to pay for its trip. Further the transportation cost is at least 6, which corresponds to the cost of a shortest path when both tolls are set to zero. This yields an upper bound of $22 - 6 = 16$ on the revenue. Interestingly, this bound cannot be reached since the optimal solution is obtained by setting the toll on arc (2, 3) to 5 and the toll on arc (4, 5) to 10.

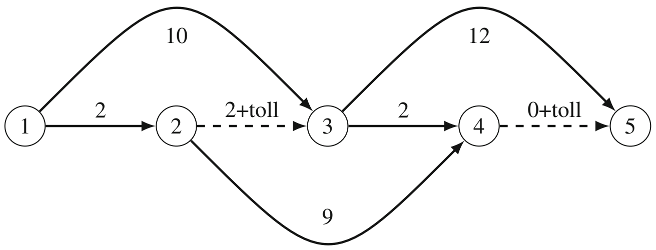


Fig. 9.1 A toll pricing network and its arc costs

In general, the multicommodity transportation network is characterized by a graph $\mathscr{G}$ with node set $\mathscr{N}$ and arc set $\mathscr{A}$. Each arc $a \in \mathscr{A}$ is endowed with a cost c_a. The set of arcs $\mathscr{A}$ is partitioned into two subsets $\mathscr{A}_1$ and $\mathscr{A}_2$, the former containing the arcs controlled by the toll manager and the latter the so-called 'toll free' arcs. The commodities $k \in \mathscr{K}$ represent groups of users willing to travel from the same origin $o^k \in \mathscr{N}$ to the same destination $d^k \in \mathscr{N}$. For commodity k, its demand is denoted by η^k and its nonnegative transportation cost on arc a by c_a^k.

The Network Pricing Problem (NPP) can be formulated as the following bilevel program that involves bilinear objectives and linear constraints:

NPP:

$$\begin{aligned}
\max_t \quad & \sum_{k\in\mathscr{K}} \eta^k \sum_{a\in\mathscr{A}_1} t_a x_a^k \\
\text{subject to} \quad & t \in T \\
\min_x \quad & \sum_{k\in\mathscr{K}} \Big(\sum_{a\in\mathscr{A}_1} (c_a^k + t_a) x_a^k + \sum_{a\in\mathscr{A}_2} c_a^k x_a^k \Big) \\
\text{subject to} \quad & x^k \in X^k \qquad\qquad k \in \mathscr{K},
\end{aligned} \tag{9.19}$$

where X^k denotes the polyhedron of feasible flows for commodity k, and T imposes constraints, such as bounds, on the set of feasible tolls. In order that the profit be bounded, there must exist at least one toll free path, i.e., containing only arcs belonging to $\mathscr{A}_2$, for each origin-destination pair (commodity). In other words, the removal of toll arcs must leave all origin-destination pairs connected.

In our variant of the network pricing problem, we assume that the set T consists of all nonnegative toll values. Note that variants involving unrestricted or upper bounded tolls have also been considered in the literature. In addition, X^k represents the set of feasible paths associated with commodity k, that is characterized by the following flow conservation constraints on variables x_a^k:

$$\sum_{a\in\delta(i)^+} x_a^k - \sum_{a\in\delta(i)^-} x_a^k = b_i^k \qquad i \in \mathscr{N} \tag{9.20}$$

$$x_a^k \geq 0 \qquad a \in \mathscr{A}, \tag{9.21}$$

where $b_i^k = +1$ if $i = o^k$, $b_i^k = -1$ if $i = d^k$, and $b_i^k = 0$ otherwise. Further, $\delta(i)^+$ (resp. $\delta(i)^-$) denotes the set of arcs having node i (resp. j) as tail (resp. head).

From the complexity point of view, it has been shown that NPP is strongly NP-hard, and actually APX-hard, even in the single commodity case. Alternatively, consider an instance of NPP that involves but one toll arc (see Fig. 9.2). A commodity k uses the toll arc only if its value is not larger than the difference, say M^k, between the value of the shortest path not using the toll arc and the value of the shortest path with the toll set to zero. It is easy to see that the optimal toll value will

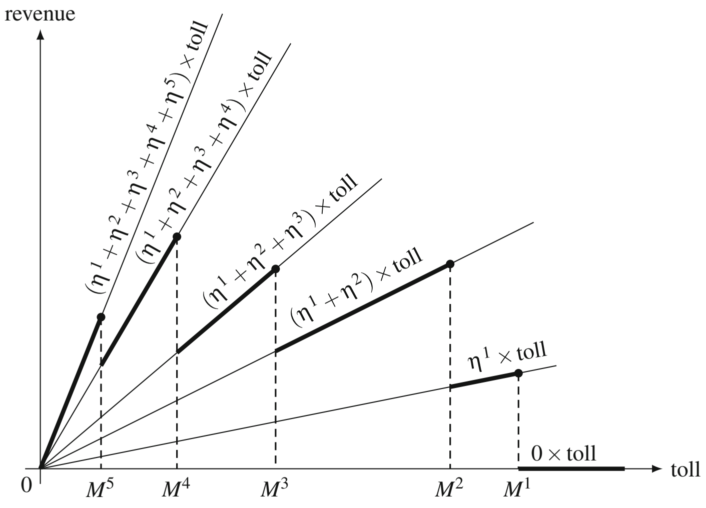


Fig. 9.2 Revenue function (in bold) for a single toll arc and 5 origin-destination pairs ($|\mathscr{K}| = 5$). The function is piecewise linear and continuous from the left. A local maximum is achieved at each 'critical' value M^k. In this instance, the optimum revenue is achieved when the toll is equal to M^4

be equal to some M^k, which are in polynomial number. If commodities are ranked in decreasing values of the M^k, the corresponding revenue is obtained, for each such M^k, by multiplying M^k by $\sum_{k' \le k} \eta^{k'}$. This yields a polynomial algorithm.

A closer look at the bilevel formulation of NPP shows that the lower level problem decomposes into shortest path problems, one for each commodity, that can be formulated as linear programs, since costs have been assumed nonnegative. To obtain a single level reformulation, one may replace the lower level problem of each commodity by its primal and dual constraints, together with a constraint imposing the equality of the primal and dual objective functions. This approach is the linear programming version of the Kuhn-Tucker-based formulation (see Sect. 2), and yields the following mixed integer linear formulation, in which the dual variables λ_i^k correspond to constraints (9.20):

MILP:

$$\max_{t,x,\lambda} \quad \sum_{k\in\mathscr{K}} \eta^k \sum_{a\in\mathscr{A}_1} t_a x_a^k$$

$$\text{subject to} \quad t_a \ge 0 \qquad a \in \mathscr{A}_1$$

$$\sum_{a\in\delta(i)^+} x_a^k - \sum_{a\in\delta(i)^-} x_a^k = b_i^k \qquad i \in \mathscr{N}, k \in \mathscr{K}$$

$$x_a^k \ge 0 \qquad a \in \mathscr{A}$$

$$\begin{aligned}
&\lambda_i^k - \lambda_j^k \le c_a^k + t_a && a = (i, j) \in \mathscr{A}_1, k \in \mathscr{K} \\
&\lambda_i^k - \lambda_j^k \le c_a^k && a = (i, j) \in \mathscr{A}_2, k \in \mathscr{K} \\
&\sum_{a \in \mathscr{A}_1} (c_a^k + t_a) x_a^k + \sum_{a \in \mathscr{A}_2} c_a^k x_a^k = \lambda_{o^k}^k - \lambda_{d^k}^k && k \in \mathscr{K}.
\end{aligned}$$

This model involves bilinear terms $t_a x_a^k$ in the objective and in some constraints. These terms can be linearized by using the standard technique consisting in substituting them by new variables, say p_a^k and appending the following new constraints, for all a in $\in \mathscr{A}_1$ and k in $\mathscr{K}$:

$$\begin{aligned}
p^k &\le M_a^k x_a^k \\
t_a - p_a^k &\le N_a (1 - x_a^k) \\
p_a^k &\le t_a \qquad (9.22) \\
t_a^k &\ge 0 \\
x_a^k &\in \{0, 1\}.
\end{aligned}$$

Note that binary constraints regarding variables x_a^k are required in order to ensure the validity of the linearization. This is consistent with the fact that these variables characterize shortest paths. The MILP formulation can be improved by tightening the values of the big-M constants M_a^k and N_a, yielding an improved linear relaxation.

Next, one may replace the original graph by an equivalent and usually smaller one, based on the observation that shortest paths are composed of alternating shortest subpaths containing only toll arcs or only toll free arcs. Finally, criteria have been proposed to eliminate some provably irrelevant variables x_a^k.

Despite these improvements, the size of instances for which solvers can address MILP is limited, thus prompting the development of (meta) heuristics, many of which iterate between path and toll phases. In this context, it is instructive to study the so-called inverse problem which consists in determining tolls maximizing the leader's revenue, under the assumption that the followers' paths are known. MILP then boils down to a network-structured linear program involving only toll variables. When there is only one follower (a single origin or a single destination), the inverse problem reduces to a shortest path problem on a modified graph.

The **Clique Pricing Problem** (CPP) is an NP-hard special case of NPP in which each commodity uses at most one toll arc. This framework fits the realm of a highway where the toll depends only on the entry and exit nodes used by the commodities, and re-entry is forbidden. The underlying network is then based on the complete graph whose vertices are the highway entry and exit nodes, and whose arcs $a \in \mathscr{A}_1$ represent all subpaths of the original highway path (see Fig. 9.3).

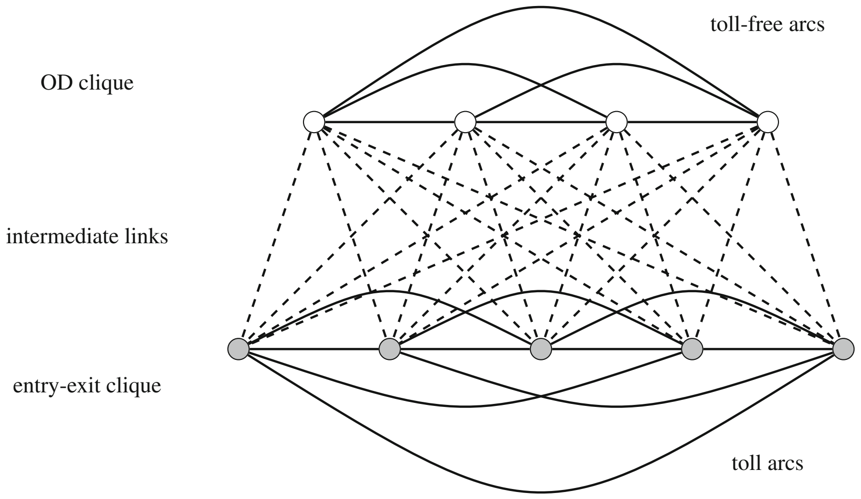


Fig. 9.3 Clique representation. All arcs are double-ended

The arcs of the OD clique represent shortest toll free paths between origins and destinations. For origin-destination pair (commodity) k, their cost is denoted by c^k_{od}. The coefficient c^k_a of an arc $a \in \mathscr{A}_1$ is the sum of three terms: the minimum cost from o^k to the tail node of a, the minimum cost from the head node of a to d^k (dashes arcs), and the cost for traversing arc a. It follows that, for each commodity, the only relevant arc of $\mathscr{A}_2$ is (o^k, d^k). The k-th objective function of the lower level problem (9.19) then simplifies to

$$\min_{x \in X^k} \quad \sum_{a \in \mathscr{A}_1} (c^k_a + t_a)x^k_a + c^k_{od}x^k_{od}$$

and the set X^k to

$$\sum_{a \in \mathscr{A}_1} x^k_a + x^k_{od} = 1 \quad k \in \mathscr{K}$$

$$x^k_a \geq 0 \quad a \in \mathscr{A}_1 \cup \mathscr{A}_2, k \in \mathscr{K}.$$

This simple structure of X^k allows to reformulate CPP as a single level problem by stating explicitly that the objective function of the k-th commodity is less than or equal to the total cost corresponding to each arc a of the highway, i.e.,

$$\sum_{a \in \mathscr{A}_1} (c^k_a + t_a)x^k_a + c^k_{od}x^k_{od} \leq c^k_b + t_b \quad b \in \mathscr{A}_1, k \in \mathscr{K}$$

$$\sum_{a\in\mathscr{A}_1}(c_a^k + t_a)x_a^k + c_{od}^k x_{od}^k \le c_{od}^k \qquad k \in \mathscr{K}, \tag{9.23}$$

which corresponds to the value function formulation (9.4). A mixed binary linear formulation is obtained by substituting the bilinear terms $t_a x_a^k$ with variables p_a^k, and appending constraints (9.22) to the model.

For all $b \in \mathscr{A}_1$ and for all $\mathscr{S} \subset \mathscr{A}_1 \setminus \{b\}$, the inequality

$$\sum_{a\in\mathscr{A}_1}(c_a^k x_a^k + p_a^k) + c_{od}^k x_{od}^k \le t_b + c_b^k + \sum_{a\in\mathscr{S}}(p_a^k + (c_a^k - c_b^k)x_a^k)$$

holds. Indeed, remembering that $x_a^k = 0$ implies that $p_a^k = 0$, the inequality reduces to (9.23) if $\sum_{a\in\mathscr{S}} x_a^k = 0$. If not, there must be exactly one arc, say b', such that $x_{b'}^k = 1$, and the inequality reads $c_{b'}^k + p_{b'}^k \le t_b + p_{b'}^k + c_{b'}^k$, which is obviously satisfied.

The inclusion of these inequalities in MILP provides an ideal formulation in the (polynomial) case of a single commodity, i.e., its linear relaxation yields an optimal solution. In the general case involving several commodities, the inequalities strengthen significantly the formulation and can be separated in polynomial time.

Interestingly, CPP is equivalent to the classical product pricing problem in economics. In this problem the price of products $a \in \mathscr{A}_1$ must be determined in order to maximize revenue. Each customer $k \in \mathscr{K}$ is endowed with a reservation price R_a^k for each product a and buys the product a that maximizes $R_a^k - t_a$, provided that this 'utility' is nonnegative. The equivalence of the two problems is obtained by setting $R_a^k = c_{od}^k - c_a^k$.

When products constitute bundles or subsets of items it may be sensible to impose monotonicity and triangle inequality constraints. Monotonicity constraints specify that if the item set of product a is included in the item set product b, then $p_a \le p_b$, and triangle inequality constraints stipulate that $p_c \le p_a + p_b$ if the item set of c is the union of the item sets of a and b, which are disjoint. These type of constraints make particular sense for the clique pricing problem since a toll arc represents a subpath of the highway, i.e., a particular subset of its arcs. The inclusion of monotonicity and triangle inequalities into CPP results in a significant increase in the integrality gap, and consequently makes more challenging its numerical resolution.

We conclude this section by mentioning other variants of NPP. The first three address more complex models of user behaviour. The first variant integrates **elastic demand** at the lower level, which allows to dispense with the existence of toll free paths. In the **multiclass model**, the relative utility of delays and out-of-pocket costs varies across the population while, in the **discrete choice model**, a random term is added to the costs c_a and d_a. In those two cases, algorithms that rely on approximations that can be formulated as mixed integer programs have been proposed.

The fourth variant involves lower level **transshipment problem**. In the modified formulation of NPP, the commodity index k is dropped, and link capacities are introduced. Unfortunately, since it cannot be assumed that origin-destination flows are assigned to a unique path, the bilinear $t_a x_a$ cannot be linearized as previously, and one may have to resort to unary or binary expansions of the flow variables x_a.

The fifth variant arises when the **design of the network** must be decided together with the tolls. In this case, new binary variables y_a stating whether an arc a is installed or not are introduced, and the upper level problem becomes

$$\max_{y,t} \quad \sum_{k\in\mathscr{K}} \eta^k \sum_{a\in\mathscr{A}_1} t_a x_a^k - \sum_{a\in\mathscr{A}_1} f_a y_a$$

where f_a represent the fixed cost for opening arc a. For the sake of consistency, the constraints

$$x_a^k \leq y_a \quad a \in \mathscr{A}_1, k \in \mathscr{K} \tag{9.24}$$

must be appended to (9.20) and (9.21) in the definition of the sets X^k.

This joint design and pricing problem presents an unusual property for a bilevel optimization problem: constraints (9.24) can be moved to the first level. Intuitively, this is due to the fact that it is in the leader's interest to adjust its price levels such that the capacity constraints are never active. In technical terms, the dual variable of a capacity constraint will always be null, even if the constraint is tight. This property of the program allows to significantly reduce the number of dual variables in the formulation, and preserves the shortest path structure of the lower level problem.

6 Bilevel Network Interdiction

Interdiction games play an important role in military and drug enforcement applications. In both cases, the goal is to disrupt elements of a transportation network in order to reduce as much as possible the enemy's movements on the network. A network interdiction problem (NIP) involves two actors whose goals are antagonistic. The interdictor or attacker acts first by disrupting some elements of the network. Next, the enemy or defender reacts after having observed the functioning state of the network. In this bilevel setting, the interdictor (leader) anticipates the rational reaction of the defender (follower).

If the game is played only once and the enemy has perfect knowledge of the interdictor's action, both players have no advantage in randomizing their actions (strategies). In other contexts, for instance if the game is repeated or the follower is not perfectly rational, it makes sense to consider mixed strategies, i.e., strategies that assign a probability to each feasible action. This is especially relevant in drug enforcement and terror prevention applications. From now on, we focus on interdiction problems that involve either shortest path or maximum flow lower level problems. The structure of the first variant of path interdiction, which consists in maximizing the shortest path length, makes randomization irrelevant. This is not the case of the second variant, where it may be in the leader's interest to implement mixed strategies.

Throughout, we will make use of the following notation, which is common to the three applications presented in the remainder of this section: given a graph $\mathscr{G} = (\mathscr{N}, \mathscr{A})$, we define the set of feasible interdictions as

$$Y = \{y_a \in \{0,1\}, a \in \mathscr{A} : \sum_{a \in \mathscr{A}} f_a y_a \leq B\},$$

where f_a denotes the interdiction cost and B the interdictor's budget.

The Shortest Path Interdiction Problem (SPIP)

In SPIP, one associates the traversal cost c_a, as well the additional cost d_a of traversing an interdicted arc. Within its budget limit, the leader selects a subset of interdicted arcs with the aim of maximizing the length of the follower's shortest path from source s to sink t consistent with the leader's design. The formulation of SPIP is:

$$\begin{aligned}
\max_{y \in Y} \quad & \min_{x} \quad \sum_{a \in \mathscr{A}} (c_a + d_a y_a) x_a && \\
& \text{subject to} \quad \sum_{a \in \delta(i)^+} x_a - \sum_{a \in \delta(i)^-} x_a = b_i && i \in \mathscr{N} \qquad (9.25)\\
& \qquad\qquad x_a \geq 0 && a \in \mathscr{A} \qquad (9.26)\\
& \qquad\qquad x_a \in \{0,1\} && a \in \mathscr{A}, \qquad (9.27)
\end{aligned}$$

where the constant b_i is equal to $+1$ if $n = s$, -1 if $n = t$, and 0 otherwise. This 'max-min' formulation is clearly a particular case of a bilevel program.

The set of feasible solutions of the lower level does not depend on the first level decisions. This can be exploited to develop a single level reformulation based on the value functions (see Sect. 2) as well as an iterative method to solve SPIP. Let X represent the set of binary vectors corresponding to simple paths from s to t and let $\overline{X} \subseteq X$. We define a master problem associated to $\overline{X}$ as

MP($\overline{X}$):

$$\begin{aligned}
\max_{y \in Y} \quad & z \\
\text{subject to} \quad & z \leq \sum_{a \in \mathscr{A}} (c_a \bar{x}_a + d_a \bar{x}_a y_a), \quad \bar{x} \in \overline{X}.
\end{aligned}$$

Problem MP(X) is a valid formulation for SPIP and MP($\bar{X}$) is a relaxation for any $\overline{X} \subset X$ that provides an upper bound on the optimal value of SPIP. On the other hand, given a leader's solution $\overline{x} \in X$, solving the follower's shortest path problem yields a lower bound on the optimal value of SPIP. Further, if this path has a smaller value than the master problem for $\bar{x}$, this new path should be added to $\overline{X}$. Hence, SPIP can be solved by alternating between solving the master and the follower's problem.

Upon replacing the lower level linear program of SPIP by its dual, one obtains the single level formulation

$$\max_{y \in Y, \pi} \quad \pi_t - \pi_s$$

$$\text{subject to} \quad \pi_j - \pi_i - d_a y_a \leq c_a \quad a = (i, j) \in \mathscr{A}.$$

It is interesting to note that the iterative procedure described above to solve SPIP amounts to applying Benders decomposition to this last formulation.

A Path Interdiction Problem Involving Mixed Strategies (PIMS)

In this problem, the leader maximizes the probability of catching a follower (evader) whose objective is to travel from a given origin s to a given destination t. The pure strategies of the evader consist in all simple paths from s to t. These can be represented by binary vectors x satisfying (9.25) and (9.26). A mixed strategy for the evader is a convex combination of such vectors. Given that the linear system defined by (9.25) and (9.26) is totally unimodular, all extreme points of the associated polytope are integer-valued and correspond precisely to these simple paths. It follows that the set of the evader's mixed strategies is $X = \{x_a, a \in \mathscr{A} :$ (9.25) and (9.26)$\}$.

The pure strategies of the interdictor consist in subsets of arcs to inspect. If an arc a is inspected by the interdictor and belongs to the path used by the evader, the interdictor catches it with probability p_a. The goal of the inspector is to determine the mixed strategy that maximizes the probability to catch the evader. A mixed strategy for the inspector consists in assigning a probability to each pure strategy.

Note that, in this game, players possess incomplete information regarding their competitor's behaviour. They know their competitor mixed strategy values but, on a given day, ignore which pure strategy will be played.

First, assume that the interdictor can inspect only one arc a at a time, and that p_a represents the probability that it detects the evader if the latter traverses this arc. Its set of mixed strategies is $Y = \{y_a, a \in \mathscr{A} : \sum_{a \in \mathscr{A}} y_a = 1, y_a \geq 0\}$.

The value of the game is given by the probability that the interdictor detects the evader, and PIMS can be formulated as the bilevel program:

$$\min_{x \in X} \max_{y \in Y} \sum_{a \in \mathscr{A}} p_a x_a y_a.$$

The game is actually a zero-sum matrix game and, according to von Neumann's theorem, an equivalent program is obtained by permuting the min and max operators. Upon introduction of v, the value of the game, and taking the dual of the inner problem, it can alternatively be formulated as

$$\begin{aligned} &\min_{v,x} v \\ &\text{subject to} \quad (9.25), (9.26) \\ &\qquad\qquad v \geq p_a x_a, \quad a \in \mathscr{A}, \end{aligned}$$

which means that the follower searches for a path minimizing its maximum regret given by the arc with the highest probability of being caught.

In our context, the above problem possesses an interesting structure. Consider the maximum value ϕ^* of a flow ϕ^*_a, $a \in \mathscr{A}$ in the graph $\mathscr{G}$ with capacities $1/p_a$. By the max flow-min cut theorem of Ford and Fulkerson, $\phi^* = \sum_{a\in\mathscr{C}^*} 1/p_a$, where $\mathscr{C}^*$ is a minimum capacity cut separating s from t. Given that this cut remains minimum if all capacities are multiplied by v, there exists a flow of value 1 in $\mathscr{G}$ as long as v exceeds $1/\phi^*$. Hence the optimal solution of PIMS is $v^* = 1/\phi^*$ and $x^*_a = \phi^*_a/\phi^*$. To retrieve the mixed strategy, i.e., the probability associated to each $s - t$ path, it suffices to use any (polynomial) algorithm that decomposes a flow into a sum of flows on simple $s - t$ paths and cycles. There will be no cycles in our application.

Intuitively, the optimal mixed strategy of the interdictor will only assign positive probabilities to the arcs of the minimum cut $\mathscr{C}^*$. One can verify that the solution $y^*_a = 1/(p_a\phi^*)$ for $a \in \mathscr{C}^*$, and $y_a = 0$ otherwise, belongs to Y. Thus, $\sum_{a\in\mathscr{A}} p^*_a x_a y^*_a = v^* = 1/\phi^*$, and this solution is optimal.

A natural extension of PIMS involves m independent interdictors. In a pure strategy, several interdictors can be assigned to the same arc. Then, the optimal solution is obtained by multiplying each y^*_a by m, and the evader's solution x^* is unchanged. If, on the contrary, at most one interdictor can be assigned to an arc in a pure strategy, then the set Y must be modified, and the above interpretation in terms of maximum flow does not hold anymore. However, the problem remains polynomial and can be solved as a linear program.

The Maximum Flow Interdiction Problem (MFIP)

In MFIP, the leader wants to minimize the maximum flow from a source s to a sink t in a capacitated network. Assuming that arc $\bar{a} = (s, t)$ has an unlimited capacity and that the other arcs $a \neq \bar{a}$ have capacity u_a, the flow from s to t can be seen as a circulation, leading to the following formulation of MFIP:

$$\begin{aligned} \min_{y\in Y} \qquad & \max_{x} \quad x_{\bar{a}} \\ & \text{subject to} \quad \sum_{a\in\delta(i)^+} x_a - \sum_{a\in\delta(i)^-} x_a = 0, && i \in \mathscr{N} \\ & \qquad\qquad x_a \leq u_a(1 - y_a) && a \in \mathscr{A} \setminus \{\bar{a}\} \end{aligned}$$

$$x_a^k \geq 0 \qquad a \in \mathscr{A}.$$

In contrast with SPIP, the feasible solutions of MFIP depend on the upper level variables, hence the Benders decomposition framework does not apply.

Strong duality for the second level problem can again be used to derive a single mixed integer nonlinear formulation:

$$\min_{y \in Y, \alpha, \beta} \sum_{a \in \mathscr{A} \setminus \{\bar{a}\}} u_a (1 - y_a) \beta_a$$

$$\text{subject to} \quad \alpha_i - \alpha_j + \beta_a \geq 0 \qquad a = (i, j) \in \mathscr{A} \setminus \{\bar{a}\} \tag{9.28}$$

$$\alpha_t - \alpha_s \geq 1 \tag{9.29}$$

$$\alpha_i, \beta_a \in \{0, 1\} \qquad i \in \mathscr{N}, a \in \mathscr{A} \setminus \{\bar{a}\}, \tag{9.30}$$

whose objective is to separate s from t by a cut whose capacity is minimal with respect to the interdicted arcs. Since there exists an optimal solution in which all interdicted arcs belong to the cut, one can rewrite MFIP as

$$\min_{y \in Y, \alpha, \beta} \sum_{a \in \mathscr{A} \setminus \{\bar{a}\}} u_a \beta_a - \sum_{a \in \mathscr{A} \setminus \{\bar{a}\}} u_a y_a$$

$$\text{subject to} \quad (9.28)\text{-}(9.30)$$

$$y_a \leq \beta_a \qquad a \in \mathscr{A} \setminus \{\bar{a}\},$$

which is amenable to a branch-and-cut approach. Indeed, the knapsack structure of the set Y, together with the cut configuration can be exploited to derive strong valid inequalities.

7 Bibliographical Notes

Bilevel programming is now almost a mature field, with a few monographs and surveys devoted to the area: Luo et al. (1996); Dempe (2002, 2003); Colson et al. (2007). Among the various methods that do not guarantee a global solution, we mention the approximation scheme of Colson et al. (2005), as well as the book edited by Talbi (2013), where metaheuristics are addressed.

Introduced by Abdulaal and LeBlanc (1979), who proposed for its solution exploratory methods, the continuous version of the network design problem has been further studied by Marcotte (1986), who analyzed the worst-case behaviour of a class of heuristics. In Gairing et al. (2017), the authors proved NP-hardship of the problem, and refined a worst-case bound obtained in Marcotte (1986). The model that considers improvements to an existing network has been studied by Marcotte and Marquis (1992). One of the first mentions of the discrete variant of the problem is due to LeBlanc and Boyce (1986), who addressed it within a branch-and-bound framework.

Among the limited literature devoted to location within a user-optimized environment, we mention the works of Marianov et al. (2008), Zhang et al. (2010) and Abouee-Mehrizi et al. (2011), whose models are closely related to the one presented in this chapter (Dan and Marcotte 2019). It is also interesting to relate these models to the literature on rational queues, that studies the performance of queueing systems when the agents involved do not (fully) cooperate. A comprehensive survey concerning this topic is available in the monograph by Hassin (2016).

The Network Pricing Problem was first considered by Labbé et al. (1998), who proposed single level reformulations and discussed polynomial special cases. Roch et al. (2005) and Joret (2011) studied the complexity of NPP, while (Dewez et al. 2008) proposed valid inequalities and a branch-and-cut algorithm, determining in the process tight values for the big-M constants that enter formulation MILP. Van Hoesel (2008) developed a branch-and-bound algorithm based on the auxiliary graph introduced by Bouhtou et al. (2007). Brotcorne et al. (2001), relying on a single-level formulation involving both primal and dual variables, proposed a heuristic algorithm that iterates between primal and dual problems.

The description of the clique pricing problem is based on Heilporn et al. (2010b) and Heilporn et al. (2011). Product pricing is discussed in Dobson and Kalish (1988) and the parallel with CP is established in Heilporn et al. (2010a). Information concerning the first three variants can be found in Kuiteing et al. (2016), Gilbert et al. (2015) and Marcotte et al. (2013), and concerning the next two in Brotcorne et al. (2000) and Brotcorne et al. (2008). An application of bilevel network pricing worth mentioning involves the transportation of hazardous material, as considered, among others, by Kara and Verter (2004) and Marcotte et al. (2009).

Interdiction problems date from antiquity: see Wood (2011) for a historical perspective. The shortest path interdiction problem has been shown to be NP-hard by Ball et al. (1989) and the Benders decomposition method for solving it has been proposed in Israeli and Wood (2002). The interpretation of the path interdiction problem based on mixed strategies is due to Washburn and Wood (1995), who also investigated variants of the basic problem. Our presentation of the maximum flow interdiction problem is based on Wood (1993) in which the problem is shown to be strongly NP-hard and a branch-and-cut algorithm is considered.

8 Conclusions and Perspectives

A large number of optimization problems involving networks lend themselves naturally to bilevel formulations. One can actually argue that this is the case of all design problems involving users of the network that are not under the direct control of the designer. Although generic algorithms may provide valuable insights, they are unlikely to scale well, which warrants the development of heuristics well suited to the underlying network structure. In parallel, one may look forward to alternative approaches inspired by new developments in artificial intelligence, and more specifically by the techniques of machine learning applied to combinatorial optimization (see Bengio et al. (2018) for instance).

Acknowledgments The research of M. Labbé was partially supported by Fonds de la Recherche Scientifique—FNRS (grant PDR T0098.18). The research of P. Marcotte was partially supported by the National Sciences and Engineering Research Council of Canada RGPIN grant 05073.

References

Abdulaal, M., & LeBlanc, J. L. (1979). Continuous equilibrium network design models. *Transportation Research Part B: Methodological, 13*(1), 19–32.

Abouee-Mehrizi, H., Babri, S., Berman, O., & Shavandi, H. (2011). Optimizing capacity, pricing and location decisions on a congested network with balking. *Mathematical Methods of Operations Research, 74*(2), 233–255.

Ball, M. O., Golden, B. L., & Vohra, R. V. (1989). Finding the most vital arcs in a network. *Operations Research Letters, 8*(2), 73–76.

Bengio, Y., Lodi, A., & Prouvost, A. (2018). *Machine learning for Combinatorial Optimization: A Methodological Tour d'Horizon*. arXiv:1811.06128.

Bouhtou, M., van Hoesel, S., van der Kraaij, A. F., & Lutton, J. L. (2007). Tariff optimization in networks. *INFORMS Journal on Computing, 19*(3), 458–469.

Brotcorne, L., Labbé, M., Marcotte, P., & Savard, G. (2000). A bilevel model and solution algorithm for a freight tariff-setting problem. *Transportation Science, 34*(3), 289–302.

Brotcorne, L., Labbé, M., Marcotte, P., & Savard, G. (2001). A bilevel model for toll optimization on a multicommodity transportation network. *Transportation Science, 35*(4), 345–358.

Brotcorne, L., Labbé, M., Marcotte, P., & Savard, G. (2008). Joint design and pricing on a network. *Operations Research, 56*(5), 1104–1115.

Colson, B., Marcotte, P., & Savard, G. (2005). A trust-region method for nonlinear bilevel programming: Algorithm and computational experience. *Computational Optimization and Applications, 30*(3), 211–227.

Colson, B., Marcotte, P., & Savard, G. (2007). An overview of bilevel optimization. *Annals of Operations Research, 153*(1), 235–256.

Dan, T., & Marcotte, P. (2019). Competitive facility location with selfish users and queues. *Operations Research, 67*(2), 479–497.

Dempe, S. (2002). *Foundations of bilevel programming*. Berlin: Springer.

Dempe, S. (2003). Annotated bibliography on bilevel programming and mathematical programs with equilibrium constraints. *Optimization, 52*(3), 333–359.

Dewez, S., Labbé, M., Marcotte, P., & Savard, G. (2008). New formulations and valid inequalities for a bilevel pricing problem. *Operations Research Letters, 36*(2), 141–149.

Dobson, G., & Kalish, S. (1988). Positioning and pricing a product line. *Marketing Science, 7*(2), 107–125.

Gairing, M., Harks, T., & Klimm, M. (2017). Complexity and approximation of the continuous network design problem. *SIAM Journal on Optimization, 27*(3), 1554–1582.

Gilbert, F., Marcotte, P., & Savard, G. (2015). A numerical study of the logit network pricing problem. *Transportation Science, 49*(3), 706–719.

Hassin, R. (2016). *Rational queueing*. New York: CRC Press.

Heilporn, G., Labbé, M., Marcotte, P., & Savard, G. (2010a). A parallel between two classes of pricing problems in transportation and marketing. *Journal of Revenue and Pricing Management, 9*(1–2), 110–125.

Heilporn, G., Labbé, M., Marcotte, P., & Savard, G. (2010b). A polyhedral study of the network pricing problem with connected toll arcs. *Networks, 55*(3), 234–246.

Heilporn, G., Labbé, M., Marcotte, P., & Savard, G. (2011). Valid inequalities and branch-and-cut for the clique pricing problem. *Discrete Optimization, 8*(3), 393–410.

Israeli, E., & Wood, R. K. (2002). Shortest-path network interdiction. *Networks, 40*(2), 97–111.

Joret, G. (2011). Stackelberg network pricing is hard to approximate. *Networks, 57*(2), 117–120.

Kara, Y. B., & Verter, V. (2004). Designing a road network for hazardous materials transportation. *Transportation Science, 38*(2), 188–196.

Kuiteing, A. K., Marcotte, P., & Savard, G. (2016). Network pricing of congestion-free networks: The elastic and linear demand case. *Transportation Science, 51*(3), 791–806.

Labbé, M., Marcotte, P., & Savard, G. (1998). A bilevel model of taxation and its application to optimal highway pricing. *Management Science, 44*(12-Part-1), 1608–1622.

LeBlanc, J. L., & Boyce, E. D. (1986). A bilevel programming algorithm for exact solution of the network design problem with user-optimal flows. *Transportation Research Part B: Methodological, 20*(3), 259–265.

Luo, Z. Q., Pang, J. S., & Ralph, D. (1996). *Mathematical programs with equilibrium constraints*. Cambridge: Cambridge University.

Marcotte, P. (1986). Network design problem with congestion effects: A case of bilevel programming. *Mathematical Programming, 34*(2), 142–162.

Marcotte, P., & Marquis, G. (1992). Efficient implementation of heuristics for the continuous network design problem. *Annals of Operations Research, 34*(1), 163–176.

Marcotte, P., Mercier, A., Savard, G., & Verter, V. (2009). Toll policies for mitigating hazardous materials transport risk. *Transportation Science, 43*(2), 228–243.

Marcotte, P., Savard, G., & Schoeb, A. (2013). A hybrid approach to the solution of a pricing model with continuous demand segmentation. *EURO Journal on Computational Optimization, 1*(1–2), 117–142.

Marianov, V., Ríos, M., & Icaza, M. J. (2008). Facility location for market capture when users rank facilities by shorter travel and waiting times. *European Journal of Operational Research, 191*(1), 32–44.

Roch, S., Savard, G., & Marcotte, P. (2005). An approximation algorithm for Stackelberg network pricing. *Networks, 46*(1), 57–67.

Talbi, E. G. E. (2013). *Metaheuristics for Bi-level optimization*. Berlin: Springer.

Van Hoesel, S. (2008). An overview of Stackelberg pricing in networks. *European Journal of Operational Research, 189*(3), 1393–1402.

Washburn, A., & Wood, K. (1995). Two-person zero-sum games for network interdiction. *Operations Research, 43*(2), 243–251.

Wood, R. K. (1993). Deterministic network interdiction. *Mathematical and Computer Modelling, 17*(2), 1–18.

Wood, R. K. (2011). Bilevel network interdiction models: Formulations and solutions. In *Wiley Encyclopedia of operations research and management science*.

Zhang, Y., Berman, O., Marcotte, P., & Verter, V. (2010). A bilevel model for preventive healthcare facility network design with congestion. *IIE Transactions, 42*(12), 865–880.

Chapter 10
Stochastic Network Design

Mike Hewitt, Walter Rei, and Stein W. Wallace

1 Introduction

As clearly shown in this book, network design models have been successfully used in a wide variety of optimization contexts. For example, service network design methodologies are applied to efficiently solve various planning problems appearing in freight transportation applications. In these cases, the network design models aim to first select a subset of logistical services that fix an organization's capabilities to perform transportation and storage operations in different geographical areas and over varying time horizons. These services are then used to move various types of freight from a series of origins to a series of destinations.

Considering the general nature of such models and their wide applicability, there has been a steady stream of research conducted on them over the years. The vast majority of this research has focused on deterministic variants of these models, i.e., methodological developments assuming that all information (or parameters) required to define the optimization models are readily available and not subject to uncertainty. However, in the planning processes in which these problems appear,

M. Hewitt
Information Systems and Supply Chain Management Department, Quinlan School of Business, Loyola University Chicago, Chicago, IL, USA
e-mail: mhewitt3@luc.edu

W. Rei (✉)
CIRRELT and Département d'Analytique, Opérations et Technologies de l'Information, École des Sciences de la Gestion, Université du Québec à Montréal, Montreal, QC, Canada
e-mail: rei.walter@uqam.ca

S. W. Wallace
Department of Business and Management Science, NHH Norwegain School of Economics, Bergen, Norway
e-mail: stein.wallace@nhh.no

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_10

there is always a time delay between the moment when the design decisions are made (i.e., selection of the services) and when the designed network is then used (i.e., moving the freight). As a consequence, there are various sources of uncertainty which affect the informational context (i.e., the parameters) that define the optimization models. In addition, even after operations start, many parameters (particularly demand) is unpredictable from one day to the next.

But why do we care about uncertainty? Isn't a deterministic model (where, for example, all random variables are replaced by their means) a reasonable approximation of the true stochastic setting? The answer is no, and it is important to understand why that is the case. From finance we know options, which we may view as investments in flexibility. They could be financial—the right, but not the duty, to buy stocks at a certain price at a certain time—or they could be real (physical)—the right, but not the duty, to rent a truck at a certain price at a certain time. Option theory tells us how to value the options. It is impossible to value an option in a model that does not contain uncertainty (random variables). We need to let the model figure out: Will I exercise the option or not?

In the context where network design models are used, there are also real options (as well as financial ones, but we exclude them here). They could be such as: send a truck on different routes depending on the demand of the day; rent trucks from a competitor; spend some extra time at depots to pick up freight that has been slightly delayed; set up routes that are more flexible with respect to the rerouting of goods. These can all be seen as options, they come at a cost, but they are not quite as simple to value as the ones from options theory. These costs may be linked to the fact that drivers require additional pay if they are to be willing to be rerouted on the day (or they will quit more often, leading to higher training and hiring costs), you may have to pay a fee to have the right to rent a truck, extra time spent at a depot wastes transportation resources, and maybe the more flexible routes cost a bit more. This kind of options cannot be valued using classical options theory as they are not explicit.

Now, let us consider an optimization model. Specifically, assume you have a deterministic model with flexible decisions involved, such as the right to rent a truck or the possibility to reroute them. Or even simpler: the right to buy insurance to protect against an undesirable random occurrence. Would the solution to a deterministic model suggest you do any of this? The answer is a clear NO, unless the options come for free—which they normally don't. And why? Because flexibility has no value (and meaning) in a deterministic world. If you know that your house will burn, you would like to buy insurance, but nobody would sell it to you. If you know it will not burn, you will not buy insurance, but many would like to sell it to you. Therefore, by solving a deterministic model the obtained solution will not prescribe to pay anything for flexibility because the term has no meaning (or value in this context); you will never pay something for nothing.

So a stochastic model is needed to find flexible solutions. This is obvious for explicit options like the right to rent a truck (presumably more important than simply renting a truck), or, the opportunity to spend extra time at the depot. But this also applies to implicit options like a more flexible schedule. Why pay for flexibility when you (know that you) will not need it? This points to the final difference between options theory and stochastic optimization: options theory can only be used to value explicitly defined options, it cannot be used to find them. Solving stochastic programs produces options as output, while options theory take them as input.

As it has been observed by numerous researchers in the field, sources of uncertainty that affect the informational context in which network design models appear can be classified in different ways. One such classification, that is readily used, categorizes these sources of uncertainty in three general groups: *randomness*, *hazards* and *deep uncertainty*.

Randomness refers to sources of uncertainty that are both foreseeable and that regularly occur: e.g., seasonal changes in the demands for given products, market value fluctuations under normal conditions, etc. For network design applications, such uncertainty is often associated with changes observed in either the costs of designing or using the networks, e.g., the transportation costs varying according to the congestion levels in the network; or in the needs that have to be met by using the network, e.g., the volume of freight to be transported from specific origins to given destinations that vary according to changes in the demands of clients.

Hazards are instead random events that occur more rarely but have a big impact on the considered problem settings. In the context of network design, such events often take the form of disruptive incidents that interfere with the use of the designed network, e.g., pipe failures occurring in a water distribution network that affect operational reliability (i.e., the network's capacity to meet the water demands of a given population).

Finally, *deep uncertainty* are events that have both profound impacts on the problem settings (impacts that oftentimes cannot even be properly evaluated) and are extremely hard to predict. Such uncertainty is usually associated with catastrophic random events that shape the problems in a radical and hard to assess way. The occurrence of a natural disaster that then requires the deployment of logistics services to provide humanitarian aid is a clear example of this type of uncertainty.

Although each source of uncertainty poses specific challenges when developing efficient network design methodologies, the present chapter will focus on the first two sources. Considering the increasingly large and diversified databases that are now accessible to organizations, by statistically analyzing such databases, it is now within reach to derive more accurate probabilistic models for the random phenomena that can be expected (i.e., randomness and hazards). Stochastic optimization then leverages such probabilistic models to define more efficient methodologies to solve optimization problems that involve uncertainty. Therefore, the overall goal of this chapter is to present how stochastic optimization is applied in the context of network design problems.

Towards this aim we will begin by presenting the different paradigms that can be used to formulate stochastic network design problems (i.e., Sect. 2). This presentation will be done by illustrating the paradigms on a specific problem, namely the stochastic fixed-charge capacitated multicommodity network design problem. We will then describe how scenario generation is applied to approximate the random distributions used to model the stochastic parameters in network design models (i.e., Sect. 3). We will next present the general solution approaches (both exact and heuristics) that can be applied to solve stochastic network design models. Specifically, our main focus will be to detail how decomposition strategies are applied to produce more efficient solution processes for the considered models (i.e., Sect. 4). Finally, we will conclude by providing some perspectives regarding the future research in the field of stochastic network design (i.e., Sect. 5).

2 Stochastic Models

In this section, we review how the stochastic optimization modelling paradigms are applied in the context of network design problems. There are two such modelling paradigms: (1) stochastic programming with recourse and (2) stochastic programming with probabilistic constraints. In the first paradigm, a problem is formulated by first establishing what are the *stages* that define the problem's informational context. These stages determine the process by which the stochastic parameters become known (i.e., the specific moments in time when the values of the stochastic parameters are observed). Decisions are then defined based on these stages, thus reacting to what information is known and what information remains uncertain. Therefore, one refers to *a priori* decisions as those that need to be made before any of the stochastic parameters are observed (i.e., the first stage). As for *recourse* decisions, they refer to the adjustments that are made once more information becomes available (i.e., decisions made from the second stage and onward when stochastic parameters are observed).

Based on these definitions, stochastic programs with recourse are referred to as either *two-stage* or *multi-stage* models. In addition to the a priori decisions occurring at the first stage, a two stage-model assumes that all recourse decisions involve a single stage (i.e., the second stage at which point all the stochastic parameters become known). In a multi-stage model, recourse decisions are instead made over multiple stages based on the stochastic parameters being gradually revealed. In all cases, stochastic programming with recourse aims to find solutions that, at each decisional stage, will maximize (or minimize) an expected utility (or cost) function that is either defined over the full random distributions of the stochastic parameters (e.g., the *Expected Value*), or, over specific occurrences that may define risky events (e.g., the *Conditional Value at Risk*).

As opposed to stochastic programming with recourse that produces a detailed formulation over different stages, stochastic programming with probabilistic constraints (also referred to as *chance constraints*), which is the second modelling paradigm, enables problems that involve stochastic parameters to be formulated without the need to explicitly define recourse actions and assess their impact. In such programs, a subset of the constraints, or part of the objective, are instead defined as probabilistic statements based on the a priori decisions to be made. Therefore, for example, the reliability of a priori decisions can be assessed in probabilistic terms. In such a case, the program would aim to find an a priori solution that is reliable to perform given tasks, or operations, for specific probabilities with respect to the random changes that may occur in the stochastic parameters considered.

The rest of this section is divided as follows. In Sect. 2.1, we discuss how network design problems can be formulated as stochastic programs with recourse by fixing how decisions, that are either related to the design of the network or usage of it, are defined according to the stages in the informational context. In Sect. 2.2, we explore how probabilistic constraints are applied in the case of network design, mainly to enforce quality of service requirements for the networks designed in the presence of uncertainty.

2.1 Stochastic Programs with Recourse

Before describing the general formulations, an important point to emphasize is the distinction between the concepts of *stages* and *periods* when applying stochastic programming with recourse to network design problems. In network design applications, decisions can often be made over multiple periods of time. When service network design is applied in the context of freight transportation, one finds that the selection and use of logistic services can occur at different points in time and be performed over multiple time periods, e.g., a maritime transportation service between two ports can thus be scheduled on different days (i.e., periods) and requires multiple days of travel time to be performed. Therefore, multiple time periods may be required to define both the design and use decisions in such cases. Stages are instead related to the informational context of the considered problem (i.e., a stage is defined based on what parameters are known and what parameters remain uncertain). As a result, a specific stage may include decisions over multiple time periods that all need to be made based on the same level of contextual information.

We will now use as an illustrative example the stochastic fixed-charge capacitated multicommodity network design problem to present how stochastic programming with recourse can be applied in this case. Let us recall that $\mathscr{G} = (\mathscr{N}, \mathscr{A})$ defines a directed graph, where $\mathscr{N}$ and $\mathscr{A}$ are the sets of nodes and arcs, respectively, and $\mathscr{K}$ defines the set of commodities. Each $k \in \mathscr{K}$ entails that a certain volume of the commodity needs to be transported from an origin $o(k) \in \mathscr{N}$ to a destination $s(k) \in \mathscr{N}$. There are two types of decision variables in fixed-charge capacitated multicommodity network design models: (1) the design variables, which fix the configuration of the network, and (2) the flow variables, which determine how the commodities are transported from their respective origins to their destinations using the designed network. Therefore, let the design variables be defined as $y_{ij} \in \{0, 1\}$, if the arc $(i, j) \in \mathscr{A}$ is included in the network (or not), and the flow variables be defined as $x^k_{ij} \geq 0$, the quantity of commodity $k \in \mathscr{K}$ that transits through arc $(i, j) \in \mathscr{A}$. In the following, we will refer to y and x as the vectors of design and flow variables, respectively.

The present modelling paradigm will be demonstrated using the two-stage stochastic version, which is quite versatile and has been successfully applied on a wide gamut of applications. Considering that such models are particularly useful to formulate tactical and strategic planning problems that occur when managing complex systems (i.e., supply chains, transportation systems, smart grids, etc.), it is often considered that the a priori decisions are the design variables (i.e., deciding on tactical or strategic plans that fix the structure of the system) and that the recourse decisions involve the flow variables (i.e., determining how the system is then used to perform regular operations).

Therefore, to formulate the problem as a two-stage stochastic model, one first needs to clearly describe how the sequence of decisions occur. Let us first state that the various sources of uncertainty are expressed here in terms of random experiments defined on a probability space for which the outcomes are defined as $\omega \in \Omega$, where Ω is the set of all possible outcomes, and where P is a function defining the probabilities associated with the possible outcomes. We further define ξ to be a random vector that contains the various stochastic parameters that are present in the problem. Once the random experiments have occurred, the specific values of the stochastic parameters can be observed and we denote these values by the vector $\xi(\omega)$. The sequence of decisions that define the problem can then be represented as the process illustrated in Fig. 10.1. In this case, the flow variables, which represent the recourse, are defined according to the specific outcome ω that is observed following the random experiments. These decisions will also depend on the specific network that is designed in the first stage.

The second step in producing the two-stage stochastic model is to determine what are the stochastic parameters present in the problem. Considering the fact that, in this case, the second stage decisions are related to how the network is used to route the commodities, there are three types of stochastic parameters that can be present: (1) flow costs, (2) commodity volumes and (3) arc capacities. The flow costs can randomly change based on sources of uncertainty affecting either the arcs (e.g., congestion on a given road) or the commodities (e.g., handling costs for given products that randomly change). Therefore, we define $c_{ij}^k(\omega) \in \Re^+$, $\forall (i, j) \in \mathscr{A}$, $k \in \mathscr{K}$, as the values of the commodity flow costs if the outcome $\omega \in \Omega$ is observed.

As for the commodities $k \in \mathscr{K}$, as previously stated, they are defined using three parameters: (1) the origin node $o(k)$, (2) the destination node $s(k)$ and (3) the volume to be transported. In network design applications, such commodities are usually associated with regular operations to be performed within the network, e.g., the supply of a specific plant from a given supplier in a supply chain or the volume of information that is exchanged between clients in a telecommunication network.

First Stage		Random Experiments		Second Stage
$y_{ij} \in \{0, 1\}, \forall (i, j) \in \mathscr{A}$	$\rightarrow$	$\xi(\omega)$	$\rightarrow$	$x_{ij}^k(\omega) \geq 0, \forall k \in \mathscr{K}.$

Fig. 10.1 Two-stage decision sequence

Therefore, the origin and destination nodes tend to be known in advance and fixed. However, the volumes can be expected to randomly change, e.g., supply volumes for plants will fluctuate based on random variations affecting product demands in a supply chain. We thus define, $\forall k \in \mathscr{K}$ and $i \in \mathscr{N}$, the following stochastic demand parameters:

$$d_i^k(\omega) = \begin{cases} v^k(\omega) & \text{if } i = o(k), \\ -v^k(\omega) & \text{if } i = s(k), \\ 0 & \text{otherwise,} \end{cases}$$

where $v^k(\omega) \in \Re^+$ defines the volume of commodity k that needs to be transported between $o(k)$ and $s(k)$, if the outcome $\omega \in \Omega$ is observed.

The capacities, associated with selected arcs, define the last type of stochastic parameters that can be present in the introduced two-stage stochastic model. The design decisions that are made in the first stage define the configuration of the network, i.e., the overall paths that connect the nodes and the capacities that are available along these paths to then flow the commodities. However, in various applications, hazards may occur that prevent the capacity associated with selected arcs to be fully available in the second stage, e.g., a mechanical problem preventing a truck to be used in a scheduled long-haul transportation service. Therefore, we define $u_{ij}(\omega) \in \Re^+$, $\forall (i, j) \in \mathscr{A}$, as the capacity values of the arcs if the outcome $\omega \in \Omega$ is observed.

Assuming that there is a fixed cost associated with the selection of each available arc, i.e., the values $f_{ij} \in \Re^+$, $\forall (i, j) \in \mathscr{A}$, then the two-stage stochastic fixed-charge capacitated multicommodity network design model can be formulated as follows:

$$\min \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + \mathbf{E}_\xi[Q(y, \xi(\omega))] \tag{10.1}$$

$$\text{s.t.} \qquad y_{ij} \in \{0, 1\}, \forall (i, j) \in \mathscr{A} \tag{10.2}$$

where, $Q(y, \xi(\omega))$ defines the total multicommodity flow cost considering that the designed network y is used when the outcome ω occurs and the values of the stochastic parameters are fixed to $\xi(\omega)$. The function $\mathbf{E}_\xi[\cdot]$ defines the expectation computed over the random vector ξ, and is often referred to as the expected recourse function (or the expected recourse cost). The purpose of such a function is usually to assess the cost (or the value) of repeatedly using the designed network under various contextual settings. In the present case, the variations that can be observed with regards to the contextual settings are captured through the support that is associated with the random vector ξ. Therefore, the model aims to find a network that minimizes the sum of the fixed costs that are incurred by the selected arcs and the expected multicommodity flow cost associated with the designed network, i.e., the objective function (10.1). As for the constraint set, one needs to impose the integrality requirements on the binary variables defining the selection decisions, i.e., constraints (10.2).

As for $Q(y, \xi(\omega))$, it is defined as follows:

$$Q(y, \xi(\omega)) = \min \sum_{k \in \mathscr{K}} \sum_{(i,j) \in \mathscr{A}} c_{ij}^k(\omega) x_{ij}^k(\omega) \tag{10.3}$$

$$\text{s.t.} \sum_{j \in \mathscr{N}^+(i)} x_{ij}^k(\omega) - \sum_{j \in \mathscr{N}^-(i)} x_{ji}^k(\omega) = d_i^k(\omega), \forall i \in \mathscr{N}, k \in \mathscr{K} \tag{10.4}$$

$$\sum_{k \in \mathscr{K}} x_{ij}^k(\omega) \le u_{ij}(\omega) y_{ij}, \quad \forall (i,j) \in \mathscr{A} \tag{10.5}$$

$$x_{ij}^k(\omega) \ge 0, \quad \forall (i,j) \in \mathscr{A}, \forall k \in \mathscr{K}. \tag{10.6}$$

The objective function is to minimize the total multicommodity flow cost (10.3), while enforcing the flow conservation constraints (10.4), the available capacities on the arcs that restrict the quantity of flow that can transit through them (10.5) and the non-negativity constraints on the flow variables (10.6).

Finally, the previous model assumes a two-stage decision sequence. This being said, there may be cases where decision-makers are interested in formulating network design problems where the information regarding the stochastic parameters are divulged over multiple stages and where decisions, either related to the network's characteristics or the flow of the commodities, are made according to these stages. In such cases, the sequence represented in Fig. 10.1 would be repeated a number of times, thus defining a multi-stage decision process. Each stage is defined here by a specific level of available information, i.e., a set of observed and stochastic parameters, as well as a set of decisions to be made (i.e., design or flow). The overall model is then obtained by linking the different stages through the inclusion of recourse cost functions (i.e., conditional expected value functions) defined over the decision variables and the level of information (i.e., the observed and the stochastic parameters) that establish the stages.

Although such stochastic multi-stage models can be applicable in different contexts, there has been very little research done on proposing methodologies to solve them directly. Therefore, we will not explicitly present such formulations. Nonetheless, we will mention that such problems can be approximated through a sequence of two-stage network design models, in which the multi-stage process is relaxed. Specifically, all stochastic parameters are assumed to be revealed at the same time.

Such approximations can be appropriate in various decision-making contexts. For example, when solving strategic (or tactical) planning problems, decision makers are often, principally interested in fixing the decisions that need to be made *a priori*. In the case of stochastic network design, these decisions involve the configuration of the network, which is then used over multiple periods that possibly involve multiple stages. In this context, the *recourse* decisions are mainly used to assess the expected future costs of the strategic (or tactical) decisions. Therefore, in such cases, relaxing the multi-stage process may still provide enough accuracy to serve this purpose efficiently.

2.2 Stochastic Programming with Probabilistic Constraints

As previously mentioned, the use of probabilistic constraints enable a stochastic problem to be formulated without the need to explicitly define detailed recourse actions. In the present case, such formulations can be used to model reliability issues when designing networks in the presence of stochastic parameters. The concept of reliability refers here to the capacity of a network to efficiently perform required tasks, or operations, when random changes occur in the stochastic parameters. Before presenting specific formulations using the stochastic fixed-charge capacitated multicommodity network design problem, let us revisit how the flow component of the problem is expressed.

In all generality, one can consider that additional resources, that are outside of the designed network (i.e., outsourcing services, renting additional equipment or manpower, etc.), can be employed to perform the necessary operations (i.e., flow the different commodities from their origins to their destinations). In addition, one can assume that, in certain cases, an organization may opt to either delay, or refuse, part of the commodities to be transported (provided that a penalty is applied). To formulate these options, we redefine the arc set as follows $\mathscr{A} = \mathscr{A}^A \cup \mathscr{A}^D$, where set $\mathscr{A}^A$ represents the design arcs (i.e., arcs that can be used to design the network) and $\mathscr{A}^D$ is a set of dummy arcs associated with the commodities. Specifically, for each considered commodity, set $\mathscr{A}^D$ includes an arc that directly connects the origin to the destination, i.e., $\mathscr{A}^D = \{(o(k), s(k)) \mid \forall k \in \mathscr{K}\}$. Considering that these dummy arcs are associated with either additional resources being used, or, a part of the commodities not being fulfilled, there are no capacity restrictions imposed on them. Simply, when they are used, a higher unit price (or penalty) is applicable. It should be noted that the inclusion of these additional dummy arcs in the previously defined stochastic program with recourse (10.1)–(10.2) would provide the relatively complete recourse property to the model.

The reliability of a network can now be defined on the basis of either how many additional resources are required to transport the commodities or how much of the commodities are unfulfilled. For example, a designed network considered reliable could be defined as one that ensures that the probability of observing a total amount of flow on the dummy arcs that is more than a threshold (i.e., value $\alpha \in \Re^+$) be at most a specified upper bound (i.e., value $\beta \in [0, 1]$). Therefore, for a given threshold α, the lower the specified value of the upper bound β, the more reliable the designed network will be.

To illustrate this modelling approach, let us consider a simpler setting for the problem, where the stochastic parameters are the demands (i.e., the commodity volumes), i.e., $\xi = [\mathbf{d}]$, where $\mathbf{d}$ is a vector containing the stochastic demands whose elements $\mathbf{d}_i^k$ may take the possible values $d_i^k(\omega)$, $\forall k \in \mathscr{K}$, $i \in \mathscr{N}$ and $\omega \in \Omega$. Furthermore, let functions $f_i^k(x, \mathbf{d})$, $\forall i \in \mathscr{N}$, $k \in \mathscr{K}$, be defined as

$$f_i^k(x, \mathbf{d}) = \left(\sum_{j \in \mathcal{N}^+(i)} x_{ij}^k - \sum_{j \in \mathcal{N}^-(i)} x_{ji}^k \right) - \mathbf{d}_i^k$$

a stochastic programming with a probabilistic constraints, that expresses the previously defined network reliability property, can then be formulated as follows:

$$\min \quad \sum_{(i,j)\in\mathcal{A}^A} f_{ij} y_{ij} + \sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^k \tag{10.7}$$

$$\text{s.t. } P\left(\left\{\mathbf{d} \mid \sum_{(i,j)\in\mathcal{A}^D} \sum_{k\in\mathcal{K}} x_{ij}^k \geq \alpha,\ f_i^k(x,\mathbf{d}) = 0, \forall i \in \mathcal{N}, k \in \mathcal{K}\right\}\right) \leq \beta \tag{10.8}$$

$$\sum_{k\in\mathcal{K}} x_{ij}^k \leq u_{ij} y_{ij}, \quad \forall (i,j) \in \mathcal{A}^A \tag{10.9}$$

$$y_{ij} \in \{0,1\}, \forall (i,j) \in \mathcal{A}^A \tag{10.10}$$

$$x_{ij}^k \geq 0, \quad \forall (i,j) \in \mathcal{A}, k \in \mathcal{K}. \tag{10.11}$$

In this case, the objective function (10.7) is to minimize the total cost, which includes both the fixed costs associated with the design decisions and the flow costs overall all considered arcs (i.e., both the design and the dummy arcs). Constraint (10.8) is the probabilistic statement that defines the reliability requirements that are imposed on the designed network. Specifically, the upper bound β is imposed on the probability of observing demand values $\mathbf{d}$ such that the feasible flow throughout the considered network (i.e., $f_i^k(x,\mathbf{d}) = 0, \forall i \in \mathcal{N}, k \in \mathcal{K}$) is such that the use of the dummy arcs is at least the threshold beyond which the reliability requirements would not be enforced (i.e., $\sum_{(i,j)\in\mathcal{A}^D} \sum_{k\in\mathcal{K}} x_{ij}^k \geq \alpha$). Constraints (10.9), once again define the capacity that is available to flow the commodities over the selected arcs. Finally, constraints (10.10) and (10.11), impose the necessary integrality and non-negativity requirements on the decision variables.

Considering that capacity uncertainty with respect to the design arcs can also greatly influence the reliability of the network, one may want to include such sources of uncertainty in the previous model. To do so, one would define the stochastic parameters as follows: $\xi = [\mathbf{d}, \mathbf{u}]$. Therefore, in addition to the stochastic demand vector $\mathbf{d}$, vector $\mathbf{u}$, which contains the stochastic capacity parameters whose elements $\mathbf{u}_{ij}$ may take the possible values $u_{ij}(\omega)$, $\forall (i,j) \in \mathcal{A}^A$ and $\omega \in \Omega$, is also included in ξ. By defining the functions $f_{ij}(y, x, \mathbf{u})$, $\forall (i,j) \in \mathcal{A}^A$, as

$$f_{ij}(y, x, \mathbf{u}) = \sum_{k\in\mathcal{K}} x_{ij}^k - \mathbf{u}_{ij} y_{ij}$$

the reliability requirement can then be formulated as the following probabilistic constraint

$$P\left(\left\{\xi \mid \sum_{(i,j)\in\mathscr{A}^D}\sum_{k\in\mathscr{K}} x_{ij}^k \geq \alpha,\ f_i^k(x,\mathbf{d}) = 0, \forall i \in \mathscr{N}, k \in \mathscr{K},\right.\right.$$
$$\left.\left. f_{ij}(y,x,\mathbf{u}) \leq 0, \forall (i,j) \in \mathscr{A}^A\right\}\right) \leq \beta. \quad (10.12)$$

In this case, both (1) the flow conservation constraints imposed on the nodes of the network (i.e., $f_i^k(x,\mathbf{d}) = 0, \forall i \in \mathscr{N}, k \in \mathscr{K}$) and (2) the capacity limits imposed on the design arcs (i.e., $f_{ij}(y,x,\mathbf{u}) \leq 0, \forall (i,j) \in \mathscr{A}^A$) are used to define the conditions on which the probabilistic constraint is based.

3 Scenario Generation for Stochastic Network Design

By scenario generation we mean how to construct discrete distributions that can be used to solve the optimization models presented earlier. We would argue that, even if scenarios are not your main interest, or not your interest at all, there are some important issues to think about. The most important challenge is that the scenarios represent the underlying uncertainty, but how can you know that they do? In some contexts, such as specific games in a casino, there is a chance that the scenarios really are the uncertainty. In such examples, the stochastic parameters take the form of discrete random variables with finite support and the set of possible outcomes can be enumerated. But for network design, this is not the case. Here, the scenarios represent an approximation of the true uncertainty. Such approximations are a necessary requirement to obtain solvable models for the considered stochastic optimization problems.

Therefore, in the present section, we discuss some of the important points to consider when generating scenarios in stochastic network design settings. It should be noted that our presentation does not cover specific sampling methods. Such methods are part of a broad scientific domain, i.e. probability theory and statistics, that is well beyond the focus of the present chapter (nonetheless we provide some important references on this subject in the bibliographical notes of the chapter). Instead, we present how network design models are formulated when using a scenario set to approximate the stochastic parameters considered (Sect. 3.1), define some of the stability tests that can be easily conducted to assess the value of the obtained scenarios for decision-making in network design (Sect. 3.2) and discuss some of the important data challenges that may arise when attempting to generate scenarios in this context (Sect. 3.3).

3.1 Scenario-Based Network Design Models

Let us first consider the case where a recourse program is formulated. A scenario-based approximation can be obtained for this program when a finite set of representative scenarios S is generated for the random vector ξ. Specifically, for $s \in S$, we define the vectors ξ^s and the values $p_s \geq 0$ (such that $\sum_{s \in S} p_s = 1$), as the possible realizations of the random vector ξ in the considered scenario set and their probability of occurence, respectively. An approximation of the original stochastic network design model can then be obtained by assuming that ξ will be fixed to one of the possible realizations ξ^s, $s \in S$. As previously defined, the flow decisions set the recourse actions that are applied when the values of the stochastic parameters become known. The use of the scenario set S entails that the random vector ξ will be fixed to one of the ξ^s, for $s \in S$. Therefore, the following decision vectors can be defined: x^s, i.e., $x_{ij}^{ks} \geq 0$, $\forall k \in \mathscr{K}$, $(i, j) \in \mathscr{A}$ and $s \in S$. These vectors establish how the commodities are flowed through the designed network if the vector ξ^s is observed. Assuming that $\xi^s = [d^s, u^s, c^s]$, $\forall s \in S$, where d^s, u^s and c^s are the vectors whose values are the scenario demands, i.e., d_i^{ks}, $\forall k \in \mathscr{K}$ and $i \in \mathscr{N}$, the scenario capacities, i.e., u_{ij}^s, $\forall (i, j) \in \mathscr{A}$, and the scenario flow costs, i.e., c_{ij}^{ks}, $\forall (i, j) \in \mathscr{A}$, $k \in \mathscr{K}$, respectively, then the scenario-based network design model with recourse can be formulated as follows:

$$\min \quad \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + \sum_{s\in S} \sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}} p_s c_{ij}^{ks} x_{ij}^{ks} \tag{10.13}$$

$$\text{s.t.} \quad \sum_{j\in\mathscr{N}^+(i)} x_{ij}^{ks} - \sum_{j\in\mathscr{N}^-(i)} x_{ji}^{ks} = d_i^{ks}, \quad \forall i \in \mathscr{N}, k \in \mathscr{K}, s \in S \tag{10.14}$$

$$\sum_{k\in\mathscr{K}} x_{ij}^{ks} \leq u_{ij}^s y_{ij}, \quad \forall (i, j) \in \mathscr{A}, s \in S \tag{10.15}$$

$$y_{ij} \in \{0, 1\}, x_{ij}^{ks} \geq 0, \quad \forall (i, j) \in \mathscr{A}, k \in \mathscr{K}, s \in S. \tag{10.16}$$

Model (10.13)–(10.16) takes the form of a large-scale deterministic linear program, in which all considered recourse decisions are explicitly defined via the use of the scenario set.

We now present how a scenario-based network design program with probabilistic constraints can be formulated using the set of representative scenarios S. Let us consider the variant where demands and capacities are the stochastic parameters. In this case, for each scenario $s \in S$, one has the vector $\xi^s = [d^s, u^s]$. Let us recall that the original probabilistic constraint is defined here as (10.12). The vectors ξ^s, for $s \in S$, are then used to approximate the probability associated to the random event on which this constraint is based. Specifically, considering a designed network, which we define as $y_{ij} = \overline{y}_{ij}$, $\forall (i, j) \in \mathscr{A}$ (i.e., vector $\overline{y}$), and its associated commodity flows $x_{ij}^k = \overline{x}_{ij}^k$, $\forall (i, j) \in \mathscr{A}, k \in \mathscr{K}$ (i.e., vector $\overline{x}$), then one seeks to identify the set $S(\overline{y}, \overline{x}) = \{s \in S \mid \sum_{(i,j)\in\mathscr{A}^D} \sum_{k\in\mathscr{K}} \overline{x}_{ij}^k \geq$

α, $f_i^k(\overline{x}, d^s) = 0, \forall i \in \mathcal{N}, k \in \mathcal{K}, f_{ij}(\overline{y}, \overline{x}, u^s) \leq 0, \forall (i, j) \in \mathcal{A}^A\}$. Therefore, if the following inequality is verified: $\sum_{s \in S(\overline{y}, \overline{x})} p_s \leq \beta$, then the solution $(\overline{y}, \overline{x})$ is considered to be feasible with respect to the probabilistic constraint. Otherwise, $(\overline{y}, \overline{x})$ should be discarded as infeasible.

To introduce these feasibility conditions within the network design program, we first need to define the following variables: let $z^s \in \{0, 1\}$, $\forall s \in S$, which indicate whether or not the scenarios enforce the condition on which the probabilistic constraint is defined. Specifically, these variables express the following relations:

$$z^s = \begin{cases} 1, & \text{If } \sum_{(i,j) \in \mathcal{A}^D} \sum_{k \in \mathcal{K}} x_{ij}^k \geq \alpha \\ 0, & \text{Otherwise.} \end{cases}$$

Therefore, the scenario-based probabilistic network design program can be defined as follows:

$$\min \sum_{(i,j) \in \mathcal{A}^A} f_{ij} y_{ij} + \sum_{(i,j) \in \mathcal{A}} \sum_{k \in \mathcal{K}} c_{ij}^k x_{ij}^k \tag{10.17}$$

$$\text{s.t.} \quad \sum_{(i,j) \in \mathcal{A}^D} \sum_{k \in \mathcal{K}} x_{ij}^k \geq \alpha z^s, \qquad \forall s \in S \tag{10.18}$$

$$f_i^k(x, d^s) = 0, \qquad \forall i \in \mathcal{N}, k \in \mathcal{K}, s \in S \tag{10.19}$$

$$f_{ij}(y, x, u^s) \leq 0, \qquad \forall (i, j) \in \mathcal{A}^A, s \in S \tag{10.20}$$

$$\sum_{s \in S} p_s z^s \leq \beta \tag{10.21}$$

$$y_{ij} \in \{0, 1\}, z^s \in \{0, 1\}, x_{ij}^k \geq 0, \quad \forall (i, j) \in \mathcal{A}, k \in \mathcal{K}, s \in S. \tag{10.22}$$

The network reliability feasible conditions defined over the considered scenario set are obtained through the constraints (10.18) and (10.21). By including them in (10.17)–(10.22), the resulting model again takes the form of a large-scale deterministic linear program.

3.2 Stability Testing

Let us start with the extremely simplifying assumption that a distribution for the stochastic parameters is actually available (this assumption will be relaxed shortly). This could be an analytical distribution, e.g., a multi-dimensional log-normal distribution, an empirical distribution, e.g., a historical data set, or the output from repeated runs of a simulator. Regardless, let us assume that this is the distribution a decision-maker is interested in using to formulate the considered model but, for numerical reasons, this is not possible (e.g., analytical distributions cannot be handled or the number of possible values in the empirical distribution is too large). In such a case, a sample of scenarios would have to be chosen from the distribution (to be challenged shortly) to obtain a solvable model. At this point, one would ask what is the appropriate size for this sample?

In an effort to provide an answer to this important question, we suggest the following systematic approach (which we illustrate using the scenario-based recourse program). First, let us define function $F(y, S) = \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + \sum_{s\in S}\sum_{k\in\mathscr{K}}\sum_{(i,j)\in\mathscr{A}} p_s c_{ij}^{ks} x_{ij}^{ks}$ and set $Y = \{y_{ij} \in \{0, 1\}, \forall(i, j) \in \mathscr{A} \mid \exists x_{ij}^{ks} \geq 0, \forall(i, j) \in \mathscr{A}, k \in \mathscr{K}, s \in S,$ (10.14), (10.15)$\}$. Using this function and decision set, model (10.13)–(10.16) can be redefined as:

$$\min_{y\in Y} F(y, S).$$

Assume that a specific sample is chosen for a predefined number of scenarios (i.e., n). Of course, the value n would be chosen to obtain a scenario-based recourse program that is solvable in a reasonable amount of time. Furthermore, suppose that the chosen number of scenarios n is sampled a given number of times, say ten. Now solve your model ten times, one time with each scenario set. Do you, more or less, get the same optimal objective function value ten times? Specifically, consider two specific scenario samples S' and S''. We define y' and y'' as the optimal solutions (i.e., networks) when the scenario-based recourse program is solved using S' and S'', respectively. Therefore, the following condition is assessed:

$$F(y', S') \approx F(y'', S''). \tag{10.23}$$

If condition (10.23) is verified overall scenario sets then you have what is referred to as *in-sample stability*.

If this condition is not observed, then a larger scenario set is required. In this case, value n should be increased until the optimal objective value stabilizes at which point, any of the ten scenario sets would work. Since objective functions in stochastic programs tend to be very flat, and because we do not want to worry about lack of uniqueness of optimal solutions, we here refer to the objective function, not the solution being stable. Of course, if the optimal objective function value is very stable with your first choice of n, you may try again with a lower number. After all, fewer scenarios is a numerical advantage. Eventually, you will hopefully find a precision that suits you, and you can handle numerically.

Another test that can be conducted is the *out-of-sample stability*. To do so, one first needs to sample a larger scenario set $\hat{S}$ (e.g., $\mid \hat{S} \mid= 10{,}000$). The ten previously found networks are then evaluated using set $\hat{S}$. Normally, this is a simpler task to perform considering that the designs are fixed and only the second-stage problems are solved for the scenarios in $\hat{S}$. This process provides a more accurate approximation of the true value of those ten solutions. One may then evaluate how much the true objective values vary. Specifically, the following condition is assessed:

$$F(y', \hat{S}) \approx F(y'', \hat{S}). \tag{10.24}$$

If condition (10.24) is verified overall obtained networks then *out-of-sample stability* is verified.

But, why do we care about stability? Presumably, if what we face is a real application, where people put real money on the design, we do not want to advise them on a design that is optimal for the chosen scenarios, but not for the true distribution. After all, it is the true distribution the investor will face. So stability will guarantee that the problem that is solved is very close to the one that should be solved, but cannot.

3.3 Data Challenges in Scenario Generation

It is common practice in algorithmic work to randomly generate problem instances. Of course, also for stochastic network design that may indeed be necessary. But there are some challenges here that one may not be used to. These challenges are mostly connected to the handling of stochastic dependence. It will be typical that demands from customers are dependent. Consider the distribution of beer in a city, then it can be expected that good weather will result in increased beer sales in all restaurants and shops, leading to positive correlations. On the other hand, a company may have customers with rather stable total supply, but one may not quite know which factory the materials come from on a given day. That will lead to negative correlations. When there is historical data that can be analyzed, correlations can be more easily identified and quantified. However, when relevant data is more limited, this may entail important challenges.

Let us recall that stochastic optimization requires a complete probabilistic model to be available to formulate the considered stochastic parameters. Thus, multi-dimensional distributions need to be applied. Implicitly, this also entails that a correlation matrix (and often a bit more) is available. A correlation matrix has a certain structure. It has 1's on the main diagonal, it is symmetric, and each entry shows the correlation between a pair of variables. Although creating such a matrix may sound easy, it is not. The correlation matrix needs to be positive semi-definite, that is, all eigenvalues must be non-negative. Although it is easy to check if a matrix is positive semi-definite (there are numerous procedures available to do just that), one may find that many matrices that appear to be correct, in fact, are not. And if the obtained matrix is not a correlation matrix, then you don't have a distribution and you certainly cannot sample; you have nothing to sample from. Although, there are methods to turn any matrix into a positive semi-definite one, using such methods may lead to problems in terms of how stochastic parameters are formulated (e.g., undesirable correlations may appear).

Here is a simple approach that often leads to a proper correlation matrix. We will use here the stochastic demands case. Put your customers into two groups (systematically or randomly). Let all correlations within a group be positive, say 0.5, and between groups be negative, say -0.4. Check to see if the matrix is positive semi-definite. There are many codes from numerical linear algebra that can do that. If the matrix is not positive semi-definite, then slightly adjust the correlations. Most likely you will find something you can use. And once you have a distribution, the situation is like the one we described above where we assumed we had one. The issue is stability and how many scenarios we need. The number will depend on how we create the scenarios from the distribution, and can be found by using the stability tests described earlier.

4 Solution Methods

In this section, we review the general solution approaches (both exact and heuristic) that have been proposed for stochastic network design models. Our presentation starts from the point of view where a finite set of representative scenarios S has been generated for the random vector ξ. However, because the literature has focused on problem variants wherein flow costs do not vary by scenario (i.e. $c_{ij}^{ks} = c_{ij}^{ks'}, \forall s, s' \in S$), we present these methods for that variant. Thus, in this section, when referring to model (10.13)–(10.16), we refer to a model with the objective function $\sum_{(i,j)\in\mathscr{A}} f_{ij}y_{ij} + \sum_{s\in S}\sum_{k\in\mathscr{K}}\sum_{(i,j)\in\mathscr{A}} p_s c_{ij}^{k} x_{ij}^{ks}$.

As detailed previously, scenarios enable the stochastic problems to be formulated as large-scaled deterministic mixed-integer linear programs. In turn, these programs are amenable to be solved by applying specialized decomposition methods. In this section, we review the two main methods that have been applied to solve stochastic network design models: the Benders decomposition method, which is referred to as the L-Shaped method when applied in the context of stochastic optimization (i.e., Sect. 4.1) and the progressive hedging method, which applies a scenario decomposition strategy (i.e., Sect. 4.2).

4.1 Benders Decomposition

Benders decomposition is an algorithmic strategy that can be viewed from different perspectives and applied to a wide variety of problem classes. While initially proposed as a solution method for large-scale deterministic linear programs, it has often been successfully applied to stochastic programs formulated as deterministic linear and/or mixed integer linear programs. In those applications it is often referred to as the *L-Shaped Algorithm*.

Regardless of the class of optimization problem, Benders is an iterative procedure that, at an iteration, first solves a *Master* problem to determine the values of a subset of decision variables present in the original problem. Then, given the values of variables prescribed by the solution to the Master problem, subproblems are solved to prescribe values for the remaining decision variables, as well as to determine whether solutions to the subproblems can be combined with the Master problem solution to construct a provably (near-)optimal solution to the original problem. When they cannot, relevant information from the subproblems is embedded in the Master problem and the procedure continues.

When the subproblems are linear programs, information embedded in the Master problem takes the form of constraints that are derived from the dual polyhedra associated with subproblems. In these cases, Benders can be viewed as a *Cutting plane* or *Delayed constraint generation* method. Relatedly, when the original optimization problem is a MILP, the choice of variables to include in the Master problem is often driven by the desire to leave subproblems that are linear programs.

Thus, most applications of Benders to the model (10.13)–(10.16) (or related variants), solve a Master problem that determines the values for the design variables, y. We refer to those values as $\bar{y}$. Then, given the values $\bar{y}$, most applications of Benders continue by solving one or more subproblems to determine the values of the flow variables, x, in each scenario. Note that when the constraint $y = \bar{y}$ is added to model (10.13)–(10.16), it reduces to a set of capacitated multicommodity network flow problems (i.e. linear programs), one problem for each scenario.

Thus, a question that must be answered when implementing Benders for this (and other stochastic programs) is whether a single subproblem should be solved that considers all scenarios, or, a subproblem should be solved for each scenario. The former strategy (sometimes referred to as *Single-cut*) generates, at each iteration, an individual cut that bounds the overall recourse cost for the associated solution to the Master problem. The former strategy (sometimes referred to as *Multi-cut*), generates, at each iteration, a series of cuts that bound the recourse cost for the associated Master solution for each scenario subproblem separately. By disaggregating the cuts that express the recourse cost values of the solutions obtained, the Multi-cut strategy enables the Master's formulation to be strengthened more rapidly. Albeit, this is done at the cost of obtaining a Master problem that iteratively becomes harder to solve when compared to one generated using the Single-cut strategy. The question of which strategy to use is typically answered computationally. However, many recent Benders applications (to this and other stochastic programs) employ the Multi-cut strategy. Thus, we will next describe Benders in detail in the context of that strategy.

We will first present the subproblem that is solved for each scenario given values $\bar{y}$ for the design variables. Then, by taking the dual of that subproblem, we will present a reformulation of model (10.13)–(10.16) that involves the extreme points and extreme rays of the dual polyhedra associated with each scenario's subproblem. This reformulated problem is the basis of the Master problem solved by Benders, although at each iteration it is (usually) based on subsets of extreme points and/or rays.

Thus, given the values $\bar{y}$, the sub-problem, $SP(y)_s$, solved for scenario $s \in S$ to derive the commodity flows $\bar{x}^s$ is as follows:

$$\min \quad \sum_{k\in\mathscr{K}} \sum_{(i,j)\in\mathscr{A}} c_{ij}^k x_{ij}^k \tag{10.25}$$

$$\text{s.t.} \sum_{j\in\mathscr{N}^+(i)} x_{ij}^{ks} - \sum_{j\in\mathscr{N}^-(i)} x_{ji}^{ks} = d_i^{ks}, \quad \forall i \in \mathscr{N}, k \in \mathscr{K} \tag{10.26}$$

$$\sum_{k\in\mathscr{K}} x_{ij}^{ks} \leq u_{ij}^s \bar{y}_{ij}, \quad \forall (i,j) \in \mathscr{A} \tag{10.27}$$

$$x_{ij}^{ks} \geq 0, \quad \forall (i,j) \in \mathscr{A}, k \in \mathscr{K}. \tag{10.28}$$

As $\bar{y}$ is data, the right-hand-sides of constraints (10.27) reduce to data, and $SP(y)_s$ is a linear program. More generally, given that $\bar{y}_{ij} \in \{0,1\}$, the subproblem $SP(y)_s$ reduces to a capacitated multicommodity network flow problem tasked with routing the demand of each commodity in scenario s over the subset of arcs in $\mathscr{A}$ that are prescribed by $\bar{y}$.

To take the dual of $SP(y)_s$, one associates dual variables α_{ik} (un-restricted in sign) with constraints (10.26) and dual variables $\beta_{ij} (\leq 0)$ with constraints (10.27). With these dual variables, the dual, $DSP(y)_s$, of $SP(y)_s$ can be written as:

$$\max \quad \sum_{k\in\mathscr{K}} \sum_{i\in\mathscr{N}} d_i^{ks} \alpha_{ik} + \sum_{(i,j)\in\mathscr{A}} u_{ij}^s \bar{y}_{ij} \beta_{ij} \tag{10.29}$$

$$\text{s.t.} \quad \alpha_{ik} - \alpha_{jk} + \beta_{ij} \leq c_{ij}^k, \quad \forall (i,j) \in \mathscr{A}, k \in \mathscr{K} \tag{10.30}$$

$$\alpha_{ik} \text{ free }, \forall i \in \mathscr{N}, k \in \mathscr{K}, \quad \beta_{ij} \leq 0, \forall (i,j) \in \mathscr{A}. \tag{10.31}$$

Note that $\bar{y}$ does not appear in constraints (10.30) or (10.31). In other words, the polyhedron, Q, defined by the constraints (10.30)–(10.31) of $DSP(y)_s$ is independent of y. This independence will enable us to embed in the Benders Master problem cuts that are based on extreme points and rays of Q.

As is typically done, we assume $Q \neq \emptyset$. Thus, it possesses a set of extreme points $\mathscr{P}$ and a set of extreme rays $\mathscr{Q}$. We denote dual decision variable values associated with the pth extreme point in $\mathscr{P}$ as α_{ik}^p and β_{ij}^p, respectively. We denote dual decision variable values associated with the q^{th} extreme ray in $\mathscr{Q}$ in a similar fashion. Given these extreme points and rays, the following, which we refer to as $MP(\mathscr{Q}, \mathscr{P})$, is a valid reformulation of (10.13)–(10.16):

$$\min \quad \sum_{(i,j)\in\mathscr{A}} f_{ij} y_{ij} + \sum_{s\in S} p_s z_s \tag{10.32}$$

$$\textstyle\sum_{k\in\mathscr{K}} \sum_{i\in\mathscr{N}} d_i^{ks} \alpha_{ik}^q + \sum_{(i,j)\in\mathscr{A}} u_{ij}^s y_{ij} \beta_{ij}^q \leq 0, \quad \forall q \in \mathscr{Q}, s \in S \tag{10.33}$$

$$\text{s.t.} \textstyle\sum_{k\in\mathscr{K}} \sum_{i\in\mathscr{N}} d_i^{ks} \alpha_{ik}^p + \sum_{(i,j)\in\mathscr{A}} u_{ij}^s y_{ij} \beta_{ij}^p \leq z_s, \quad \forall p \in \mathscr{P}, s \in S \tag{10.34}$$

$$y_{ij} \in \{0,1\}, \quad \forall (i,j) \in \mathscr{A}, z_s \geq 0, \forall s \in S. \tag{10.35}$$

The validity of this reformulation, both for stochastic network design problems and stochastic programs in general, is based on Duality theory. Constraints (10.33) ensure that for each scenario s, the design, $\bar{y}$, induces a feasible subproblem $SP(\bar{y})_s$. Given the assumption that the dual polyhedron, Q, is non-empty, duality theory implies that when $SP(\bar{y})_s$ is infeasible, its dual must be unbounded. To prevent this, constraints (10.33) ensure that the Master problem does not prescribe values $\bar{y}$ that form a direction of unboundedness with any extreme rays of Q. Thus, constraints (10.33) are often referred to as *Feasibility cuts*.

The decision variables z_s represent an estimate of the optimal cost of $SP(y)_s$. Given the presence of the Feasibility cuts, we can presume that every design vector, $\bar{y}$, will induce a feasible subproblem $SP(\bar{y})_s$ for each scenario s. Weak duality implies that the optimal value of the scenario s subproblem is no less than the quantity $\sum_{k \in \mathscr{K}} \sum_{i \in \mathscr{N}} d_i^{ks} \alpha_{ik}^p + \sum_{(i,j) \in \mathscr{A}} u_{ij}^s \bar{y}_{ij} \beta_{ij}^p, \forall p \in \mathscr{P}$, as $\mathscr{P}$ contains extreme points and hence feasible solutions to the dual of the subproblem. Strong duality implies that the optimal value is exactly equal to that quantity for at least one extreme point. Thus, constraints (10.34) are often referred to as *Optimality cuts*.

Formulating and solving $MP(\mathscr{Q}, \mathscr{P})$ requires enumerating the extreme points and rays of the dual polyhedron, Q, which is computationally prohibitive. As a result, Benders considers subsets of $\mathscr{Q}$ and $\mathscr{P}$, respectively denoted by $\underline{\mathscr{Q}}$ and $\underline{\mathscr{P}}$. As considering subsets of extreme points and rays is equivalent to removing constraints from $MP(\mathscr{Q}, \mathscr{P})$, $MP(\underline{\mathscr{Q}}, \underline{\mathscr{P}})$ is a relaxation of the reformulation and the original model (10.13)–(10.16).

Thus, at each iteration, Benders solves $MP(\underline{\mathscr{Q}}, \underline{\mathscr{P}})$ to generate a lower bound, $\underline{v}$, on the optimal value of model (10.13)–(10.16), a set of scenario subproblem cost estimates, $\bar{z}_s$, and a vector of design variable values, $\bar{y}$. It then solves each subproblem $SP(\bar{y})_s$ to determine whether it is feasible, and if it is, to derive its optimal value $\bar{v}_s$. When $SP(\bar{y})_s$ is infeasible, the dual extreme ray that is a certificate of its infeasibility is extracted and used to generate a cut of the form (10.33). Alternately, when $SP(\bar{y})_s$ is feasible, but $\bar{z}_s < \bar{v}_s$, we have that the estimate of the flow costs associated with scenario s given the design $\bar{y}$ is too low. More precisely, we have that the set of extreme points used to construct $MP(\underline{\mathscr{Q}}, \underline{\mathscr{P}})$ is insufficient, and thus the extreme point associated with the optimal solution to the dual of $SP(\bar{y})_s$ is used to generate a cut of the form (10.34).

Formally, the basic Benders algorithm proceeds as described in Algorithm 1 below.

While Algorithm 1 is correct, there are many enhancements that have been proposed to Benders, both when solving general optimization problems and when solving models like (10.13)–(10.16). We will first discuss enhancements proposed for general problems, albeit in the context of solving stochastic network design models. First, note that Algorithm 1 builds and solves $MP(\underline{\mathscr{Q}}, \underline{\mathscr{P}})$, a MILP. In general, there can be computational overhead incurred when building such a MILP. Relatedly, note that $MP(\underline{\mathscr{Q}}, \underline{\mathscr{P}})$ is a relaxation that is strengthened at each iteration. Thus, when $MP(\underline{\mathscr{Q}}, \underline{\mathscr{P}})$ is solved by a Branch-and-Bound-based method, search decisions such as pruning nodes from the branch-and-bound tree remain valid from one iteration of Algorithm 1 to the next.

As a result, many implementations of Benders are not iterative as presented in Algorithm 1. Instead, these implementations solve a single MILP with a Branch-

Algorithm 1 Benders decomposition for stochastic network design

```
 1: Initialization:
 2: Set $\underline{\mathcal{Q}} = \underline{\mathcal{P}} = \emptyset$,
 3: while stopping criterion not met do
 4:     Solve $MP(\underline{\mathcal{Q}}, \underline{\mathcal{P}})$ for lower bound, $\underline{v}$, scenario cost estimates, $\bar{z}_s$, and design vector $\bar{y}$
 5:     for $s \in S$ do
 6:         Solve $SP(\bar{y})_s$ for cost $\bar{v}_s$ and flow variables $\bar{x}^s$
 7:         if infeasible then
 8:             Determine dual extreme ray $\bar{q}$ from set $\mathcal{Q}$ that is certificate of infeasibility of $SP(\bar{y})_s$
 9:             Set $\underline{\mathcal{Q}} = \underline{\mathcal{Q}} \cup \bar{q}$
10:         else
11:             if $\bar{z}_s < \bar{v}_s$ then
12:                 Determine dual extreme point $\bar{p}$ that is optimal for dual of $SP(\bar{y})_s$
13:                 Set $\underline{\mathcal{P}} = \underline{\mathcal{P}} \cup \bar{p}$
14:     if No extreme rays or points added to $MP(\underline{\mathcal{Q}}, \underline{\mathcal{P}})$ then
15:         Stop. Solution composed of $(\bar{y}, \bar{x}^1, \ldots, \bar{x}^{|S|})$ is optimal and optimal objective function value is $\underline{v}$.
```

and-Bound-based method, and, in the course of searching the Branch-and-Bound tree, dynamically generate Benders Feasibility and/or Optimality cuts. This style of implementation is in line with the view of Benders as a Cutting plane method. In this type of implementation, the search for violated Benders cuts is typically done at nodes in the branch-and-bound tree wherein feasible solutions to the MILP are discovered. However, it has also been observed that fractional designs prescribed by solving the linear relaxation of $MP(\underline{\mathcal{Q}}, \underline{\mathcal{P}})$ can be used to generate Benders cuts that are valid for model (10.32)–(10.35). Thus, an implementation could instead generate Benders cuts at every node within the Branch-and-Bound tree. This observation is the inspiration for another enhancement, which is to first solve the linear relaxation of model (10.32)–(10.35) with Benders, collect the Benders cuts generated when doing so, and then solve model (10.32)–(10.35), albeit with those cuts added to the formulation.

Other, general, enhancements focus on the process of generating Benders cuts. Some enhancements have focused on speeding up that process. For example, solving the $DSP(\bar{y})_s$ has been observed to provide many computational advantages. This is in part because the feasible region of the dual does not vary by scenario. Thus, for a given $\bar{y}$, the optimal basis found when solving $DSP(\bar{y})_s$ for scenario s can be used to warm-start the solution process of solving $DSP(\bar{y})_{s'}$ for scenario s'. Similarly, the feasible region of the dual subproblem does not depend on $\bar{y}$. This provides additional opportunities for re-using solution information from one subproblem solve to another.

Other enhancements have focused on generating Benders cuts that are likely to enable the overall algorithm to converge more quickly. This is in part due to the fact that the subproblem $SP(\bar{y})_s$ is often highly degenerate, and thus multiple optimal solutions to its dual exist. As such, the same design $\bar{y}$ may be used to generate multiple cuts, with some, often referred to as *Pareto-optimal cuts* guiding Benders towards a provably optimal solution much quicker than others. One method for generating such cuts is to solve an additional dual subproblem.

We next focus on enhancements proposed for solving stochastic network design problems. One of the weaknesses of the basic Benders method is that the reformulation (10.32)–(10.35) omits much of the structure that is inherent to this class of problem. Thus, some enhancements have focused on re-introducing that structure into the Master problem through valid inequalities that only involve the design variables, y. For example, as each commodity must depart its origin, any optimal design must include arcs that enable such flow. Based on this reasoning, the valid inequality $\sum_{(i,j)\in\mathscr{A}:i=o(k)} y_{ij} \geq 1$ can be added to (10.32)–(10.35). This inequality can be strengthened by recognizing the total demand originating at a node and computing the minimum number of arcs, L, that must be included in the design to carry that flow. When $L > 1$, the valid inequality can be strengthened by replacing the right-hand-side with L. Analogous inequalities can be added to (10.32)–(10.35) based on the destinations of commodities. This line of reasoning can be generalized to *cut-set*-type inequalities, which are similar in spirit to the inequalities just described, but consider subsets of nodes.

A more direct way to retain structure in (10.32)–(10.35) is to use what has been referred to as a *Partial Benders Decomposition.* In this approach, variables and constraints from scenarios are retained in the Master problem. As an example, consider a partition of the scenario set S into two sets, S_R, S_P. A Partial Benders Decomposition-based approach would solve the following Master problem:

$$\min \quad \sum_{(i,j)\in\mathscr{A}} f_{ij}y_{ij} + \sum_{s\in S_R}\sum_{k\in\mathscr{K}}\sum_{(i,j)\in\mathscr{A}} p_s c_{ij}^k x_{ij}^{ks} + \textstyle\sum_{s\in S_P} p_s z_s \tag{10.36}$$

$$\text{s.t.} \quad \sum_{j\in\mathscr{N}^+(i)} x_{ij}^{ks} - \sum_{j\in\mathscr{N}^-(i)} x_{ji}^{ks} = d_i^{ks}, \quad \forall i \in \mathscr{N}, k \in \mathscr{K}, s \in S_R \tag{10.37}$$

$$\sum_{k\in\mathscr{K}} x_{ij}^{ks} \leq u_{ij}^s y_{ij}, \quad \forall (i,j) \in \mathscr{A}, s \in S_R \tag{10.38}$$

$$\textstyle\sum_{k\in\mathscr{K}}\sum_{i\in\mathscr{N}} d_i^{ks}\alpha_{ik}^q + \sum_{(i,j)\in\mathscr{A}} u_{ij}^s y_{ij}\beta_{ij}^q \leq 0, \quad \forall q \in \mathscr{Q}, s \in S_P \tag{10.39}$$

$$\text{s.t.} \textstyle\sum_{k\in\mathscr{K}}\sum_{i\in\mathscr{N}} d_i^{ks}\alpha_{ik}^p + \sum_{(i,j)\in\mathscr{A}} u_{ij}^s y_{ij}\beta_{ij}^p \leq z_s, \quad \forall p \in \mathscr{P}, s \in S_P \tag{10.40}$$

$$y_{ij} \in \{0,1\}, \quad \forall (i,j) \in \mathscr{A}, \tag{10.41}$$

$$x_{ij}^{ks} \geq 0, \quad \forall (i,j) \in \mathscr{A}, k \in \mathscr{K}, s \in S_R, \tag{10.42}$$

$$z_s \geq 0, \forall s \in S_P. \tag{10.43}$$

While the idea of retaining explicit scenario information in the Master problem is general, it is particularly effective for classes of problems like network design wherein valid inequalities based on problem structure have been developed and implemented in off-the-shelf solvers. In addition to implying the commodity origin/destination-based valid inequalities discussed above, it has been shown that retaining well-chosen scenarios can completely eliminate the need to generate Benders Feasibility cuts. In addition, the Master problem can be strengthened with *artificial* scenarios that are appropriately constructed based on scenarios in S.

4.2 Progressive Hedging

Another solution approach that has been applied to efficiently solve model (10.13)–(10.16) is the progressive hedging method. This solution method also relies on the use of a decomposition strategy to separate the stochastic network design model into a series of subproblems that are both smaller and easier to solve. Specifically, model (10.13)–(10.16) is decomposed to produce a subproblem for each scenario that is included in set S, by relaxing the non-anticipativity constraints. The non-anticipativity constraints ensure that the two-stage structure of the model is followed. Specifically, these constraints enforce the fact that one cannot wait for the stochastic parameters to be observed to tailor the a priori decisions to the specific scenario that occurs. Thus, as specified in (10.13)–(10.16), a single network is designed in the first stage, represented by variables $y_{ij}, \forall (i,j) \in \mathscr{A}$, which is then used in the second stage regardless of the scenario that is observed.

If one is to relax the non-anticipativity constraints, then these constraints must first be explicitly formulated in the model. To do so, we start by duplicating for each $s \in S$, a set of design decisions $y_{ij}^s \in \{0,1\}, \forall (i,j) \in \mathscr{A}$, thus enabling a specific network to be fixed for each scenario that is considered. We further let y^s define the vector of all design decisions associated with the scenario $s \in S$. The non-anticipativity requirements can then be imposed by adding the following constraints:

$$y_{ij}^s = y_{ij}^{s'} \; \forall s, s' \in S, s \neq s', (i,j) \in \mathscr{A}.$$

The number of such constraints is quite high, i.e., for each possible arc a constraint is imposed to state that the selection decision made on this arc must be equal for all pairs of distinct scenarios. A more compact version of these contraints can be obtained by defining, in addition to $y_{ij}^s, \forall (i,j) \in \mathscr{A}$, a set of design decisions that will represent an overall network, i.e., $\overline{y}_{ij} \in \{0,1\}, \forall (i,j) \in \mathscr{A}$. We again define $\overline{y}$ as the vector containing all of the overall design decisions. Using the overall network design variables, the non-anticipativity constraints can be expressed as follows:

$$y_{ij}^s = \overline{y}_{ij} \; \forall s \in S, (i,j) \in \mathscr{A}.$$

By forcing all scenario networks to be equal to the overall network will again reinforce the non-anticipativity constraints.

An equivalent reformulation of (10.13)–(10.16) can now be obtained by explicitly adding the non-anticipativity constraints to the model:

$$\min \sum_{s \in S} p_s \left(\sum_{(i,j) \in \mathscr{A}} f_{ij} y_{ij}^s + \sum_{k \in \mathscr{K}} \sum_{(i,j) \in \mathscr{A}} c_{ij}^k x_{ij}^{ks} \right) \tag{10.44}$$

$$\text{s.t.} \quad \sum_{j \in \mathscr{N}^+(i)} x_{ij}^{ks} - \sum_{j \in \mathscr{N}^-(i)} x_{ji}^{ks} = d_i^{ks} \qquad \forall i \in \mathscr{N}, k \in \mathscr{K}, s \in S \tag{10.45}$$

$$\sum_{k\in\mathcal{K}} x_{ij}^{ks} \le u_{ij}^s y_{ij}^s \qquad \forall (i,j)\in\mathcal{A}, s\in S \tag{10.46}$$

$$y_{ij}^s = \overline{y}_{ij} \qquad \forall (i,j)\in\mathcal{A}, s\in S, \tag{10.47}$$

$$\overline{y}_{ij} \in \{0,1\}, y_{ij}^s \in \{0,1\} \qquad \forall (i,j)\in\mathcal{A}, s\in S \tag{10.48}$$

$$x_{ij}^{ks} \ge 0 \qquad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K}, s\in S. \tag{10.49}$$

The decomposition steps are then applied as follows. First, constraints (10.47) are relaxed through an augmented Lagrange strategy. We thus obtain the following objective function for the relaxed stochastic network design model:

$$\min (10.44) + \sum_{s\in S} p^s \Bigg(\sum_{(i,j)\in\mathcal{A}} \lambda_{ij}^s (y_{ij}^s - \overline{y}_{ij}) + \frac{1}{2} \sum_{(i,j)\in\mathcal{A}} \rho (y_{ij}^s - \overline{y}_{ij})^2 \Bigg), \tag{10.50}$$

where λ_{ij}^s, $\forall (i,j) \in \mathcal{A}$ and $s \in S$, define the Lagrange multipliers associated with the relaxed constraints (10.47) (we also set λ as the vector containing these multipliers) and ρ is the penalty value associated with the augmented term.

Considering the binary requirements of the first stage decision variables, the relaxed model can be simplified as follows:

$$\begin{aligned} \min \ & \sum_{s\in S} p^s \Bigg(\sum_{(i,j)\in\mathcal{A}} \big(f_{ij} + \lambda_{ij}^s - \rho\overline{y}_{ij} + \tfrac{\rho}{2}\big) y_{ij}^s + \sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^{ks} \Bigg) \\ & - \sum_{s\in S} \sum_{(i,j)\in\mathcal{A}} p^s \lambda_{ij}^s \overline{y}_{ij} + \sum_{(i,j)\in\mathcal{A}} \tfrac{1}{2} \rho \overline{y}_{ij} \\ \text{s.t.} \ & (10.45), (10.46), (10.48), (10.49). \end{aligned}$$

Regarding this model, an important observation to make is that if the overall design is set (i.e., if the variables $\overline{y}_{ij}$, $\forall (i,j) \in \mathcal{A}$ are fixed) then the model becomes scenario separable. In such a case, a series of deterministic (or single scenario) subproblems are obtained. Specifically, for each $s \in S$, we define:

$$\begin{aligned} \min \ & \sum_{(i,j)\in\mathcal{A}} \big(f_{ij} + \lambda_{ij}^s - \rho\overline{y}_{ij} + \tfrac{\rho}{2}\big) y_{ij}^s + \sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} c_{ij}^k x_{ij}^{ks} \\ \text{s.t.} \ & (10.45), (10.46), (10.48), (10.49). \end{aligned}$$

It should be highlighted that, through the present relaxation strategy, it is the fixed costs associated with the design decisions that are adjusted (by updating the Lagrange multipliers λ and the penalty ρ) to reinforce the non-anticipativity constraints.

The overall search process then proceeds by iteratively performing the following three-step approach: *Step 1:* solve each scenario subproblem separately, thus producing a specific (and possibly different) network (i.e., the networks y^s, $\forall s \in$

S); *Step 2:* using these networks derive an aggregate solution that represents the level of design consensus among the different scenario solutions obtained (i.e., the solution $\overline{y}$); *Step 3:* in each scenario subproblem, adjust the fixed costs associated with the design decisions to promote network consensus (i.e., all solutions to the subproblems converging to an identical network and thus enforcing the non-anticipativity constraints).

As indicated in *Step 2*, an aggregate solution needs to be obtained from the scenario networks. As was originally proposed, when the progressive hedging method was first introduced, the expectation can be used to derive this aggregation solution. Thus, one defines the following overall solution:

$$\overline{y}_{ij} = \sum_{s \in S} p_s y^s_{ij}, \ \forall (i, j) \in \mathscr{A}. \tag{10.51}$$

Considering the binary requirements imposed on the design decisions, by applying (10.51), one may fail to produce a feasible network and the overall solution process may not even converge. This being said, a feasible network can always be obtained as follows:

$$y^{\mathrm{M}\nu}_{ij} = \bigvee_{s \in S} y^{s\nu}_{ij}, \quad \forall (i, j) \in \mathscr{A}, \tag{10.52}$$

which would be referred to as the *Max design* (let $y^{\mathrm{M}\nu}$ denote the vector containing the max design variables at iteration ν), obtained by selecting the arcs that appear in at least one scenario solution. From solution (10.52), an upper bound can be derived for the overall model and this bound can be iteratively updated. However, such a solution is not necessarily appropriate to be used as the aggregate solution. Specifically, it may bias the search towards finding unduly large networks.

As for the value $\overline{y}_{ij}$, they indicate the level of consensus between the scenario networks with respect to the inclusion, or exclusion, of arc $(i, j) \in \mathscr{A}$. Specifically, for a given $(i, j) \in \mathscr{A}$, considering the scenario networks that were obtained y^s, $\forall s \in S$, by applying (10.51) then one of the following cases will be observed:

$$\overline{y}_{ij} = \begin{cases} 0 & \text{if } y^s_{ij} = 0, \forall s \in S, \\ 1 & \text{if } y^s_{ij} = 1, \forall s \in S, \\ 0 < \overline{y}_{ij} < 1 & \text{otherwise } y^s_{ij} = 1, \forall s \in \overline{S} \text{and } y^{s'}_{ij} = 0, \forall s' \in S \setminus \overline{S}. \end{cases}$$

The first two cases indicate that consensus is reached among the scenario networks to either exclude, or include, the specific arc that is considered. The third case shows that consensus has not yet been reached. In this case, the specific value to which $\overline{y}_{ij}$ is fixed provides the general trend that is observed among the scenario designs. If $\overline{y}_{ij} \approx 1$ then there is a general trend to include the arc in the overall network, while $\overline{y}_{ij} \approx 0$ indicates instead a general trend to exclude the arc. The level of consensus is lowest in the case where $\overline{y}_{ij} \approx 0.5$, where an overall trend cannot be deduced.

Therefore, the general idea behind *Step 3* is to adjust the fixed costs associated with each scenario subproblem to gradually incentivize consensus.

Let us consider a specific iteration of the progressive hedging method that is performed. We denote this iteration by the counting index ν. As soon as *Step 1* is completed, a series of scenario networks is obtained for the current iteration, i.e., $y^{s\nu}, \forall s \in S$. By then applying *Step 2*, the current aggregate solution is derived, i.e., $\overline{y}^{\nu} = \sum_{s \in S} p_s y^{s\nu}$. *Step 3* then looks to adjust the fixed costs defined in the scenario subproblems to promote consensus. Let value $f_{ij}^{s\nu}$ define the modified fixed cost of arc $(i, j) \in \mathscr{A}$ in the objective function of the subproblem associated with scenario $s \in S$ at the current iteration ν. To determine how $f_{ij}^{s\nu}$ should be modified, the difference between the scenario design decision regarding arc (i, j) should be compared to the associated decision in the aggregate solution. Specifically, if either $\overline{y}_{ij}^{\nu} = 0$ or $\overline{y}_{ij}^{\nu} = 1$, then consensus is reached overall scenario networks and $f_{ij}^{s\nu}$ should remain unchanged. In the case where $0 < \overline{y}_{ij}^{\nu} < 1$, then the following disjunction applies:

$$y_{ij}^{s\nu} < \overline{y}_{ij}^{\nu} \bigvee y_{ij}^{s\nu} > \overline{y}_{ij}^{\nu}.$$

If $y_{ij}^{s\nu} < \overline{y}_{ij}^{\nu}$, then one can deduce that $y_{ij}^{s\nu} = 0$ while $\overline{y}_{ij}^{\nu} > 0$. In this case, the value of $f_{ij}^{s\nu}$ should be decreased considering that the aggregate decision regarding the arc (i, j) indicates that there may be value (at least in some scenario networks) to include the arc (i.e., value $f_{ij}^{s\nu}$ is thus decreased to incentivize inclusion). Furthermore, a more pronounced decrease should be applied if $\overline{y}_{ij} \approx 1$, whereas a much milder decrease is warranted when $\overline{y}_{ij} \approx 0$. On the other hand, if $y_{ij}^{s\nu} > \overline{y}_{ij}^{\nu}$, then one can deduce that $y_{ij}^{s\nu} = 1$ while $\overline{y}_{ij}^{\nu} < 1$. In this case, the logic is reversed, the value of $f_{ij}^{s\nu}$ should be increased considering that the aggregate decision regarding the arc (i, j) indicates that there may be value (at least in some scenario networks) to exclude the arc (i.e., value $f_{ij}^{s\nu}$ is thus increased to incentivize exclusion). Once again, the extent to which the increase should be set needs to reflect the trend that is observed. Therefore, a higher increase is applied if $\overline{y}_{ij} \approx 0$, whereas a much smaller increase is applicable when $\overline{y}_{ij} \approx 1$. These are the general guidelines that are applied when adjusting the fixed costs within the scenario subproblems. Based on these guidelines, different strategies can be defined.

A first such strategy is directly defined via the augmented Lagrange method that is applied. Therefore, at iteration ν, the modified fixed costs are defined as $f_{ij}^{s\nu} = f_{ij} + \lambda_{ij}^{s\nu-1} - \rho^{\nu-1}\overline{y}_{ij}^{\nu-1} + \frac{\rho^{\nu-1}}{2}, \forall (i, j) \in \mathscr{A}, s \in S$. The fixed cost values are then adjusted by updating the Lagrange multipliers and the penalty value of the augmented term as follows:

$$\lambda_{ij}^{s\nu+1} \leftarrow \lambda_{ij}^{s\nu} + \rho^{\nu}(y_{ij}^{s\nu+1} - \overline{y}_{ij}^{\nu}), \forall (i, j) \in \mathscr{A}, \tag{10.53}$$

$$\rho^{\nu+1} \leftarrow \alpha\rho^{\nu}, \tag{10.54}$$

where $\alpha > 1$ is a given constant that is fixed a priori and ρ^0 is defined as a positive value such that $\rho^\nu \to \infty$ as the number of iterations ν increases. When setting these values it is important to consider that one seeks to obtain a solution process that will reach consensus in a gradual manner (i.e., if the solution differences are overly penalized too quickly, then this may lead to forcing consensus on a sub-optimal solution).

Another strategy can be defined using heuristic search principles. Specifically, a set of *global* adjustments can first be made based on the overall trend that is observed regarding the inclusion or exclusion of the arcs at a given iteration ν. This is done by first defining c^{low} and c^{high} as thresholds indicating whether or not the value of $\overline{y}_{ij}^{\nu}$ shows a strong trend to exclude or include the arc $(i, j) \in \mathscr{A}$, respectively. The *global* adjustments can then be defined as follows:

$$f_{ij}^{\nu+1} = \begin{cases} \beta f_{ij}^{\nu} & \text{if } \overline{y}_{ij}^{\nu} < c^{\text{low}}, \\ \frac{1}{\beta} f_{ij}^{\nu} & \text{if } \overline{y}_{ij}^{\nu} > c^{\text{high}}, \\ f_{ij}^{\nu} & \text{otherwise.} \end{cases} \tag{10.55}$$

It should be noted that the following conditions are applied on the values of the different parameters used in the previous strategy: $\beta > 1$, $0 < c^{\text{low}} < 0.5$, and $0.5 < c^{\text{high}} < 1$. Furthermore, in this case, f_{ij}^{ν} represents the modified fixed cost of arc (i, j) for all scenarios $s \in S$.

A second set of adjustments, referred to as *local*, can be performed at the level of the scenario subproblems to penalize the differences observed between the scenario networks and the overall solution. By defining c^{far} as the threshold at which point a modification of the fixed costs are applied, then the following *local* adjustments are defined:

$$f_{ij}^{s\nu+1} = \begin{cases} \beta f_{ij}^{\nu} & \text{if } |y_{ij}^{s\nu} - \overline{y}_{ij}^{\nu}| \geq c^{far} \text{ and } y_{ij}^{s\nu} = 1, \\ \frac{1}{\beta} f_{ij}^{\nu} & \text{if } |y_{ij}^{s\nu} - \overline{y}_{ij}^{\nu}| \geq c^{far} \text{ and } y_{ij}^{s\nu} = 0, \\ f_{ij}^{\nu} & \text{otherwise.} \end{cases} \tag{10.56}$$

Again, the following conditions are applicable on the parameters used in this strategy: $\beta > 1$ and $0.5 < c^{far} < 1$. In this case, $f_{ij}^{s\nu}$ represents the modified fixed cost of arc (i, j) in the scenario subproblem s at iteration ν.

This general idea of applying *local* adjustments can also be used to further restrict the scenario subproblems to promote solution consensus. Specifically, if $|y_{ij}^{s\nu} - \overline{y}_{ij}^{\nu}| \leq c^{\text{near}}$, given a consensus threshold $0 < c^{\text{near}} < 0.5$, then one may apply the following restriction: $y_{ij}^{s\nu+1} = y_{ij}^{s\nu}$, thus fixing the status of this arc for the next iteration. In turn, this will yield a smaller (and simpler) scenario subproblem to be solved.

It should be noted that, even when applying these various strategies, the solution process may fail to converge to a feasible network. Therefore, the progressive hedging method, when applied to solve model (10.13)–(10.16), is implemented by performing two general search phases. In the first phase, the model is decomposed into scenario subproblems and the search for a single network is performed through the iterative three-step approach of solving the scenario subproblems, finding the aggregated solution and adjusting the scenario subproblems to incentivize consensus. This phase is performed until the process either converges to a single feasible network, or, a stopping criteria is reached (e.g., maximum allotted time, maximum number of iterations without improvements, etc.).

If the first phase fails to produce a consensus network, then considering the last aggregate solution obtained, let us denote it by $\overline{y}^{\text{end}}_{ij}$, $\forall(i, j) \in \mathscr{A}$, there exists a set $\overline{\mathscr{A}} \subset \mathscr{A}$ such that $\overline{\mathscr{A}} = \{(i, j) \in \mathscr{A} \mid \overline{y}^{\text{end}}_{ij} = 0 \vee \overline{y}^{\text{end}}_{ij} = 1\}$. Set $\overline{\mathscr{A}}$ includes the arcs for which consensus was reached as to their status (either selected or not) in the overall network. Therefore, to produce a feasible solution, a second search phase is applied to establish the status on the arcs for which consensus was not reach, i.e., the search is performed on the space defined by the decision variables $y_{ij} \in \{0, 1\}$, $\forall(i, j) \in \mathscr{A} \setminus \overline{\mathscr{A}}$. Specifically, the following constraints are added to model (10.13)–(10.16): $y_{ij} = \overline{y}^{\text{end}}_{ij}$, $\forall(i, j) \in \overline{\mathscr{A}}$, and the resulting restriction is solved directly to obtain a network y^{end}, where the status of the remaining non-consensus arcs $\forall(i, j) \in \mathscr{A} \setminus \overline{\mathscr{A}}$ are fixed.

The progressive hedging method is summarized in Algorithm 2, where the **L** or **H** are two predicates indicating whether or not the Lagrange or heuristic fixed cost adjustment strategies are used, respectively, to solve model (10.13)–(10.16). On lines 1 to 10, the method initializes the parameters and evaluates the first aggregated solution. The first phase of the progressive hedging method is then performed on the lines 11–29. Finally, if consensus overall arcs could not be reached at the end of the first phase, then the second phase of the method is applied on the lines 30–35. Throughout the solution process, $BestSolution$ stores the best found network.

An important point to highlight regarding the progressive hedging method is that it enables algorithmic innovations that were developed to solve deterministic network design models to be leveraged to solve their stochastic counterparts. As it was clearly illustrated previously, the decomposition strategy that is applied here enables model (10.44)–(10.49) to be decomposed according to the scenarios $s \in S$, thus producing $|S|$ deterministic network design models. These are the scenario subproblems being solved each time line 24 is performed in Algorithm 2. Therefore, any method (e.g., exact, heuristic, matheuristic, etc.) that is available to solve the deterministic fixed-charge capacitated multicommodity network design model can be applied here to perform these resolutions (thus allowing for deterministic optimization innovations to be directly used in the stochastic settings).

Finally, it should be noted that the only modifications that are being made iteratively to these deterministic models are the adjustments to the fixed costs associated with the arcs, which are applied using either the **L** or **H** strategy. In this case, with the exception of the restrictions defined on lines 21–23 (which effectively reduce the size of the feasible regions of the scenario subproblems), the constraint sets of the deterministic models do not change throughout the progressive hedging

Algorithm 2 The progressive hedging method with fixed cost adjustment **L** or **H**

```
 1: Initialization:
 2: $\nu \leftarrow 0$;
 3: if L = TRUE then
 4:     $\lambda_{ij}^{s\nu} \leftarrow 0,\ \forall (i,j) \in A, \forall s \in S$;
 5:     $\rho^{\nu} \leftarrow \rho^{0}$;
 6: for $s \in S$ do
 7:     $f_{ij}^{s\nu} \leftarrow f_{ij}, \forall (i,j) \in \mathscr{A}$;
 8:     Solve the corresponding scenario subproblem;
 9: $\overline{y}_{ij}^{\nu} \leftarrow \sum_{s \in S} p^s y_{ij}^{s\nu},\ \forall (i,j) \in \mathscr{A}$;
10: $BestSolution \leftarrow y^{\mathrm{M}\nu}$;
11: First Phase:
12: while stopping criterion not met do
13:     $\nu \leftarrow \nu + 1$;
14:     if H = TRUE then
15:         Apply the global adjustments $f_{ij}^{\nu}, \forall (i,j) \in \mathscr{A}$ using equation (10.55);
16:     for $s \in S$ do
17:         if L = TRUE then
18:             $f_{ij}^{s\nu} \leftarrow f_{ij} + \lambda_{ij}^{s\nu-1} - \rho^{\nu-1}\overline{y}_{ij}^{\nu-1} + \frac{\rho^{\nu-1}}{2}, \forall (i,j) \in \mathscr{A}$;
19:         if H = TRUE then
20:             Apply the local adjustments $f_{ij}^{s\nu}, \forall (i,j) \in \mathscr{A}$ using equation (10.56);
21:         for $(i,j) \in \mathscr{A}$ do
22:             if $|y_{ij}^{s\nu-1} - \overline{y}_{ij}^{\nu-1}| \leq c^{\mathrm{near}}$ then
23:                 Add the constraint $y_{ij}^{s} = y_{ij}^{s\nu-1}$ to the corresponding scenario subproblem;
24:         Solve the corresponding scenario subproblem;
25:     $\overline{y}_{ij}^{\nu} \leftarrow \sum_{s \in \mathscr{S}} p^s y_{ij}^{s\nu}, \forall (i,j) \in \mathscr{A}$;
26:     if L = TRUE then
27:         $\lambda_{ij}^{s\nu} \leftarrow \lambda_{ij}^{s\nu-1} + \rho^{\nu-1}(y_{ij}^{s\nu} - \overline{y}_{ij}^{\nu-1}), \forall (i,j) \in \mathscr{A}$;
28:         $\rho^{\nu} \leftarrow \alpha\rho^{\nu-1}$;
29:     Update $BestSolution \leftarrow y^{\mathrm{M}\nu}$ if appropriate;
30: Second Phase:
31: if $\overline{\mathscr{A}} \neq \emptyset$ then
32:     for $(i,j) \in \overline{\mathscr{A}}$ do
33:         Add the constraint $y_{ij} = \overline{y}_{ij}^{\mathrm{end}}$ to (10.13)–(10.16);
34:     Solve the restricted (10.13)–(10.16) and obtain the solution $y^{\mathrm{end}}$;
35:     Update $BestSolution \leftarrow y^{\mathrm{end}}$ if appropriate.
```

search process. Therefore, at a given iteration ν, a method that is applied to solve the deterministic network design model associated with a scenario $s \in S$ can easily be enhanced by using the information generated through the solution processes that were applied to solve the model in the first $\nu - 1$ iterations. Thus, one can warm start the search process applied at the current iteration by using solutions, valid inequalities, or any other useful information generated in the previous steps.

5 Conclusions and Perspectives

The network design problem is a general and classical optimization problem that has been used to inform strategic, tactical, and operational planning processes in many industries. Two industries that have inspired a great deal of research on network design are freight transportation and telecommunications. However, in these industries (and others), all information regarding the operational contexts in which a plan (e.g. a design) will be executed are rarely known with certainty. This chapter presented a primer on how to incorporate such uncertainty into optimization models for network design.

The chapter began by introducing two of the most commonly-used paradigms for explicitly recognizing uncertainty in an optimization model, and illustrated the use of those paradigms on a specific problem, the stochastic fixed-charge capacitated multicommodity network design problem. The first paradigm, stochastic programming with recourse, enables the modeler to define what decisions are made before information is revealed as well as what decisions can be made after to best adapt to that revealed information (i.e. *recourse decisions*). However, in some contexts, a modeler may not want to define such recourse actions. Instead, the goal is to produce a design for which the probabilities of meeting some performance standards meet or exceed some thresholds. Such a setting can be addressed with the second paradigm, stochastic programming with probabilistic constraints.

Having presented these paradigms for how to recognize uncertainty in network design problems, the chapter then focused on how to represent uncertainty in the optimization models. Both paradigms are often implemented by approximating probability distributions with sets of discrete scenarios. In these implementations, some model parameters are seen as random variables, and a given scenario represents a realization of those random variables. This chapter illustrated the implementation of each paradigm given such a set of scenarios. Such an implementation necessitates the generation of a set of scenarios. As much of the literature on scenario generation is not specific to network design, the chapter did not discuss specific methods for generating scenarios. Instead, it discussed some important considerations when generating scenarios, including knowing when a set of scenarios sufficiently represents a distribution and challenges that can arise when collecting sufficient data to generate a sufficiently representative set of scenarios.

The chapter then presented two solution strategies commonly used for solving scenario-based optimization models. While the strategies are general, each has shown promise at solving instances of stochastic network design-type models. The first, Benders decomposition, is the basis of an exact method, in that it is guaranteed to yield an optimal solution and a certificate of that solution's optimality. The second, progressive hedging, is only the basis of an exact method for certain classes of models (e.g. stochastic linear programs). However, as most network design problems do not fall into one of those classes, a progressive hedging-based strategy is effectively a heuristic. In addition to presenting the base strategies, the chapter also presented some of the most effective enhancements that have been proposed for solving instances of stochastic network design models.

While stochastic network design as a general research topic has received a great deal of attention, and the state of the art has advanced tremendously, there is still much work to be done. Typically, networks are designed to support the routing of some form of demand. Much of the literature on stochastic network design has focused on models that only recognize uncertainty in those demands. However, in many settings, the designed network involves arcs that have capacities. Far less literature has considered models that recognize uncertainty in the capacity that is established when an arc is installed. Such uncertainty is particularly prevalent when an organization is designing a network that leverages assets that are shared with other organizations. In such situations, the capacity available to one organization is partly a function of the capacity used by another, which likely will not be known at the time of design. Of course, in such settings there would likely be uncertainty in both capacities and demands.

Relatedly, most stochastic programming-based approaches proposed in the literature have considered two stage models. In such models, the design is established in the first stage, information (usually regarding demands) is revealed, and then demands are routed. However, in reality, designs are often altered, and at potentially regular intervals, with some arcs removed from the network and others added. A common reason for periodically altering a design is seasonality in demand patterns. However, in such a setting, treating each season as a separate two stage problem would allow the network to completely change from one season to the next. In many settings, such a dramatic change in operations would not be desirable, and may not even be feasible. In those settings, a multi-stage model would be more appropriate, as it would allow the model to explicitly constrain the difference in the network from one season to the next. Such a model has received little attention in the literature.

Turning to solution methods, there are still many industrial-sized stochastic network design problems that can not be solved to optimality in run-times that are reasonable for practical planning. This is typically due to instance size. In some settings, the instance of the deterministic equivalent of the problem itself may be too large. However, the nature of scenario-based formulations can be a root cause of large model instances. While scenario aggregation-based approaches have shown promise, there is more work to be done, particularly with respect to stochastic network design. Relatedly, decomposition schemes that are similar in spirit, albeit different, than what is used in one of the two basic strategies (Benders, Progressive hedging) have shown promise. However, these strategies are often static in nature. Namely, the decomposition is determined when the algorithm begins, and then the algorithm proceeds. Dynamically changing the decomposition during algorithm execution is a promising avenue for future research.

Lastly, a barrier to optimization models impacting practice is that decision-makers are often wary to implement suggestions from a "black box." Such reticence may be even more prevalent when an optimization model explicitly recognizes uncertainty, as the model can recognize the impact of a decision across multiple scenarios much better than a human can. Stochastic network design models have a greater chance of impacting practice if there are mechanisms for identifying and communicating why such a model would prescribe flexibility in some form. There have been studies dedicated to understanding the differences that are observed between networks designed by solving a deterministic model versus networks designed by solving a stochastic model. Such efforts have been helpful to assess how uncertainty affects the decision-making in network design. However, there is still a lot to be done on this important issue.

6 Bibliographical Notes

The number of studies dedicated to how uncertainty should be addressed in the context of designing networks has been steadily growing. A comprehensive literature review on this subject was produced by Klibi et al (2010), in which the previously mentioned three-group classification of the sources of uncertainty that affect network design was presented. One can now find in the literature an abundant number of specific studies, where stochastic network design methodologies (that apply the presented modelling paradigms) are developed for a variety of applications in logistics, transportation and telecommunications, to name a few.

In the context of logistics, Alonso-Ayuso et al (2003) and Santoso et al (2005) were among the first to propose comprehensive methods to solve stochastic supply chain network design models. Since these original works, a steady stream of research has produced stochastic optimization methods to solve logistics planning problems in a variety of settings, e.g., reverse logistics, see Trochu et al (2020), maintenance planning, see Schrotenboer et al (2020), medical supply chain management, see Pishvaeea et al (2014), humanitarian relief distribution, see Noyan et al (2016), and many other cases. The use of stochastic network design methods has also been prevalent in the context of solving transportation planning problems. For example, two-stage stochastic models have been successfully applied to tackle scheduled service network design problems relevant to the planning of transportation operations when acquiring and managing resources, see Hewitt et al (2019), in the context of operating two-tiered freight distribution systems, see Crainic et al (2016a) and to conduct freight transportation when rerouting is considered as part of the recourse actions, see Bai et al (2014). Telecommunication applications have also been the source of many studies conducted on applying stochastic optimization to solve network design problems, the reader is referred to Gaivoronski (2006) for a general overview of this field.

As for how two-stage models can be used in the context of solving multi-stage stochastic network design problems, we refer the reader to Cadarso et al (2018) for a specific example of this. Within a multi-stage model used to formulate a strategic rail network design problem, the authors imbed a series of two-stage scenario trees to approximate the operations that are conducted within the network. In addition, Powell (2014) provides a clear overview of how two-stage formulations can be applied to perform broader sequential decision processes that involve uncertainty.

Regarding the scenario generation methods, as previously stated, there is a vast literature on the subject. Nonetheless, we can refer the reader to Kaut et al (2012) for thorough introduction to the field. Furthermore, the stability assessment methods to verify the quality of samples were developed and clearly presented in Kaut and Wallace (2007).

As for the solution methods that were presented in this chapter, the reader will find a thorough review of the Benders decomposition algorithm in Rahmaniani et al (2017). Furthermore, a specialized version of the Benders algorithm was developed for the stochastic demand variant of the fixed-charge capacitated multicommodity network design problem in Rahmaniani et al (2018). The partial Benders decomposition strategy was developed in Crainic et al (2021) and applied to the more general variant of the problem (i.e., where both the demands and the capacities are stochastic). The progressive hedging method for the fixed-charge capacitated multicommodity network design problem with stochastic demands was introduced in Crainic et al (2011). A follow-up on this original work, that improved the scenario decomposition strategy that is applied to implement the solution method (i.e., creating multi-scenario subproblems that enable a more efficient search for a good consensus solution), was performed by Crainic et al (2014).

Finally, we can highlight some of the studies that have been performed to understand the differences that occur in the characteristics of the networks obtained when solving stochastic optimization models when compared to solving deterministic ones. A first such study is due to Lium et al (2009), where the authors illustrated how, when solving stochastic network design models, consolidation strategies applied on multiple paths connecting the origins and destinations of commodities enabled more efficient networks to be obtained. This being said, networks obtained by solving deterministic and stochastic models can share common characteristics, see Thapalia et al (2012b), Thapalia et al (2012a) and Wang et al (2018). Considering that this observations can actually be generalized to other problem settings, see Maggioni and Wallace (2012), pursuing this type of study appears to be a fruitful avenue of research.

References

Alonso-Ayuso, A., Escudero, L. F., Garin, A., Ortuno, M. T., & Perez, G. (2003). An approach for strategic supply chain planning under uncertainty based on stochastic 0–1 programming. *Journal of Global Optimization, 26*(1), 97–124.

Bai, R., Wallace, S. W., Li, J., & Chong, A. Y. (2014). Stochastic service network design with rerouting. *Transportation Research B: Methodological, 60*, 50–65.

Cadarso, L., Escudero, L. F., & Marìn, A. (2018). On strategic multistage operational two-stage stochastic 0–1 optimization for the rapid transit network design problem. *European Journal of Operational Research, 271*(2), 577–593.

Crainic, T. G., Errico, F., Rei, W., & Ricciardi, N. (2016a). Modeling demand uncertainty in two-tier city logistics tactical planning. *Transportation Science, 50*(2), 559–578.

Crainic, T. G., Fu, X., Gendreau, M., Rei, W., & Wallace, S. W. (2011). Progressive hedging-based metaheuristics for stochastic network design. *Networks, 58*(2), 114–124.

Crainic, T. G., Hewitt, M., Maggioni, F., & Rei, W. (2021). Partial Benders Decomposition: General Methodology and Application to Stochastic Network Design. *Transportation Science, 55*(2), 414–435.

Crainic, T. G., Hewitt, M., & Rei, W. (2014). Scenario grouping in a progressive hedging-based meta-heuristic for stochastic network design. *Computers & Operations Research, 43*, 90–99.

Gaivoronski, A. A. (2006). Stochastic optimization in telecommunications. In M. C. Resende, & P. M. Pardalos (Eds.), *Handbook of optimization in telecommunications* (pp. 761–799). Boston, MA, USA: Springer.

Hewitt, M., Crainic, T. G., Nowak, M., & Rei, W. (2019). Scheduled service network design with resource acquisition and management under uncertainty. *Transportation Research Part B: Methodological, 128*, 324–343.

Kaut, M., King, A. J., & Wallace, S. W. (2012). Scenario-tree generation. In *Modeling with stochastic programming*. New York, NY: Springer.

Kaut, M., & Wallace, S. W. (2007). Evaluation of scenario-generation methods for stochastic programming. *Pacific Journal of Optimization, 3*(2), 257–271.

Klibi, W., Martel, A., & Guitouni, A. (2010). The design of robust value-creating supply chain networks: A critical review. *European Journal of Operational Research, 203*(2), 283–293.

Lium, A. G., Crainic, T. G., & Wallace, S. W. (2009). A study of demand stochasticity in service network design. *Transportation Science, 43*(2), 144–157.

Maggioni, F., & Wallace, S. W. (2012). Analyzing the quality of the expected value solution in stochastic programming. *Annals of Operations Research, 200*(1), 37–54.

Noyan, N., Balcik, B., & Atakan, S. (2016). A stochastic optimization model for designing last mile relief networks. *Transportation Science, 50*(3), 1092–1113.

Pishvaeea, M. S., Razmib, J., & Torabi, S. A. (2014). An accelerated Benders decomposition algorithm for sustainable supply chain network design under uncertainty: A case study of medical needle and syringe supply chain. *Transportation Research Part E: Logistics and Transportation Review, 67*, 14–38.

Powell, W. B. (2014). Clearing the jungle of stochastic optimization. In *INFORMS TutORials in operations research* (pp. 109–137). Catonsville, MD, USA: Informs.

Rahmaniani, R., Crainic, T. G., Gendreau, M., & Rei, W. (2017). The Benders decomposition algorithm: A literature review. *European Journal of Operational Research, 259*(3), 801–817.

Rahmaniani, R., Crainic, T. G., Gendreau, M., & Rei, W. (2018). Accelerating the Benders decomposition method: Application to stochastic network design problems. *SIAM Journal on Optimization, 28*(1), 875–903.

Santoso, S., Ahmed, S., Goetschalckx, M., & Shapiro, A. (2005). A stochastic programming approach for supply chain network design under uncertainty. *European Journal of Operational Research, 16*(1), 96–115.

Schrotenboer, A. H., Ursavas, E., & Vis, I. F. (2020). Mixed integer programming models for planning maintenance at offshore wind farms under uncertainty. *Transportation Research Part C: Emerging Technologies, 112*, 180–202.

Thapalia, B. K., Kaut, M., Crainic, T. G., & Wallace, S. W. (2012a). Single-commodity network design with random edge capacities. *European Journal of Operational Research, 220*(2), 394–403.

Thapalia, B. K., Wallace, S. W., Kaut, M., & Crainic, T. G. (2012b). Single source single-commodity stochastic network design. *Computational Management Science, 9*(1), 139–160.

Trochu, J., Chaabane, A., & Ouhimmou, M. (2020). A carbon-constrained stochastic model for eco-efficient reverse logistics network design under environmental regulations in the crd industry. *Journal of Cleaner Production, 245*(1), 1–16.

Wang, X., Crainic, T. G., & Wallace, S. W. (2018). Stochastic network design for planning scheduled transportation services: The value of deterministic solutions. *INFORMS Journal on Computing, 31*(1), 153–170.

Chapter 11
Robust Network Design

Arie M. C. A. Koster and Daniel R. Schmidt

1 Introduction

Designing a network is usually done on the basis of a forecast of the future/expected demand. Such a forecast will by definition not be an accurate representation of the reality. If the design process involves long-term and/or strategic decisions, the quality of the forecast determines the feasibility of the network design for its future purpose. Where an increase of the forecast values might be a simple solution to this problem, it is obvious that such a line of action might be too rough and hence unnecessary costly. Robust optimization offers a more informed alternative in such cases.

In the following, we first introduce the basics of robust optimization. Next, we survey its application to single- and multicommodity network design. At appropriate times, extensions of the basic idea of robust optimization are also introduced.

A. M. C. A. Koster (✉)
Lehrstuhl II für Mathematik, RWTH Aachen University, Aachen, Germany
e-mail: koster@math2.rwth-aachen.de

D. R. Schmidt
Institute for Computer Science V, University of Bonn, Bonn, Germany
e-mail: daniel.schmidt@uni-bonn.de

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_11

2 Robust Optimization

2.1 *What Is Robust Optimization?*

According to a text book by Ben-Tal et al. (2009), robust optimization is a "*methodology for handling optimization problems with uncertain data*". We extend this notion of robustness and say that a solution to an optimization problem is robust if it is feasible for a prescribed range of scenarios rather than in a single situation. Let us illustrate this concept with an example. Suppose we are to plan a network for an internet provider. We are provided with forecasts for the planning period and as a first step, we convert the forecasts into hard numbers for our problem input. This step will almost certainly introduce rounding errors and is prone to rule out potentially useful solutions. After this "rounding" step, we run an *exact* optimization algorithm – but only *after* we introduced errors and inaccuracies! How can we make sure that this solution is even feasible for the real (unknown) network requirements? There is another problem with our approach. We might be planning a network that sees different usage scenarios throughout the day, and while classical optimization can find a network for each scenario, it lacks the methodology to find a network that works in *all* of them. The methodology in this chapter gives us the ability to find solutions that are robust against imprecisions in the input and shifting use cases. Among other things, it will let us cope with inprecise numbers and lets us plan a network that can support different traffic peaks without requiring that all peaks can be handled simultaneously. Throughout, our approach will be as follows. First, we need to identify a set of possible network configurations or scenarios. This set is called the *uncertainty set*. Then, we will look for *worst-case* robust solutions, i.e. solutions that are feasible no matter which scenario occurs. The challenge here is to carefully select an appropriate uncertainty set: The broader the set, the more expensive our solution becomes. Still, if we were to accept that some solution is *not* feasible in all scenarios, we would accept that in some scenarios we violate our side constraints. Thus, a worst-case model is appropriate if in the application, safety is critical and a failure of the optimized system is not permitted or more expensive than guarding against it. If we are sure that all parameters realizations will occur eventually or if the probability distribution of the realizations is not known, then worst-case robustness is a good modeling choice as well (if only for the lack of alternatives).

2.2 *Chance-Constrained Model*

Robust optimization can be viewed as a specialization of chance-constrained optimization. As it is often impossible to map all possible inputs onto the uncertainty set, there remains a (hopefully small) chance that a robust solution is infeasible. Stated otherwise, a chance-constrained optimization model can be reformulated as a robust optimization model by determining an uncertainty set guaranteeing that solutions to the robust optimization model satisfy the original constraints with high probability. More formally, assuming uncertainties in the constraint matrix of an arbitrary linear program $\min\{c^{\mathrm{T}}x \mid Ax \geq b, x \geq 0\}$ only, a chance-constrained optimization program in its general form is:

$$\text{Minimize} \qquad \sum_{j=1}^{n} c_i x_i \tag{11.1}$$

$$\text{Subject to} \quad \Pr\left(\sum_{j=1}^{n} a_{ij}x_j \geq b_i\right) \geq 1 - \epsilon_i \quad \forall\, i = 1, \ldots, m \tag{11.2}$$

$$x_j \geq 0 \qquad \forall\, j = 1, \ldots, n. \tag{11.3}$$

Here, $\epsilon_i > 0$ is the maximum probability of violating the i-th constraint and the matrix entries a_{ij} are no longer deterministic values, but random variables following a (possibly unknown) distribution. Alternatively, all constraints can be considered jointly in a single chance constraint:

$$\Pr\left(\sum_{j=1}^{n} a_{ij}x_j \geq b_i \;\forall\, i = 1, \ldots, m\right) \geq 1 - \epsilon \tag{11.4}$$

In this case, a single value $\epsilon > 0$ specifies the maximum probability that a chance-constrained solution is infeasible.

Chance-constrained models are often difficult to solve, in particular if information on the probability distribution is not (or only limitedly) available. Tractability is further restricted by dependencies between the random variables. In the context of network design, we refer to Pascali (2009) for a chance-constrained approach. In the context of broadband wireless networks, we are aware of another work reducing the model to a deterministic problem in case of independent random variables, see Claßen et al. (2014). Alternatively, by following a robust optimization approach, we sometimes can guarantee that the solution satisfies the inequalities with high probability, either in theory, or in practice (by evaluating historical data). Therefore, we next describe a number of commonly used uncertainty sets.

2.3 Interval Uncertainty

In many cases, parts of the constraint matrix are based on physical measurements or forecasts and are thus not known with arbitrary precision. To capture this kind of uncertain input, assume that each coefficient a_{ij} of A has a nominal value $\bar{a}_{ij}$ (the value that was measured or predicted) and that the true value for a_{ij} can deviate by at most $\hat{a}_{ij} \geq 0$ from our nominal choice. We define an uncertainty set $\mathcal{U}_I$ that consists of all matrices A with coefficients $a_{ij} \in [\bar{a}_{ij} - \hat{a}_{ij}, \bar{a}_{ij} + \hat{a}_{ij}]$:

$$\mathscr{U}_I := \left\{ (a_{ij})_{i,j=1}^{m,n} \in \mathbf{R}^{m \times n} \;\middle|\; a_{ij} \in [\bar{a}_{ij} - \hat{a}_{ij}, \bar{a}_{ij} + \hat{a}_{ij}] \; \forall\, i, j \right\}. \tag{11.5}$$

Since there is no coupling between the individual coefficients, the worst-case scenario occurs when all coefficients deviate in the worst possible way. This happens when a_{ij} is set to $\bar{a}_{ij} - \hat{a}_{ij}$ for all i, j (assuming a system of type $Ax \geq b$) and results in the following program.

$$\text{Minimize} \quad \sum_{j=1}^{n} c_i x_i \tag{11.6}$$

$$\text{Subject to} \quad \sum_{j=1}^{n} (\bar{a}_{ij} - \hat{a}_{ij}) x_j \geq b_i \quad \forall\, i = 1, \ldots, m \tag{11.7}$$

$$x_j \geq 0 \qquad \forall\, j = 1, \ldots, n. \tag{11.8}$$

This is a classical result by Soyster (1973). As an example, suppose that our input numbers were measured with an accuracy of 1%. Then, we set $\bar{a}_{ij}$ to the value that was measured and let $\hat{a}_{ij} = 0.01 \cdot \bar{a}_{ij}$ for all i, j.

2.4 Budget Uncertainty

In a practical setting, it is unlikely that all coefficients deviate in the worst possible way at the same time. In consequence, solutions from the interval uncertainty model tend to be unnecessarily costly. To obtain a less conservative model, we assume an uncertainty budget that limits the total deviation. Changing the budget will allow us to control the conservatism of the model.

In order to control the level of robustness, let us introduce a parameter vector $\Gamma \in \mathbf{Z}^m_{\geq 0}$ whose i-th entry Γ_i decides how many coefficients of the i-th constraint may deviate from their nominal value at the same time.[1] We define the following set of uncertain matrices based on the choice of Γ.

[1]The model can be extended to choose a fractional $\Gamma \in \mathbf{R}^m_{\geq 0}$ with the interpretation that $\lfloor \Gamma_i \rfloor$ coefficients of constraint i deviate maximally and a single coefficient a_{ij} of constraint i deviates by the remaining amount $(\Gamma_i - \lfloor \Gamma_i \rfloor)\hat{a}_{ij}$. The extension is straight-forward, yet omitting it makes the exposition significantly easier. For the full model, see Bertsimas and Sim (2004).

$$\mathscr{U}_B(\Gamma) := \left\{ (a)_{i,j=1}^{m,n} \in \mathbf{R}^{m\times n} \;\middle|\; \begin{array}{ll} S_1, \dots, S_m \subseteq \{1, \dots, n\} & \\ |S_i| \leq \Gamma_i \;\forall\, i = 1, \dots, m & \\ a_{ij} \in [\bar{a}_{ij} - \hat{a}_{ij}, \bar{a}_{ij} + \hat{a}_{ij}], & \text{if } j \in S_i \\ a_{ij} = \bar{a}_{ij}, & \text{otherwise} \end{array} \right\}.$$

We say that a solution is Γ-robust if it is feasible for all $A \in \mathscr{U}_B(\Gamma)$. By increasing some Γ_i, we increase the robustness as well as the cost of the solution; this is what we call the price of robustness. In this way, solving the model for different values of Γ allows us to find a good trade-off between robustness and cost; the extreme cases being the interval model (set $\Gamma_i = n$ for all $i = 1, \dots, m$) or a non-robust model (set $\Gamma_i = 0$ for all $i = 1, \dots, m$). We have the following program with respect to $\mathscr{U}_B(\Gamma)$.

$$\text{Minimize} \quad \sum_{j=1}^{n} c_j x_j \tag{11.9}$$

$$\text{Subject to} \quad \sum_{j=1}^{n} \bar{a}_{ij} x_j \;-\; \max_{\substack{S \subseteq \{1,\dots,n\} \\ |S| \leq \Gamma_i}} \sum_{j \in S} \hat{a}_{ij} x_j \geq b_i \quad \forall\, i = 1, \dots, m \tag{11.10}$$

$$x_j \geq 0 \qquad \forall\, j = 1, \dots, n \tag{11.11}$$

If we fix a selection S of deviating coefficients in any constraint i of (11.9)–(11.11), then this constraint is most restrictive if a_{ij} deviates to the lower bound $\bar{a}_{ij} - \hat{a}_{ij}$ for all $j \in S$, as $x_j \geq 0$. This is modeled by the reformulated constraint (11.10). Program (11.9)–(11.11) can be casted into a linear program by replacing the inner optimization problem by its dual; a method that we shall see in more detail in Sect. 5.3.

The budget uncertainty model has nice theoretical properties with respect to chance-constrained optimization: Let $\tilde{A}$ result by randomly (but symmetrically) perturbing the coefficients of the original matrix A while obeying the maximum deviations. Then, the probability that a Γ-robust solution violates the i-th constraint of $\tilde{A}x \geq b$ is bounded by $\exp(-\Gamma_i^2/2n)$; independently of the distribution of the perturbation.[2]

[2] A tighter, but more involved bound can be shown; see Bertsimas and Sim (2004).

2.5 Polyhedral Uncertainty and the Robust Counterpart

To model interval and budget uncertainty, we have collected all possible realizations of the constraint matrix A in an uncertainty set $\mathscr{U} \subseteq \mathbf{R}^{m\times n}$ and looked for a solution $x \geq 0$ that is robust feasible, i.e., one that satisfies $Ax \geq b$ for all choices of $A \in \mathscr{U}$. In general, we can choose any closed, bounded set $\mathscr{U}$ as our uncertainty set. Furthermore, we can always replace $\mathscr{U}$ by its convex hull conv $\mathscr{U}$ without changing the feasible region of the uncertain program. In fact, we shall assume that $\mathscr{U}$ is a polytope in the following. Given an arbitrary linear program $\min\{c^{\mathrm{T}}x \mid Ax \geq b, x \geq 0\}$ and a polytope $\mathscr{U}$, we call the system

$$\text{Minimize} \quad \sum_{j=1}^{n} c_j x_j \tag{11.12}$$

$$\text{Subject to} \quad \sum_{j=1}^{n} a_{ij} x_j \geq b_i \;\; \forall\, i = 1, \ldots, m, \;\; \forall\, (a_{ij})_{i,j=1}^{m,n} \in \mathscr{U} \tag{11.13}$$

$$x_j \geq 0 \qquad \forall\, j = 1, \ldots, n \tag{11.14}$$

the robust counterpart of our original linear program. In order to model an uncertain right-hand side b of a linear program as well, we can introduce a fixed auxiliary variable and move b to the constraint matrix.

Polynomial time optimization over the feasible region of (11.12)–(11.14) is equivalent to having a polynomial time separation algorithm for the feasible region. This is an algorithm that decides whether a given point $x \in \mathbf{R}^n_{\geq 0}$ is feasible for (11.12)–(11.14) and if not, yields a scenario $A \in \mathscr{U}$ and an index $i \in 1, \ldots, m$ such that $\mathrm{row}_i(A)^{\mathrm{T}}x < b_i$, where $\mathrm{row}_i(A)$ denotes the i-th row of A. Such a separation algorithm for the robust counterpart exists if a separation oracle for $\mathscr{U}$ exists; indeed, to separate x^* from the feasible region of (11.12)–(11.14), it suffices to solve

$$b_i^* := \min\left\{\mathrm{row}_i(A)^{\mathrm{T}}x^* \;\middle|\; A \in \mathscr{U}\right\} \tag{11.15}$$

for all $i = 1, \ldots, m$. If for some i we have $b_i^* < b_i$, then there is a separating inequality $\mathrm{row}_i(A)^{\mathrm{T}}x \geq b_i$. Yet, solving (11.15) is possible in polynomial time if and only if there is a polynomial time separation algorithm for $\mathscr{U}$.

Theorem 1 (Ben-Tal and Nemirovski (1999)) *Let $\mathscr{U} \subseteq \mathbf{R}^{m\times n}$ and let $b \in \mathbf{R}^m$. Then the robust counterpart*

$$\min\left\{c^{\mathrm{T}}x \;\middle|\; Ax \geq b,\; x \geq 0,\; \forall\, A \in \mathscr{U}\right\}$$

is solvable in time polynomial in m and n if there is separation algorithm for $\mathscr{U}$ with a running time polynomial in m and n. □

Thus, we can solve worst-case robust linear programs polynomially if we can separate polynomially over the uncertainty set.

If the uncertainty polytope is given in its vertex description $\mathscr{U} = \text{conv}\{A_1, \ldots, A_k\}$ then separation is always possible in time polynomial in n, m and k (although k may be exponentially large in m or n). We say that $\mathscr{U}$ is a *discrete set* in this case.

If the uncertainty polytope is described by a system of linear inequalities, a dualization approach similar to the budget uncertainty case can be applied.

2.6 Multi-stage Robustness

In the classical worst-case robust model, we take all the decisions in a *single stage* before we know the realization of the uncertain parameters. This modelling is not always desirable: We could instead imagine fixing some variables here-and-now while adjusting other variables once (part of) the uncertain parameters have realized. As an example in the network design context, we might decide on the capacities in a first stage (without knowing the realizations of the uncertain demands) and postpone the routing of the traffic to a point in time where the demands are known with certainty. Or, we might conservatively *buy* parts of the network in advance and *rent* additional capacity as needed. Depending on fewer uncertain parameters, such *multi-stage* models allow for less expensive solutions without sacrificing their robustness. The price for the additional flexibility is a computionally harder model, as even models with two stages of robustness tend to be NP-hard to solve. A middle ground is achieved by assuming a tractable dependence between the uncertain parameters and the adjustable, later stage variables. Models with *affine robustness* assume that the adjustable variables can be computed as affine functions from the uncertain data. *Recoverable robustness* more generally asks for a tractable algorithm that computes feasible values for all adjustable variables given the first stage decisions and the parameter realization.

3 Robust Network Designs

In the sequel, we will apply the robust optimization approach to network design. As opposed to the introductory Chap. 2, we suppose that $\mathscr{G} = (\mathscr{N}, \mathscr{E})$ is an *undirected* graph with a node set $\mathscr{N}$ and a set of potential edges $\mathscr{E} \subseteq \binom{\mathscr{N}}{2}$. To emphasize the difference, we use the notation $\{i, j\}$ to distinguish an undirected edge between $i, j \in \mathscr{N}$ from the directed arcs (i, j) and (j, i). Then, a flow f on $\mathscr{G}$ assigns a flow value f_{ij} and f_{ji} to both possible orientations of each edge $\{i, j\}$ and a capacity vector $u \in \mathbf{R}^E$ admits a flow f if $f_{ij} + f_{ji} \leq u_{ij}$. Following the robust optimization paradigm, we consider a setting where the supplies / demands of the nodes in the graph are uncertain. The *robust network design problem* is the task to select a capacity $u_{ij} \geq 0$ for each edge $\{i, j\}$ such that *all* possible demand realizations can be satisfied while minimizing the total costs for installing the capacities.

Exactly *how* the demands are satisfied will vary throughout the chapter. We will first consider some single-commodity models and then generalize to multiple commodities. Given that we optimize over multiple scenarios, another modeling choice arises: Do we need to fix the routing of the demands before we know which scenario will realize or are we allowed to select a suitable routing once we know the scenario realization? In the former case, we need to compute a *routing template*, i.e. [0, 1]-valued flow. Given an arc (i, j), we interpret the template flow $f_{ij} \in [0, 1]$ as the percentage of the demand that is routed along (i, j). In the terms of the previous section, this yields a single-stage optimization problem and we say that we model a *static routing* in this case. If in the latter case, we are allowed to choose the routing *after* a scenario has realized, we perform a two-stage optimization. This is known as a *dynamic routing*. In general, dynamic routings offer more flexibility and thus, cheaper solutions. They are, however, much harder to compute and do not fit all practical applications. Models in between static and dynamic routing exist; see for instance Poss and Raack (2013).

4 Single-Commodity Formulations

In Chap. 2, Sect. 2, we defined a deterministic (i.e., non-robust) single-commodity flow by saying that every node has a demand w_i of the single commodity. We generalize this notion by introducing an uncertainty set $\mathscr{U} \subseteq \mathbf{R}^{\mathscr{N}}$ such that any scenario $w \in \mathscr{U}$ defines a demand w_i for each node. As before, we say that $i \in \mathscr{N}$ has a supply of the single commodity *in the scenario* w if $w_i > 0$, we say that i has a *demand* of the commodity in scenario w if $w_i < 0$ and that i is a transshipment node in scenario w if $w_i = 0$. It is possible for nodes to a have a supply in one scenario and a demand in another which means that our partitioning of the nodes into origin nodes $\mathscr{N}_w^o$, destination nodes $\mathscr{N}_w^d$ and transshipment nodes $\mathscr{N}_w^t$ now depends on the scenario $w \in \mathscr{U}$. As before, we assume that the supplies and demands are balanced in all scenarios in $\mathscr{U}$ and observe that if $\sum_{i \in \mathscr{N}} w_i \neq 0$ for some $w \in \mathscr{U}$, then no flow can satisfy the supplies and demands simultaneously and the problem instance is infeasible.

Given $\mathscr{G}$, an edge-cost vector $c \in \mathbf{R}_{\geq 0}^{\mathscr{N}}$ and an uncertainty set $\mathscr{U} \subseteq \mathbf{R}^{\mathscr{N}}$, the *Single-Commodity Capacitated Robust Network Design Problem* (SSCCRND) is the task to find an integral minimum-cost capacity vector $u \in \mathbf{Z}^{\mathscr{E}}$ that admits a feasible flow for each $w \in \mathscr{U}$ while minimizing the capacity installation costs $\sum_{\{i,j\} \in \mathscr{E}} c_{ij} u_{ij}$. I.e., it is a two-stage robust optimization problem. The capacity vector u has to be determined before the realization of the demand vector w is known, but the flow can be scenario-specific.

The SSCCRND problem is NP-hard, even if $\mathscr{U}$ is a discrete uncertainty set with three scenarios that only use demands $w_i \in \{-1, 0, 1\}$ and that agree on a common origin node. The deterministic variant of the problem (i.e., $|\mathscr{U}| = 1$) is polynomial time solvable as a minimum-cost flow problem, however, which is in contrast to the deterministic SCFND in Chap. 2, Sect. 2. It is currently unknown if the SSCCRND problem with two scenarios is NP-hard.

4.1 A Flow-Based Formulation

The general problem admits a flow-based formulation of the SSCCRND problem that is similar to the flow-based formulation in Chap. 2, Sect. 2. It has an integer capacity variable u_{ij} for each edge $\{i, j\} \in \mathscr{E}$ and two continuous arc-flow variables f_{ij}^w, f_{ji}^w for each edge $\{i, j\} \in \mathscr{E}$ and each scenario $w \in \mathscr{U}$ (modeling the flow on $\{i, j\}$ in scenario w).

$$\text{Minimize} \quad \sum_{\{i,j\}\in\mathscr{E}} c_{ij} u_{ij} \tag{11.16}$$

$$\text{Subject to} \quad \sum_{\{i,j\}\in\mathscr{E}} \left(f_{ij}^w - f_{ji}^w\right) = w_i \ \forall\, i \in \mathscr{N}, \forall\, w \in \mathscr{U} \tag{11.17}$$

$$f_{ij}^w + f_{ji}^w \le u_{ij} \qquad \forall\, \{i, j\} \in \mathscr{E}, \forall\, w \in \mathscr{U} \tag{11.18}$$

$$f_{ij}^w, f_{ji}^w \ge 0 \qquad \forall\, \{i, j\} \in \mathscr{E}, \forall\, w \in \mathscr{U} \tag{11.19}$$

$$u_{ij} \in \mathbf{Z}_{\ge 0}^E \qquad \forall\, \{i, j\} \in \mathscr{E} \tag{11.20}$$

This formulation is known as the *flow*-based formulation and matches the definition of SSCCRND exactly; any feasible solution defines a feasible flow f^w for all scenarios $w \in \mathscr{U}$ along with minimum integer capacities that support the flows. The constraint matrix of formulation (11.16)–(11.20) is not totally unimodular and thus the integrality requirement for the capacity variables is necessary. Given integer values for u, however, we can always find a feasible f that is integer as well (provided w is integer), even though integrality of the scenario flows is not required in the definition of the SSCCRND problem.

Formulation (11.16)–(11.20) potentially has an infinite number of variables. Assuming $\mathscr{U}$ is a polytope, to make it a finite formulation, we can equivalently replace $\mathscr{U}$ by the set of its vertices. Still, not only may the number of vertices of $\mathscr{U}$ be large, if $\mathscr{U}$ is given in a linear description, it is non-trivial to compute all vertices of $\mathscr{U}$ efficiently.

4.2 A Cut-Set-Based Formulation

We obtain a formulation of finite size by projecting out the flow variables in (11.16)–(11.20). The result is a *cut-set*-based formulation that has an integer capacity variable u_{ij} for each edge $\{i, j\}$. Before we introduce the formulation itself, let us define a robust cut-set-based inequality for the SSCCRND problem as a generalization of the cut-set-based inequalities (2.16) in Chap. 2. Consider any cut $\mathcal{S} \subseteq \mathcal{N}$ and some scenario $w \in \mathcal{U}$. Any feasible choice of capacities u must satisfy $\sum_{\{i,j\}\in(\mathcal{S},\bar{\mathcal{S}})} u_{ij} \geq W_{\mathcal{S}}(w)$, where $W_{\mathcal{S}}(w) := |\sum_{i\in\mathcal{S}} w_i|$ is defined analogously to Chap. 2, Sect. 2.1. This is true for all $w \in \mathcal{U}$ and thus, it is necessary that u satisfies

$$\sum_{\{i,j\}\in(\mathcal{S},\bar{\mathcal{S}})} u_{ij} \geq \max_{w\in\mathcal{U}} W_{\mathcal{S}}(w). \tag{11.21}$$

Inequality (11.21) is a *robust cut-set-based inequality* for the SSCCRND problem.

It turns out that a capacity vector u is feasible if it satisfies the robust cut-set-based inequality (11.21) for all $\mathcal{S} \subseteq \mathcal{N}$. This yields the following cut-set-based formulation for the SSCCRND problem.

$$\text{Minimize} \quad \sum_{\{i,j\}\in\mathcal{E}} c_{ij} u_{ij} \tag{11.22}$$

$$\text{Subject to} \quad \sum_{\{i,j\}\in(\mathcal{S},\bar{\mathcal{S}})} u_{ij} \geq \max_{w\in\mathcal{U}} \Big|\sum_{i\in\mathcal{S}} w_i\Big| \quad \forall\, \mathcal{S} \subseteq \mathcal{N} \tag{11.23}$$

$$u_{ij} \in \mathbf{Z}_{\geq 0} \qquad \forall\, \{i, j\} \in \mathcal{E} \tag{11.24}$$

The cut-set-based formulation is equivalent to the flow-based formulation (11.16)–(11.20) in the sense that the (fractional) solutions of the projection of (11.16)–(11.20) onto the u-space are exactly those defined by (11.23)–(11.24). The linear programming bounds obtained from the relaxations of the flow-based formulation and the cut-set-based formulation are hence the same. A cut-set-based inequality (11.23) for a set $\mathcal{S}$ with $\max_{w\in\mathcal{U}} \sum_{i\in\mathcal{S}} w_i > 0$ defines a facet of the feasible region of (11.23)–(11.24) if both $\mathcal{S}$ and the complement set $\bar{\mathcal{S}}$ induce a connected subgraph of $\mathcal{G}$. The non-negativity constraint (11.24) for u_{ij}, $\{i, j\} \in \mathcal{E}$ is dominated by a cut-set-based inequality if removing $\{i, j\}$ disconnects $\mathcal{G}$ into two partitions with non-zero total balance. In all other cases, the constraint defines a facet of the feasible region of the cut-set-based formulation.

4.3 Separating Robust Cut-Set-Based Inequalities

The size of the cut-set-based formulation is independent of $\mathscr{U}$. However, the number of constraints in the formulation is exponential in the size of $\mathscr{G}$. Whether these constraints can be separated with sufficient efficiency depends on the uncertainty set; for general uncertainty sets, the separation is NP-hard. In this case, applying the generic transformation of the robust counterpart from Sect. 2.5 leads to a linear program with exponentially many constraints and variables. Neither a separation nor a pricing algorithm is known for this program. Alternatively, the problem can be reformulated as a non-convex quadratic program.

There are known tractable special cases, however. The first tractable case occurs when the vertices of $\mathscr{U}$ can be enumerated efficiently (for instance, because $\mathscr{U}$ is a discrete uncertainty set). Then, the separation requires one run of a minimum s-t-cut algorithm per vertex of $\mathscr{U}$. We discuss two other tractable special case in more detail below.

4.3.1 The Single-Commodity Hose Uncertainty Set

Hose uncertainty corresponds to the interval robustness model in the previous section, with the additional constraint that all scenarios must induce balanced supplies and demands. For each node $i \in \mathscr{N}$, we define a lower bound $w_i^{\min} \in \mathbf{Z}$ and an upper bound $w_i^{\max} \in \mathbf{Z}$ on the demand at i and consider any fluctuation of the demands that lies within these bounds and defines a balanced scenario. This results in the following uncertainty set.

$$\mathscr{U}_H(w^{\min}, w^{\max}) := \Big\{ w \in \mathbf{R}^V \;\Big|\; w_i \in [w_i^{\min}, w_i^{\max}] \forall\, i \in \mathscr{N} \;\wedge\; \sum_{i \in \mathscr{N}} w_i = 0 \Big\}. \tag{11.25}$$

The SSCCRND problem with Hose uncertainties remains NP-hard, as does the separation problem for the robust cut-set-based inequalities. However, the separation problem can be reformulated as a mixed integer linear program (MILP) in the following way. The MILP computes a cut $\mathscr{S}$ and the value of the right-hand side of the corresponding cut-set-based inequality. For each node $i \in \mathscr{N}$, we introduce a variable π_i indicating if $i \in \mathscr{S}$ and a binary decision variable ρ_{ij} indicating if $\{i, j\} \in (\mathscr{S}, \bar{\mathscr{S}})$. An additional continuous variable W holds the right-hand side value of the cut-set-based inequality corresponding to $\mathscr{S}$. The MIP minimizes the slack of the cut-set-based inequality induced by $\mathscr{S} = \{i \in \mathscr{N} \mid \pi_i = 1\}$.

$$\text{Minimize} \quad \sum_{\{i,j\} \in E} u_{ij}^* \rho_{ij} - W \tag{11.26}$$

$$\text{Subject to} \qquad W \leq \sum_{i \in \mathscr{N}} \pi_i w_i^{\max} \tag{11.27}$$

$$W \leq -\sum_{i \in \mathscr{N}} (1 - \pi_i) w_i^{\min} \tag{11.28}$$

$$\pi_i - \pi_j \leq \rho_{ij} \qquad \forall \{i, j\} \in \mathscr{E} \tag{11.29}$$

$$\pi_j - \pi_i \leq \rho_{ij} \qquad \forall \{i, j\} \in \mathscr{E} \tag{11.30}$$

$$\pi_i \in \{0, 1\} \qquad \forall\, i \in \mathscr{N} \tag{11.31}$$

$$\rho_{ij} \in \{0, 1\} \qquad \forall \{i, j\} \in \mathscr{E} \tag{11.32}$$

Here, the essential insight is that we can assume without loss of generality that the minimum slack is attained by a cut-set-based inequality for a set $\mathscr{S}$ where $\max_{w \in \mathscr{U}_H} \sum_{i \in \mathscr{S}} w_i$ is non-negative (otherwise, replace $\mathscr{S}$ by the complement $\bar{\mathscr{S}}$). This observation implies that the right-hand side of the inequality simplifies to

$$\max_{w \in \mathscr{U}_H} \Big| \sum_{i \in \mathscr{S}} w_i \Big| = \max \Big\{ \sum_{i \in \mathscr{S}} w_i^{\max}, -\sum_{i \in \bar{\mathscr{S}}} w_i^{\min} \Big\}, \tag{11.33}$$

which is modeled by the constraints (11.27) and (11.28). Computational experiments show that this MILP is solvable for reasonable instance sizes.

4.3.2 Network Containment

The *network containment* uncertainty polytope defines another tractable special case of the SSCCRND problem. In this model, the demand is not defined directly at each node, but through *demand requests*: For each pair of nodes $i, j \in \mathscr{N}$, we define a minimum and a maximum amount $r_{ij}^{\min}, r_{ij}^{\max}$, respectively, of the global single commodity that j can request from i. We then project the demand requests down onto node demands by writing the corresponding uncertainty polytope as

$$\mathscr{U}_C := \Big\{ \big(\sum_{j \in \mathscr{N}} (r_{ij} - r_{ji}) \big)_{i \in \mathscr{N}} \in \mathbf{R}^{\mathscr{N}} \;\Big|\; r_{ij}^{\min} \leq r_{ij} \leq r_{ij}^{\max} \;\forall\, i, j \in \mathscr{N} \Big\} \tag{11.34}$$

Thus, for any choice of demand requests r, we obtain a scenario w where the demand w_i at node i is exactly the total amount $\sum_{j \in \mathscr{N}} r_{ij}$ of the commodity requested *from* i minus the total amount $\sum_{j \in \mathscr{N}} r_{ij}$ of the commodity requested *by* i. In contrast to the Hose model, the scenarios defined in this way are always balanced even without an explicit balancing constraint. The definition also implies that at any node i, the demand request r_{ij} may be satisfied by any node j' (i.e., in general, we may have $j' \neq j$) as long as i receives or sends the correct amount of the commodity. The cut-set separation problem for the network containment uncertainty polytope can be solved by a mixed integer linear program similarly to the Hose polytope.

4.4 Strengthening the Formulations

The feasible regions of the flow-based and of the cut-set-based formulation admit a project-and-lift cut generation procedure that works by contracting edges in an SSCCRND instance. To contract any edge $\{i, j\} \in E$, we merge i and j into a super node i'. All nodes that were previously adjacent to i or j are now adjacent to i' and we delete all resulting parallel edges. In all scenarios $w \in \mathscr{U}$, we set the demand of i' to be $w_i + w_j$. By applying this procedure repeatedly, we can project any SSCCRND instance I into an instance I' with fewer nodes and edges. This smaller instance has nice properties: Any valid inequality for I' can be turned into a valid inequality for I by appropriately lifting the coefficients and moreover, the lifting procedure ensures that facet-defining inequalities for I' remain facet-defining for I.

To find valid inequalities for I', we can repeatedly contract edges until we obtain an instance of constant size. We then apply the *target cut approach* that finds facet-defining inequalities by solving a linear program that has an inequality for each vertex of the feasible region of I'. In the special case where I' is a triangle graph, its three super-nodes define a partitioning of the node set into three disjoint sets $\mathscr{S}_1 \cup \mathscr{S}_2 \cup \mathscr{S}_3 = \mathscr{N}$. This partitioning gives rise to the class of 3-partition inequalities:

$$\sum_{\{i,j\}\in(\mathscr{S}_1:\mathscr{S}_2)} u_{ij} + \sum_{\{i,j\}\in(\mathscr{S}_1:\mathscr{S}_3)} u_{ij} + \sum_{\{i,j\}\in(\mathscr{S}_2:\mathscr{S}_3)} u_{ij} \geq \left\lceil \frac{W_{\mathscr{S}_1} + W_{\mathscr{S}_2} + W_{\mathscr{S}_3}}{2} \right\rceil . \tag{11.35}$$

Here, for any $\mathscr{S}, \mathscr{S}' \in \mathscr{N}$, we define $(\mathscr{S} : \mathscr{S}') := \{\{i, j\} \in \mathscr{E} \mid i \in \mathscr{S} \wedge j \in \mathscr{S}'\}$ to be the set of edges with one endpoint in $\mathscr{S}$ and one endpoint in $\mathscr{S}'$. Further, we let $W_{\mathscr{S}} := \max_{w\in\mathscr{U}} W_{\mathscr{S}}(w)|$ be the right-hand side value of the cut-set-based inequality induced by $\mathscr{S}$. The 3-partition inequalities are facet-defining for the feasible region defined by the cut-set-based formulation (11.22)–(11.24) if $W_{\mathscr{S}_1} + W_{\mathscr{S}_2} + W_{\mathscr{S}_3} > 0$ is odd and each of the sets $\mathscr{S}_1, \mathscr{S}_2, \mathscr{S}_3$ induces a connected subgraph. The 3-partition inequality corresponding to $\mathscr{S}_1, \mathscr{S}_2$ and $\mathscr{S}_3$ can be generated as a $\{0, \frac{1}{2}\}$-Chvátal-Gomory cuts from the cut-set-based inequalities corresponding to $\mathscr{S}_1$, $\mathscr{S}_2$ and $\mathscr{S}_1 \cup \mathscr{S}_2$.

4.5 Variants of the Problem

It is straight-forward to rewrite (11.22)–(11.24) for a setting where opening an edge $\{i, j\}$ incurs a *fixed-cost* of c_{ij}, but provides a fixed capacity u_{ij}. The variant with fixed costs and *uncapacitated edges* can then be seen as the special case where the fixed capacities are set to the sum $W = \sum_{i\in\mathscr{N}} \max_{w\in\mathscr{U}} |w_i|$ of the maximum demands. If $\mathscr{U}$ is described by a system of inequalities, this value can be obtained by solving a linear program for each node $i \in \mathscr{N}$. Transportation costs (in the worst-case) can be added to the objective function of the flow-based formulation (11.16)–

(11.20). Neither variants with fixed costs, transportation costs, nor uncapacitated variants have been considered in the literature so far, to the best of our knowledge.

5 Multicommodity Formulations

If the traffic between different pairs of nodes in our network has to be distinguished, then the multicommodity flow model is an appropriate modeling choice. Analogously to Chap. 2, Sect. 3 we assume that each commodity $k \in \mathcal{K}$ is given as an origin-destination pair $(O(k), D(k)) \in \mathcal{N} \times \mathcal{N}$. To model the uncertain demands, we consider an uncertainty polytope $\mathcal{U} \subseteq \mathbf{R}^{\mathcal{K}}$. Any scenario $d \in \mathcal{U}$ specifies a demand d^k for commodity $k \in \mathcal{K}$. As before, we can re-write the demands as flow balances by setting

$$w_i^{k,d} := \begin{cases} d^k, \text{ if } i = O(k) \\ -d^k, \text{ if } i = D(k) \\ 0, \text{ otherwise} \end{cases} \qquad \forall\, d \in \mathcal{U}, \forall\, k \in \mathcal{K}, \forall\, i \in \mathcal{N}. \tag{11.36}$$

Here, however, the uncertainty polytope introduces a dependence of w on the scenario d. Given an undirected graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ with commodities $(O(k), D(k))$ for $k \in \mathcal{K}$, a scenario polytope $\mathcal{U} \subseteq \mathbf{R}^{\mathcal{K}}$, and an installation cost c_{ij} for each edge $e \in \mathcal{E}$, the MSCCRND problem is to find an integer capacity u_{ij} for each edge $\{i, j\}$ such that $\sum_{\{i,j\}\in\mathcal{E}} c_{ij}u_{ij}$ is minimum and all demands in $\mathcal{U}$ can be routed. In the dynamic routing case, the demands must be routed with a multicommodity flow. In the case of static routing, the demands are routed with a routing template as briefly described in Sect. 3.

5.1 *Standard Uncertainty Sets*

Consider the application of the budget uncertainty approach to multicommodity network design (cf. Sect. 2.4). We denote by $\bar{d}^k$ a nominal value for the demand of each commodity $k \in \mathcal{K}$ and we suppose that the true demand of the commodity can deviate from its nominal value by at most $\hat{d}^k$. Additionally, there can be at most $\Gamma \in \mathbf{Z}_{\geq 0}$ deviations at the same time. We define the resulting uncertainty set as the Γ-robustness polytope for the MSCCRND.

$$\mathcal{U}_B(\bar{d},\hat{d},\Gamma) := \operatorname{conv}\left\{ d \in \mathbf{R}^K_{\geq 0} \;\middle|\; \begin{array}{ll} d^k \in [\bar{d}^k - \hat{d}^k, \bar{d}^k + \hat{d}^k], & \text{if } k \in S \\ d^k = \bar{d}^k, & \text{otherwise} \\ \forall\, S \subseteq \mathcal{K} \text{ with } |S| \leq \Gamma & \end{array} \right\}. \tag{11.37}$$

Defining $S(\Gamma) := \{\sigma \in [0,1]^K \mid \sum_{k\in\mathcal{K}} \sigma^k \leq \Gamma\}$ as the set of possible deviations, we can rewrite the Γ-robustness polytope equivalently as the following set.

$$\mathcal{U}_B(\bar{d},\hat{d},\Gamma) = \bar{d} + \left\{ (\sigma^k \hat{d}^k)_{k\in\mathcal{K}} \in \mathbf{R}^K_{\geq 0} \;\middle|\; \sigma \in S(\Gamma) \right\}. \tag{11.38}$$

An alternative to the budget uncertainty set was proposed by Fingerhut et al. (1997) and Duffield et al. (1999) independently. In the *Hose* model, we only assume that we know the maximum *incoming* traffic d_i^{in} and the maximum *outgoing* traffic d_i^{out} at each node $i \in V$. We then define a commodity (s,t) with demand d^{st} for all pairs of nodes $s, t \in \mathcal{N}$ and consider any demand vector d that adheres to the traffic bounds. We call the resulting uncertainty set

$$\mathcal{U}_H(d^{\text{in}}, d^{\text{out}}) := \left\{ d \in \mathbf{R}^{\mathcal{N}\times\mathcal{N}}_{\geq 0} \;\middle|\; \sum_{t\in\mathcal{N}} d^{it} \leq d_i^{\text{out}} \wedge \sum_{s\in\mathcal{N}} d^{si} \leq d_i^{\text{in}} \;\forall\, i \in \mathcal{N} \right\} \tag{11.39}$$

the (multicommodity) Hose polytope. We speak of the symmetric Hose polytope if $d_i^{\text{in}} = d_i^{\text{out}}$ for all nodes $i \in \mathcal{N}$.

For this uncertainty model, we only need to estimate $\Theta(|\mathcal{N}|)$ parameters (as opposed to a worst case of $\Theta(|\mathcal{N}|^2)$ in the budget uncertainty case). Additionally, these parameters are easier to predict and can even be known exactly if they stem from technical specifications or legal contracts.

5.2 The VPN Problem

The MSCCRND problem with Hose uncertainties and static routing is known as the *Virtual Private Network* (VPN) design problem. The problem is NP-hard in general, but can be solved efficiently if the Hose polytope is symmetric in the above sense and if we force the flow to be unsplittable. In that case, there is always an optimum routing template that forms a tree and optimum *tree routing templates* for unsplittable flows can be found in polynomial time. We will see in the following section how the general VPN problem can be solved with ILP methods.

5.3 Static Routing: Arc-Flow Based Formulations

In the static routing case, the flow formulation does not need a set of arc-flow variables for every scenario: As we use the same routing template in all scenarios, a single set is sufficient. Here, for all commodities $k \in \mathscr{K}$ and all edges $\{i, j\} \in \mathscr{E}$, the routing template variables f_{ij}^k and f_{ji}^k denote the fraction of the demand of the commodity k that is routed via the arcs (i, j) and (j, i), respectively, in each scenario $d \in \mathscr{U}$. As before, we use an integer variable u_{ij} to model the capacity of the edge $\{i, j\}$, for all $\{i, j\} \in \mathscr{E}$.

$$\text{Minimize} \quad \sum_{\{i,j\}\in E} c_{ij} u_{ij} \tag{11.40}$$

$$\text{Subject to} \quad \sum_{\{i,j\}\in E} f_{ij}^k - f_{ji}^k = \begin{cases} 1, & \text{if } i = O(k) \\ -1, & \text{if } i = D(k) \\ 0, & \text{otherwise} \end{cases} \quad \begin{array}{l} \forall\, k \in \mathscr{K} \\ \forall\, i \in \mathscr{N} \end{array} \tag{11.41}$$

$$\sum_{k\in\mathscr{K}} d^k \cdot \left(f_{ij}^k + f_{ji}^k\right) \le u_{ij} \quad \begin{array}{l} \forall\, \{i, j\} \in \mathscr{E} \\ \forall\, d \in \mathscr{U} \end{array} \tag{11.42}$$

$$f_{ij}^k, f_{ji}^k \in [0, 1] \quad \begin{array}{l} \forall\, \{i, j\} \in \mathscr{E} \\ \forall\, k \in \mathscr{K} \end{array} \tag{11.43}$$

$$u_{ij} \in \mathbf{Z}_{\ge 0} \quad \forall\, \{i, j\} \in \mathscr{E} \tag{11.44}$$

Again, it is sufficient to include the constraints (11.42) for the vertices of $\mathscr{U}$. Forcing that f is binary leads to a variant where the template for each commodity uses a unique path and thus, a routing template for an unsplittable flow.

Let us now robustify the arc-flow formulation (11.40)–(11.44) with the Γ-robustness model.

$$\text{Minimize} \quad \sum_{\{i,j\}\in E} c_{ij} u_{ij} \tag{11.45}$$

Subject to

$$\sum_{\{i,j\}\in\mathscr{E}} f_{ij}^k - f_{ji}^k = \begin{cases} 1, & \text{if } i = O(k) \\ -1, & \text{if } i = D(k) \\ 0, & \text{otherwise} \end{cases} \quad \begin{array}{l} \forall\, k \in \mathscr{K} \\ \forall\, i \in V \end{array} \tag{11.46}$$

$$\sum_{k\in\mathscr{K}} \bar{d}^k (f_{ij}^k + f_{ji}^k) + \max_{\sigma\in\mathscr{U}_B(\Gamma)} \sum_{k\in\mathscr{K}} \sigma_k \hat{d}^k \left(f_{ij}^k + f_{ji}^k\right) \le u_{ij} \quad \forall\, \{i, j\} \in \mathscr{E} \tag{11.47}$$

$$f_{ij}^k, f_{ji}^k \in [0, 1] \qquad \forall \{i, j\} \in \mathscr{E} \quad \forall k \in \mathscr{K} \tag{11.48}$$

$$u_{ij} \in \mathbf{Z}_{\geq 0} \qquad \forall \{i, j\} \in \mathscr{E} \tag{11.49}$$

The constraint (11.47) contains an optimization problem in itself so that the formulation is a two level program. We would like to collapse the program into a single level. Unfortunately, in the inner level inside of the constraint (11.47), we seek to *maximize* $\sum_{k\in\mathscr{K}} \sigma_k \hat{d}^k(f_{ij}^k + f_{ji}^k)$ over $\mathscr{U}_B(\Gamma)$ while the outer level (11.45)–(11.48) strives to minimize this value. If we could rewrite the maximization as an equivalent minimization problem, we could collapse the two levels as desired. Given fixed template flows f^k, this can be achieved by modeling $\max_{\sigma in \mathscr{U}_B(\Gamma)} \sum_{k\in\mathscr{K}} \sigma_k \hat{d}^k\big(f_{ij}^k + f_{ji}^k\big)$ as a linear program and replacing it by its dual

$$\text{Minimize} \quad \gamma_{ij} \cdot \Gamma + \sum_{k\in\mathscr{K}} \tau_{ij}^k \tag{11.50}$$

$$\text{Subject to} \quad \gamma_{ij} + \tau_{ij}^k \geq \hat{d}^k(f_{ij}^k + f_{ji}^k) \ \forall k \in \mathscr{K} \tag{11.51}$$

$$\tau_{ij}^k \geq 0 \qquad \forall k \in \mathscr{K} \tag{11.52}$$

$$\gamma_{ij} \geq 0 \tag{11.53}$$

for all edges $\{i, j\} \in \mathscr{E}$. The result is a compact mixed-integer linear program with $\Theta(|\mathscr{N}|^2|\mathscr{E}|)$ variables and $\Theta(|\mathscr{N}|^3 + |\mathscr{N}|^2|\mathscr{E}|)$ constraints.

$$\text{Minimize} \sum_{\{i,j\}\in\mathscr{E}} c_{ij} u_{ij} \tag{11.54}$$

$$\text{Subject to} \quad \sum_{\{i,j\}\in\mathscr{E}} f_{ij}^k - f_{ji}^k = \begin{cases} 1, & \text{if } i = O(k) \\ -1, & \text{if } i = D(k) \\ 0, & \text{otherwise} \end{cases} \qquad \forall k \in \mathscr{K} \quad \forall i \in \mathscr{N} \tag{11.55}$$

$$\sum_{k\in\mathscr{K}} \bar{d}^k(f_{ij}^k + f_{ji}^k) + \gamma_{ij} \cdot \Gamma + \sum_{k\in\mathscr{K}} \tau_{ij}^k \leq u_{ij} \ \forall \{i, j\} \in \mathscr{E} \tag{11.56}$$

$$\gamma_{ij} + \tau_{ij}^k - \hat{d}^k(f_{ij}^k + f_{ji}^k) \geq 0 \qquad \forall k \in \mathscr{K} \quad \forall \{i, j\} \in \mathscr{E} \tag{11.57}$$

$$\tau_{ij}^k \geq 0 \qquad \forall k \in \mathscr{K} \quad \forall \{i, j\} \in \mathscr{E} \tag{11.58}$$

$$\gamma_{ij} \geq 0 \qquad \forall \{i, j\} \in \mathscr{E} \tag{11.59}$$

$$f_{ij}^k, f_{ji}^k \in [0,1] \qquad \begin{array}{l} \forall\, k \in \mathscr{K} \\ \forall\, \{i,j\} \in \mathscr{E} \end{array} \tag{11.60}$$

$$u_{ij} \in \mathbf{Z}_{\geq 0} \qquad \forall\, \{i,j\} \in \mathscr{E} \tag{11.61}$$

As the objective function makes sure that the left-hand side of constraint (11.56) is minimized, we can omit the explicit minimization without inserting complementary slackness conditions.

Alternatively, we can solve the separation problem for the constraints (11.47). Given fixed f and u, the problem amounts to finding a deviation $\sigma \in \mathscr{U}_B(\Gamma)$ and an edge $\{i,j\} \in E$ such that

$$\sum_{k \in \mathscr{K}} \bar{d}^k(f_{ij}^k + f_{ji}^k) + \sum_{k \in \mathscr{K}} \sigma^k \hat{d}^k\big(f_{ij}^k + f_{ji}^k\big) > u_{ij} \tag{11.62}$$

or to decide that none such combination of a deviation and an edge exists. The problem can be solved separately for each fixed edge $\{i,j\} \in \mathscr{E}$. Then, it amounts to solving

$$\max_{\sigma \in \mathscr{U}_B(\Gamma)} \sum_{k \in \mathscr{K}} \sigma^k \hat{d}^k\big(f_{ij}^k + f_{ji}^k\big). \tag{11.63}$$

If the optimum value of (11.63) is larger than $u_{ij} - \sum_{k \in \mathscr{K}} \bar{d}^k(f_{ij}^k + f_{ji}^k)$, then we found a violated inequality; otherwise, no violated inequality involving the edge $\{i,j\}$ exists. To solve (11.63), we can sort the values $\hat{d}^k(f_{ij}^k + f_{ji}^k)$ for $k \in \mathscr{K}$ in non-increasing order. Then, the first Γ commodities determine a worst-case deviation. This approach yields a program that initially has $\Theta(|\mathscr{K}||\mathscr{N}|)$ constraints and $\Theta(|\mathscr{K}||\mathscr{E}|)$ variables. To solve the linear programming relaxation of the problem, we need to solve $\Theta(|\mathscr{E}|)$ separation problems per iteration of the separation algorithm.

To combine the arc-flow formulation with the Hose polytope, we can equivalently replace constraint (11.42) by the optimization

$$u_{ij} = \max \sum_{s,t \in V} d^{st}(f_{ij}^{st} + f_{ji}^{st}) \tag{11.64}$$

$$\text{Subject to} \quad \sum_{t \in V} d^{st} \leq d_s^{\text{out}} \qquad \forall\, s \in V \tag{11.65}$$

$$\sum_{s \in V} d^{st} \leq d_t^{\text{in}} \qquad \forall\, t \in V \tag{11.66}$$

$$d^{st} \geq 0 \qquad \forall\, s,t \in V \tag{11.67}$$

for all $\{i,j\} \in \mathscr{E}$. For fixed f, this gives us a bounded, feasible linear program for each edge $\{i,j\} \in E$. Again, we now replace these programs by their dual. In the linear program for edge $\{i,j\}$, denote by ω_s^{ij} and υ_t^{ij} the dual variables corresponding to the constraints (11.65) and (11.66), respectively. This yields the following dual for each edge $\{i, j\} \in \mathscr{E}$.

$$\text{Minimize} \quad \sum_{s \in \mathscr{N}} d_s^{\text{out}} \omega_s^{ij} + \sum_{t \in \mathscr{N}} d_t^{\text{in}} \upsilon_t^{ij} \tag{11.68}$$

$$\text{Subject to } \omega_s^{ij} + \upsilon_t^{ij} \geq f_{ij}^{st} + f_{ji}^{st} \quad \forall\, s, t \in V \tag{11.69}$$

$$\omega_s^{ij} \geq 0 \qquad \forall\, s \in V \tag{11.70}$$

$$\upsilon_t^{ij} \geq 0 \qquad \forall\, t \in V \tag{11.71}$$

This program is linear, even for a non-fixed f. We insert it into formulation (11.40)–(11.44), replacing constraint (11.41).

$$\text{Minimize} \quad \sum_{\{i,j\} \in E} c_{ij} u_{ij} \tag{11.72}$$

$$\text{Subject to} \quad \sum_{\{i,j\} \in E} f_{ij}^{st} - f_{ji}^{st} = \begin{cases} 1, & \text{if } i = s \\ -1, & \text{if } i = t \\ 0, & \text{otherwise} \end{cases} \quad \begin{array}{l} \forall\, s, t \in \mathscr{N} \\ \forall\, i \in \mathscr{N} \end{array} \tag{11.73}$$

$$\sum_{s \in \mathscr{N}} d_s^{\text{out}} \omega_s^{ij} + \sum_{t \in \mathscr{N}} d_s^{\text{in}} \upsilon_s^{ij} \leq u_{ij} \qquad \forall\, \{i, j\} \in \mathscr{E} \tag{11.74}$$

$$\omega_s^{ij} + \upsilon_t^{ij} \geq f_{ij}^{st} + f_{ji}^{st} \qquad \begin{array}{l} \forall \{i, j\} \in \mathscr{E} \\ \forall\, s, t \in \mathscr{N} \end{array} \tag{11.75}$$

$$\omega_s^{ij}, \upsilon_s^{ij} \geq 0 \qquad \begin{array}{l} \forall\, \{i, j\} \in \mathscr{E} \\ \forall\, s \in \mathscr{N} \end{array} \tag{11.76}$$

$$f_{ij}^{st}, f_{ji}^{st} \in [0, 1] \qquad \begin{array}{l} \forall\, \{i, j\} \in \mathscr{E} \\ \forall\, s, t \in \mathscr{N} \end{array} \tag{11.77}$$

$$u_{ij} \in \mathbf{Z}_{\geq 0} \qquad \forall\, \{i, j\} \in \mathscr{E} \tag{11.78}$$

In this way, we directly obtain a single level mixed integer linear program and do not need any further linearization. Solving the program gives us minimum cost integer capacities for MSCCRND with static routing over the Hose polytope. The program has $\Theta(|\mathscr{N}|^2|\mathscr{E}|)$ variables and $\Theta(|\mathscr{N}|^3 + |\mathscr{N}|^2|\mathscr{E}|)$ constraints. A similar approach for a problem variant with multiple facilities was given by Altin et al. (2011). Requiring that f is integral yields an MIP formulation for the VPN problem.

5.4 Static Routing: Path Based Formulations

In the static routing case, the MSCCRND problem with an arbitrary uncertainty set $\mathscr{U}$ can be cast into a path-formulation. In this formulation, we additionally assume that there is an upper bound $\bar{u}_{ij}$ on the capacity u_{ij} of edge $\{i, j\} \in \mathscr{E}$ so that the feasible region is bounded. If these bounds are not desired, they can be replaced by a sufficiently large number (for instance, we can set the upper bound to $\sum_{i \in \mathscr{N}} \sum_{k \in \mathscr{K}} \max_{d \in \mathscr{U}} d_i^k$ for all edges). For ease of notation, we let $\mathscr{P}_k$ be the set of all paths between the origin $O(k)$ of the k-th commodity and its destination $D(k)$, for all $k \in \mathscr{K}$. As before, let $\mathscr{P} = \cup_{k \in \mathscr{K}} \mathscr{P}_k$. We use a continuous path variable x_p for each $p \in \mathscr{P}$.

$$\text{Minimize} \sum_{\{i,j\} \in \mathscr{E}} c_{ij} u_{ij} \tag{11.79}$$

$$\text{Subject to} \quad \sum_{p \in \mathscr{P}_k} x_p = 1 \qquad \forall\, k \in \mathscr{K} \tag{11.80}$$

$$\sum_{k \in \mathscr{K}} \sum_{\substack{p \in \mathscr{P}_k: \\ \{i,j\} \in p}} d^k x_p \leq u_{ij} \qquad \begin{matrix} \forall\, \{i, j\} \in \mathscr{E} \\ \forall\, d \in \mathscr{U} \end{matrix} \tag{11.81}$$

$$x_p \in [0, 1] \qquad \forall\, p \in \mathscr{P} \tag{11.82}$$

$$u_{ij} \in \{0, \dots, \bar{u}_{ij}\} \qquad \forall\, \{i, j\} \in \mathscr{E} \tag{11.83}$$

To model unsplittable routing it suffices to turn the path-flow variables x in program (11.79)–(11.83) into binary variables.

We now decompose the path formulation (11.79)–(11.83) into a master and two satellite problems. The *master* problem consists of a variant of the path formulation. It maintains a set $\bar{\mathscr{P}} \subseteq \mathscr{P}$ of relevant paths as well as a set of relevant scenarios $\bar{\mathscr{U}} \subseteq \mathscr{U}$. Both sets are initially empty.

$$\text{Minimize} \sum_{\{i,j\} \in E} c_{ij} u_{ij} \tag{11.84}$$

$$\text{Subject to} \quad \sum_{p \in c\bar{P}_k} x_p \geq 1 \quad \forall\, k \in \mathscr{K} \tag{11.85}$$

$$\sum_{\substack{p \in \mathscr{P}_k: \\ \{i,j\} \in p}} x_p \leq f_{ij}^k \quad \forall\, k \in \mathscr{K}, \quad \forall\, \{i, j\} \in \mathscr{E} \tag{11.86}$$

$$\sum_{k \in \mathscr{K}} f_{ij}^k d^k \leq u_{ij} \ \forall\, \{i, j\} \in \mathscr{E} \ \forall\, (d_{st})_{s,t \in V} \in \bar{\mathscr{U}} \tag{11.87}$$

$$u_{ij} \leq \bar{u}_{ij} \qquad \forall \{i,j\} \in \mathscr{E} \tag{11.88}$$

$$x_p \in [0,1] \qquad \forall p \in \bar{\mathscr{P}} \tag{11.89}$$

$$f_{ij}^k \in [0,1] \qquad \forall k \in \mathscr{K}, \ \forall \{i,j\} \in \mathscr{E} \tag{11.90}$$

The master problem (11.84)–(11.90) is bounded. Suppose for the moment that it is feasible as well and let u^* be an optimum solution for the problem. In order to guarantee that u^* is globally optimum, we need to make sure that adding additional paths to $\bar{\mathscr{P}}$ cannot improve the value of u^*. Moreover, u^* must be globally feasible, i.e., the capacities must be sufficient to route all scenarios in $\mathscr{U}$ (and not only those in $\bar{\mathscr{U}}$). For the former problem, we solve a *path satellite problem*.

It consists of computing a shortest path between all origin-destination pairs with respect to the dual variables π and ρ of the constraints (11.85) and (11.86). Indeed, one can argue that if $\sum_{\{i,j\}\in p} \rho_{ij}^k < \pi^k$ for some $O(k)$-$D(k)$-path $p \notin \bar{\mathscr{P}}$, then p will improve the current solution u^*. In this case, we add the path p to $\bar{\mathscr{P}}$. To ensure global feasibility on the other hand, we separate inequalities of type (11.87) in a *demand satellite* problem. Given fixed routing variables f and fixed capacities u from an optimum solution of (11.84)–(11.90), it suffices to solve the linear program

$$u_{ij}^{\max} := \max \quad \sum_{k\in\mathscr{K}} f_{ij}^k d^k \tag{11.91}$$

$$\text{Subject to} \qquad d \in \mathscr{U} \tag{11.92}$$

for all edges $\{i,j\} \in E$. Notice that here, we optimize over the entire scenario set. If for some edge $\{i,j\} \in E$ we find that $u_{ij}^{\max} > u_{ij}$, then the inequality

$$\sum_{k\in\mathscr{K}} f_{ij}^k d^k \leq u_{ij} \tag{11.93}$$

is violated by (f,u). We add the corresponding optimum solution of (11.91)–(11.92) to $\bar{\mathscr{U}}$, thus adding the violated inequality (11.93) to the master problem.

To solve the master problem to global optimality, it now suffices to iteratively call the path satellite, the demand satellite and the master problem itself until neither new paths nor new scenarios are found. If at some point during the computation the master problem becomes infeasible, we call the path satellite and if no improving paths can be found, then the problem instance must be globally infeasible (i.e., the upper bounds for the capacities are too restrictive to route all scenarios in $\mathscr{U}$).

5.5 Dynamic Routing: Arc-Flow Based Formulations

The robustification of the capacitated multicommodity network design problem works analogously to the SSCCRND case. For the arc-flow formulation, we introduce one set of arc-flow variables for each scenario $d \in \mathscr{U}$ and each commodity $k \in \{1, \ldots, K\}$. This gives us a robustified version of the classical arc-flow formulation.

$$\text{Minimize} \quad \sum_{\{i,j\} \in E} c_{ij} u_{ij} \tag{11.94}$$

$$\text{Subject to} \quad \sum_{\{i,j\} \in E} f_{ij}^{k,d} - f_{ji}^{k,d} = w_i^{k,d}, \;\; \forall\, i \in V, \forall\, d \in \mathscr{U}, \quad \forall\, k \in \mathscr{K} \tag{11.95}$$

$$\sum_{k \in \mathscr{K}} f_{ij}^{k,d} + f_{ji}^{k,d} \le u_{ij}, \quad \forall\, \{i,j\} \in \mathscr{E}, \quad \forall\, d \in \mathscr{U}, \tag{11.96}$$

$$f_{ij}^{k,d},\; f_{ji}^{k,d} \ge 0, \qquad \forall\, k \in \mathscr{K}, \;\; \forall\, d \in \mathscr{U}, \forall\, \{i,j\} \in \mathscr{E} \tag{11.97}$$

$$u_{ij} \in \mathbf{Z}_{\ge 0}^{E}, \qquad \forall\, \{i,j\} \in \mathscr{E} \tag{11.98}$$

Featuring $2|\mathscr{K}||\mathscr{E}|$ flow-variables for each scenario $d \in \mathscr{U}$ and $\Theta(|\mathscr{E}|)$ constraints (11.96) for all $d \in \mathscr{U}$, this formulation is of infinite size. Again, if $\mathscr{U}$ is a polytope, we can replace $\mathscr{U}$ by the set of its vertices to obtain a finite (although potentially inpractical) formulation.

5.6 Dynamic Routing: Formulations Without Flow Variables

As also can be observed for deterministic multi-commodity network design problems, Gale's cut condition is not sufficient for the existence of a multi-commodity flow and thus, these problems cannot be cast into a cut-set based formulation in general. There is, however, a generalization of Gale's condition, called the *Japanese Theorem*, by Onaga and Kakusho (1971) that enables us to formulate the problem with capacity variables only. Let $\mathscr{M} \subseteq \mathbf{R}_{\ge 0}^{\mathscr{N} \times \mathscr{N}}$ be the metric cone, i.e., set of all real metrics on $\mathscr{N}$. Given capacities $u \in \mathbf{R}_{\ge 0}^{\mathscr{E}}$, a feasible multicommodity flow exists if and only if

$$\sum_{\{i,j\} \in \mathscr{E}} \mu_{ij} u_{ij} \ge \sum_{k \in \mathscr{K}} d^k \cdot \operatorname{dist}_\mu(O(k), D(k)) \quad \forall \text{ metrics } \mu \in \mathscr{M}, \tag{11.99}$$

where $\operatorname{dist}_\mu(s,t)$ denotes the shortest path distance from $s \in \mathscr{N}$ to $t \in \mathscr{N}$ with respect to μ. To see why the condition is necessary, observe the following: Let $\mu \in \mathscr{M}$ be any metric and let us interpret μ as edge weights. Then, the network has a total weighted capacity of $U := \sum_{\{i,j\} \in E} \mu_{ij} u_{ij}$. Sending one unit of flow from $O(k)$ to $D(k)$ along a path $p \in \mathscr{P}_k$ "consumes" a weighted capacity of $\sum_{\{i,j\} \in p} \mu_{ij} u_{ij}$ and there exists a feasible multicommodity flow if all demands can be sent while consuming at most the total weighted capacity U. But how much weighted capacity do we need to consume at least in order to send all the demands? The best we can do is to send d^k units along a shortest $O(k)$-$D(k)$-path for all $k \in \mathscr{K}$ and this consumes a weighted capacity of exactly $\sum_{k \in \mathscr{K}} d^k \operatorname{dist}_\mu(O(k), D(k))$. Thus, if $\sum_{\{i,j\} \in E} \mu_{ij} u_{ij} < \sum_{k \in \mathscr{K}} d^k \operatorname{dist}_\mu(O(k), D(k))$ for any metric μ, then no feasible multicommodity flow can exist under our choice of u. A rigorous proof of both the sufficiency and the necessity of the condition follows from applying Farkas' Lemma to the path formulation (2.32)–(2.36) in Chap. 2, Sect. 3.

The Japanese Theorem directly leads to the following capacity formulation for the MSCCRND problem with dynamic routing.

$$\text{Minimize} \sum_{\{i,j\} \in \mathscr{E}} c_{ij} u_{ij} \tag{11.100}$$

Subject to

$$\sum_{\{i,j\} \in \mathscr{E}} \mu_{ij} u_{ij} \geq \max_{d \in \mathscr{U}} \sum_{k \in \mathscr{K}} d^k \cdot \operatorname{dist}_\mu(O(k), D(k)) \ \forall\, \mu \in \mathscr{M} \tag{11.101}$$

$$u_{ij} \in \mathbf{Z}_{\geq 0} \qquad \forall\, \{i, j\} \in E \tag{11.102}$$

The inequalities (11.101) are called *metric inequalities* and while there is an infinite number of metrics $\mu \in \mathscr{M}$, it is sufficient to include metric inequalities only for the (finitely many) extreme rays of the metric cone $\mathscr{M}$. The result is a finite integer linear program, but to make it practically viable, we need a separation algorithm. The only known separation algorithm for the metric inequalities so far is to write the separation problem as a bi-level (continuous) linear program and to then apply a standard transformation to turn it into a single-level quadratic program with complementary slackness conditions. This program can then be linearized; however, the linearization requires additional integer variables and big-M constraints.

For static routing and budget uncertainty set $\mathscr{U}_B(\bar{d}, \hat{d}, \Gamma)$, a straight-forward generalization of the metric inequalities is not sufficient. In Claßen et al. (2015) the correct right hand side of (11.101) for this case is derived alongside with a polynomial time separation algorithm.

5.7 Strengthening the Formulations

The correctness of all the above formulations does not imply that the linear relaxation is close to an optimal integer solution. To improve the performance of branch-and-bound based solvers, the formulations can be strengthened with (facet-defining) valid inequalities. For this, the metric inequalities (11.101) are of particular interest. These inequalities are valid for the earlier formulations and connect capacities of different edges. Let us define the cut-metric

$$\mu_{ij} := \begin{cases} 1, & \text{if } i \in \mathscr{S} \text{ and } j \in \bar{\mathscr{S}} \\ 0, & \text{otherwise} \end{cases} \tag{11.103}$$

for some cut-set $\mathscr{S} \subseteq V$, i.e., the edges between $\mathscr{S}$ and $\bar{\mathscr{S}}$ have length 1 whereas all other edges have length 0. Clearly, the cut-metric is a metric, and hence the robust cut-set inequality

$$\sum_{\{i,j\}\in(\mathscr{S},\bar{\mathscr{S}})} u_{ij} \geq \left\lceil \max_{d\in\mathscr{U}} \left(\sum_{k\in\mathscr{K}:O(k)\in\mathscr{S},D(k)\in\bar{\mathscr{S}}} d^k + \sum_{k\in\mathscr{K}:O(k)\in\bar{\mathscr{S}},D(k)\in\mathscr{S}} d^k \right) \right\rceil \tag{11.104}$$

is a valid inequality (since the left hand side is integer-valued, the right hand side can be rounded up to the next integer). For budget uncertainty, the right hand side can be computed by selecting the Γ largest deviations among those commodities crossing the cut (in addition to the nominal values). In fact, in this case the robust cut-set inequality (11.104) define a facet if both $\mathscr{S}$ and $\bar{\mathscr{S}}$ induce connected subgraphs and actual rounding is performed.

Further valid inequalities can be derived by considering k-partitions of the node set (cf. Sect. 4.4) or considering a single edge capacity constraint (11.47) (generalizing the so-called arc-residual capacity constraints, cf. Kutschka (2013)).

6 Bibliographical Notes

Robust optimization is an emerging field of research. Probably the earliest work in the field was reported by Soyster (1973). In the context of discrete optimization, several contribution were made by Kouvelis and Yu (1997). The budget uncertainty set was introduced by Bertsimas and Sim (2003, 2004) and is probably the most successful approach to date. A generalization of the budget uncertainty model to be used in the context of network design has been introduced by Büsing and D'Andreagiovanni (2012). We refer to Ben-Tal et al. (2009) for a robust optimization in continuous optimization. Multi-stage robustness was proposed by Ben-Tal et al. (2004). See Liebchen et al. (2009) for an introduction to recoverable robustness.

The first works for the single-commodity case of robust network design discuss the case where the uncertainty set is a finite list of scenarios. This case was first studied by Minoux (1989) and by Sanità (2009). Buchheim et al. (2011) propose the arc-flow-based formulation. Target cuts are due to Buchheim et al. (2008). Álvarez-Miranda et al. (2012) introduce the cut-set-based formulation with a separation algorithm for discrete scenario sets. The separation for cut-set inequalities under Hose uncertainty is due to Cacchiani et al. (2016). The 3-partition inequalities and their facet-defining properties were studied by Magnanti et al. (1993), Agarwal (2006), Cacchiani et al. (2016), and Schmidt (2014). Pesenti et al. (2004) study the network containment problem.

In the context of multicommodity network design, robust optimization was merely applied to communication networks. Belotti et al. (2008) consider network engineering. Altin et al. (2011) study network design with under the Hose uncertainty model and we refer to Koster et al. (2013) for network design under budget uncertainty.

The arc-flow based formulation for the MSCCRND problem with static routing and budget uncertainty is due to Koster et al. (2013); this includes the reformulation as a linear program and the separation algorithm. Altin et al. (2007) robustify the arc-flow formulation with Hose uncertainties. The path-flow formulation together with the solution algorithm was proposed by Ben-Ameur and Kerivin (2005).

The separation of robust metric inequalities for dynamic routing is due to Mattia (2013). Claßen et al. (2015) derive robust metric inequalities for static routing; the derivation of robust cutset inequalities as Chvátal-Gomory cuts is due to Koster et al. (2013).

The polynomial time algorithm for optimum tree routing templates for the VPN problem with unsplittable flows was given by Gupta et al. (2001). Goyal et al. (2008) prove that the VPN problem with unsplittable flows always has an optimum tree routing template.

For multi-stage robustness and affine recourse models in network design we refer to Atamtürk and Zhang (2007), Ben-Ameur (2007), and Poss and Raack (2013).

Instances of robust network design can be found at SNDlib Orlowski et al. (2010).

7 Conclusions and Perspectives

In recent years, network design has been on the one hand a fruitful application area for the emerging field of robust optimization. But, on the other hand, the robust network design has also been stimulating the further development of the robust optimization methodology. It can be expected that both these developments will continue in the years to come. In particular, multi-stage robustness concepts are still in their development and different applications will require new angles of view to arrive at suitable solutions. The increasing complexity by robustness concepts in general, and multi-stage concepts in particular, will force the development of new algorithmic ideas to deal with them.

References

Agarwal, Y. K. (2006). k-Partition-based facets of the network design problem. *Networks, 47*(3), 123–139.

Altin, A., Amaldi, E., Belotti, P., & Pinar, M. Ç. (2007). Provisioning virtual private networks under traffic uncertainty. *Networks, 49*(1), 100–155.

Altin, A., Yaman, H., & Pinar, M. (2011). The robust network loading problem under hose demand uncertainty: formulation, polyhedral analysis, and computations. *INFORMS Journal on Computing, 23*(1), 75–89.

Álvarez-Miranda, E., Cacchiani, V., Dorneth, T., Jünger, M., Liers, F., Lodi, A., et al. (2012). Models and algorithms for robust network design with several traffic scenarios. In A. Ridha Mahjoub, V. Markakis, I. Milis, & V. T. Paschos (Eds.), ISCO 2012, Revised Selected Papers. *Lecture notes in computer science* (vol. 7422, pp. 261–272). Springer.

Atamtürk, A., & Zhang, M. (2007). Two-stage robust network flow and design under demand uncertainty. *Operations Research, 55*(4), 662–673.

Belotti, P., Capone, A., Carello, G., & Malucelli, F. (2008). Multi-layer mpls network design: The impact of statistical multiplexing. *Computer Networks, 52*(6), 1291–1307.

Ben-Ameur, W. (2007). Between fully dynamic routing and robust stable routing. In *6th International Workshop on Design of Reliable Communication Networks, 2007, DRCN 2007* (pp. 1–6). IEEE

Ben-Ameur, W., & Kerivin, H. (2005). Routing of uncertain demands. *Optimization and Engineering, 3*, 283–313.

Ben-Tal, A., & Nemirovski, A. (1999). Robust solutions of uncertain linear programs. *Operations Research Letters, 25*(1), 1–13.

Ben-Tal, A., Goryashko, A., Guslitzer, E., & Nemirovski, A. (2004). Adjustable robust solutions of uncertain linear programs. *Mathematical Programming, 99*(2), 351–376.

Ben-Tal, A., Ghaoui, L. E., & Nemirovski, A. (2009). *Robust optimization*. Princeton: Princeton University Press

Bertsimas, D., & Sim, M. (2003). Robust discrete optimization and network flows. *Mathematical Programming, 98*(1), 49–71.

Bertsimas, D., & Sim, M. (2004). The price of robustness. *Operations Research, 52*(1), 35–53.

Buchheim, C., Liers, F., & Oswald, M. (2008). Local cuts revisited. *Operations Research Letters, 36*(4), 430–433.

Buchheim, C., Liers, F., & Sanità, L. (2011). An exact algorithm for robust network design. In J. Pahl, T. Reiners, & S. Voß (Eds.) *Proceedings of the INOC*, INOC'11 (pp. 7–17). Springer.

Büsing, C., & D'Andreagiovanni, F. (2012). New results about multi-band uncertainty in robust optimization. In *Proceedings Experimental Algorithms - 11th International Symposium*, SEA 2012, Bordeaux, France, June 7–9, 2012, pp. 63–74.

Cacchiani, V., Jünger, M., Liers, F., Lodi, A., & Schmidt, D. R. (2016). Single-commodity robust network design with finite and Hose demand sets. *Mathematical Programming, 157*(1), 297–342.

Claßen, G., Koster, A. M. C. A., Coudert, D., & Nepomuceno, N. (2014). Chance-constrained optimization of reliable fixed broadband wireless neworks. *INFORMS Journal on Computing, 26*(4), 893–909.

Claßen, G., Koster, A. M. C. A., Kutschka, M., & Tahiri, I. (2015). Robust metric inequalities for network loading under demand uncertainty. *Asia-Pacific Journal of Operational Research, 32*(5), 1550038, 1–27

Duffield, N. G., Goyal, P., Greenberg, A., Mishra, P., Ramakrishnan, K. K., & van der Merwe, J. E. (1999). A flexible model for resource management in virtual private networks. In *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, SIGCOMM '99 (pp. 95–108). ACM Press.

Fingerhut, J. A., Suri, S., & Turner, J. S. (1997). Designing least-cost nonblocking broadband networks. *Journal of Algorithms, 24*(2), 287–309.

Goyal, N., Olver, N., & Shepherd, B. (2008). The VPN conjecture is true. *Proceedings of the STOC*, 443–450.

Gupta, A., Kleinberg, J., Kumar, A., Rastogi, R., & Yener, B. (2001). Provisioning a virtual private network: a network design problem for multicommodity flow. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*, STOC '01 (pp. 389–398). ACM.

Koster, A. M. C. A., Kutschka, M., & Raack, C. (2013). Robust network design: Formulations, valid inequalities, and computations. *Networks, 61*(2), 128–149.

Kouvelis, P., & Yu, G. (1997). *Robust discrete optimization and its applications*. Norwell, MA: Kluwer Academics Publishers.

Kutschka, M. (2013). Robustness concepts for knapsack and network design problems under data uncertainty. Ph.D. thesis, RWTH Aachen University, https://cuvillier.de/de/shop/publications/6558.

Liebchen, C., Lübbecke, M., Möhring, R., & Stiller, S. (2009). The concept of recoverable robustness, linear programming recovery, and railway applications. In R. K. Ahuja, R. H. Möhring, & C. D. Zaroliagis (Eds.) *Robust and online large-scale optimization: models and techniques for transportation systems* (pp 1–27). Springer.

Magnanti, T. L., Mirchandani, P., & Vachani, R. (1993). The convex hull of two core capacitated network design problems. *Mathematical Programming, 60*(1–3), 233–250.

Mattia, S. (2013). The robust network loading problem with dynamic routing. *Computational Optimization and Applications, 54*, 619–643.

Minoux, M. (1989). Networks synthesis and optimum network design problems: Models, solution methods and applications. *Networks, 19*(3), 313–360.

Onaga, K., & Kakusho, O. (1971), On feasibility conditions of multicommodity flows in networks. *Transactions on Circuit Theory, 18*(4), 425–429.

Orlowski, S., Wessäly, R., Pióro, M., & Tomaszewski, A. (2010). SNDlib 1.0–survivable network design library. *Networks, 55*(3), 276–286.

Pascali, F. (2009). Chance constrained network design. Ph.D. thesis, University of Pisa. https://etd.adm.unipi.it/t/etd-11262009-005543/.

Pesenti, R., Rinaldi, F., & Ukovich, W. (2004). An exact algorithm for the min-cost network containment problem. *Networks, 43*(2), 87–102.

Poss, M., & Raack, C. (2013). Affine recourse for the robust network design problem: Between static and dynamic routing. *Networks, 61*(2), 150–155.

Sanità, L. (2009). Robust network design. Ph.D. thesis, Università La Sapienza, Roma.

Schmidt, D. R. (2014). Robust design of single-commodity networks. Ph.D. thesis, Universtität zu Köln.

Soyster, A. L. (1973). Convex programming with set-inclusive constraints and applications to inexact linear programming. *Operations Research, 21*, 1154–1157.

Part III
Applications in Transportation and Logistics

Chapter 12
Service Network Design

Teodor Gabriel Crainic and Mike Hewitt

1 Introduction

The term *Service Network Design* (*SND*) is generally used to designate a set of issues and decisions aimed to plan the activities and resources of the *supply* side of a transportation system, in order to satisfy a given or estimated *demand* efficiently, profitably, and within the quality standards agreed upon with the customers generating this demand. *Service* is then understood as operating a vehicle, or a convoy, e.g., a railroad train, between two stations/terminals in the network, with or without intermediary stops, to transport a single or a group of people or freight loads. The service follows a given route on the appropriate infrastructure, and displays a number of physical, e.g., vehicle type and capacity, and operational, e.g., departure time, total trip duration and cost, characteristics. While all transportation systems and carriers offer "services" to their customers, SND occurs mainly in the context of *consolidation*-based transportation, an umbrella term for companies and systems, the *carriers*, which group and transport within the same vehicle several people who contracted the trip separately or several freight loads of different customers. In all cases, the alternative of a dedicated, direct transport is not economically justifiable or even feasible. Public-transport carriers in urban areas, by bus, light rail, and collective taxi, and those providing interurban transport by coach, train or airplane "consolidate" passengers who do not want or can move by

T. G. Crainic (✉)
CIRRELT and AOTI, Université du Québec à Montréal, Montréal, QC, Canada
e-mail: TeodorGabriel.Crainic@cirrelt.net

M. Hewitt
Information Systems and Supply Chain Management Department, Quinlan School of Business, Loyola University Chicago, Chicago, IL, USA
e-mail: mhewitt3@luc.edu

a dedicated vehicle between their respective origins and destinations. Postal and small-package transportation companies, less-than-truckload (LTL) motor carriers, railroads, ocean/maritime liner navigation companies, and land- and water (coastal, river, etc)-based intermodal carriers perform similar services for freight. Noticeable are the "new" transportation-system types introduced for urban, e.g., City Logistics, and interurban, e.g., Physical Internet and truck platooning, settings, which are heavily based on consolidation and resource sharing. Carriers may be publicly or privately owned/operated, while groups of carriers, operating under some form of cooperation agreement, may also be involved.

Carriers need to be profitable, while consolidation raises two challenges. First, that the vehicle movements, that is, the services offered, cannot be planned to address the demand of individual potential customers, but must satisfy as closely as possible the requirements of as many potential customers as possible (while probably not satisfying any of them entirely; contrasting taxi and public-transport services illustrates the point). This has implications for the service network, including on topology, i.e, where to propose and operate services, timelines, i.e., when to operate services, and performance measures, e.g., cost, efficiency, and quality of service. Second, that operations need to be efficient from the point of view of using the carrier's material and human resources. Indeed, one observes, on the one hand, a continuous increase in the size of vehicles on long-haul routes, e.g., mega container ships (capacity exceeding 20,000 twenty-foot equivalent containers), 120 to 130-car long trains running on the North American rail networks, and large passenger aircraft types. One also notices, on the other hand, that the cost of operating a service is greatly dependent upon the costs of vehicles and power units used for transport. System efficiency and cost reductions may then be achieved through economies of scale capacity utilization, obtained by assigning the most appropriate vehicles, and other associated resources (power units, people, etc.), to each movement, filling them well with the passengers or freight requiring transport, and routing them through multi-service *itineraries* and inter-service transfers at terminals. The availability of resources constrains the range of alternatives, however, while multi-service itineraries may imply additional costs and delays at terminals.

Trade-offs must thus be achieved in planning the service network, to balance customer demand for faster and cheaper transportation, on the one hand, and the pursuit of economies of scale and profitable and efficient carrier activities, on the other hand. Trade-offs must also be achieved among the various components of the carrier transportation system and operations as improving one aspect often has negative implications for other aspects, e.g., increasing the number of times a service is operated during a certain time interval improves customer service but may decrease the availability of resources for other services as well as increase congestion in terminals, thus deteriorating customer service. *Service Network Design* aim to address these issues network-wide and determine the services and itineraries to operate.

SND is closely related to network design. We emphasize these relations in this chapter, as well as the particular characteristics applications bring to SND. Several chapters in this book address SND in the context of such applications, namely, public transport (Chap. 17), motor carriers (Chap. 14), railroads (Chap. 13), navigation (Chap. 15), and City Logistics (Chap. 16). The goal of this chapter is to present a comprehensive overview of the general SND methodology, in terms of models, solution methods and utilization, that cuts across application fields. To focus the presentation, however, we will use in the following the vocabulary of consolidation-based freight carrier planning.

The chapter is organized as follows. Section 2 recalls the structure and main components of the physical and service networks of consolidation-based freight carriers, as well as the associated tactical planning issues, and the main *Service Network Design* (*SND*) formulation classes with their utilization within the carrier-planning processes. Section 3 is dedicated to the static problem setting and formulations, while Sect. 4 introduces the explicit representation of time and time-related attributes in the basic SND models. Section 5 broadens the scope of SND methodology to integrate the management of the resources required to operate the selected services. Addressing uncertainty within SND is the topic of Sect. 6. Section 7 proposes an historical view of the field, in terms of the models and main solutions methods specifically developed for various SND settings. We conclude in Sect. 8 with a number of research issues we deem important and challenging.

2 Problem Settings

We initiate this section with a brief description of the physical and service networks typical of consolidation-based freight carriers. We then proceed to discuss the associated planning of operations and introduce the main classes of service network design models proposed to address them.

2.1 Consolidation-Based Freight Carriers

Carriers providing consolidation-based services operate on an infrastructure network made up of terminals connected by physical, e.g., highways and rail tracks, or conceptual, e.g., maritime and air corridors, links. Terminals come in several designs and sizes, targeting particular transportation modes, e.g., rail marshaling/consolidation yards and stations, LTL motor-carrier breakbulk and regional terminals, and maritime and river ports. Terminals may be owned/managed by and dedicated to the carrier, e.g., railroad yards and LTL breakbulk terminals, or may be shared by several carriers irrespective of ownership and management, e.g., maritime ports and terminals, intermodal terminals, passenger airports, etc. Inter-terminal links may also be proprietary (but may still be used by other carriers for a fee), e.g., rail tracks in North America, or shared, e.g., rail tracks in Europe and roads and highways mostly everywhere.

Carriers operate single or multi-modal networks on the infrastructure. LTL motor carriers and railroads operating exclusively trucks and trains, respectively, are usually identified as single-mode. Postal/express-courier services, City Logistics systems, and container intermodal transportation often involve more than one transportation mode, the transfer of loads from one to the next taking place at intermodal terminals. Notice, however, that many carriers traditionally classified as single mode actually operate multi or intermodal networks, the latter occurring when freight packaged at origin, e.g., in containers, is not handled before it is unpacked at destination. Railroads, owning LTL motor carriers, and maritime shipping companies, owning railroads or motor carriers, illustrate this case when they plan services and freight movements on the entire network. Moreover, particular vehicle and convoy configurations (in terms of power, speed, capacity, etc.) are also often identified for planning purposes as "modes" with their own tariffs, due to their different performances in terms of costs and travel time. We therefore address multimodal networks in this chapter, each service being of a particular "mode" according to the infrastructure, vehicle and convoy configuration, speed and priority, etc .

Consolidation transportation carriers are organized into so-called hub-and-spoke networks. One identifies two main categories of nodes in such a network. The largest category consists of *local/regional terminals* where most of the demand from the corresponding regions is brought in to be transported by the system, and where the demand flows terminate their trips before being distributed to their final destinations. Rail stations, LTL regional terminal, most deep-sea and river/canal ports belong to this type. The *hubs* make up the second category. One finds in this category LTL breakbulks, major classification/blocking railroad yards, and major maritime ports for intermodal (container-based) traffic such as Hong Kong, Singapore, and Rotterdam. While these terminals play the same role as the regional terminals for their hinterlands, their main role is to *consolidate* the flows in and out of their associated regional terminals for efficient long-haul transportation and economies of scale.

Carriers offer service between origin and destination (*OD*) points corresponding to their terminals. The volume (or value or both) of most of the OD demands, identified in the following as *commodities* to recall that each may concern a specific product with specific transportation requirements, is too low, however, to justify a profitable direct service with reasonable service quality. Thus, for example, when the volume is too low with respect to the capacity of the usual vehicle for the corresponding distance, the cost of the transportation would yield tariffs few customers are willing to pay. Alternatively, waiting to fill up the vehicle with other demands to the same destination generally requires delays customers are not ready to accept. The combination of such phenomena gave rise to consolidation-based transportation, for freight and people, the number of commodities (OD demands) being significantly larger than the number of direct, origin to destination services operated by the carrier, which aims for economies of scale. Carriers thus first move low-volume loads available at a regional terminal to a hub, through what is known as feeder services. At hubs, loads are sorted (*classified* is the term used in several settings, e.g., freight railroads) and consolidated into larger flows, which are routed to other hubs by high-frequency, high-capacity services. Loads may thus go through more than one intermediary hub before reaching the regional-terminal destination, being transferred from one service to another or undergoing re-classification and re-consolidation. Notice that, when the level or value of demand justifies it, high-frequency, high-capacity services may be run between a hub and a regional terminal or between two regional terminals. Notice also that, more than one service, of possibly different modes, may be operated between consolidation and regional terminals.

A *service* follows a route through the physical network. It may be direct, without intermediary stops from the origin of the service to its destination, or it may include stops at one or several terminals to drop and pick up loads and, eventually, vehicles, e.g., car and blocks for railroads and trailers for LTL motor carriers operating multi-trailer road trains. The route may also include stops that serve purposes other than consolidation. As an example, governmental highway safety regulations often limit the number of hours a driver may drive before resting. Yet a transportation service may be longer, in terms of drive-time, than that limit. Thus, if one driver executes the service, the duration of the service would have to reflect the driver's need to rest while en route. To reduce the service's duration, the vehicle could instead stop at an intermediate location, wherein an exchange of drivers occurs. Note this intermediate location need not be a terminal in the physical network. Instead, the driver exchange could occur at a rest stop on a highway.

The set of services the carrier selects to operate makes up the *service network*, which will be used to respond to the demand of customers who require their loads to be transported between particular origins and destinations. In most planning problems addressed with SND methodology, these locations are assumed to be the regional or hub terminals, planning processes not targeting the local pick up and delivery activities to bring loads to origin terminals to initiate transportation and to distribute them at destination. We follow this approach in this chapter.

Demand is thus multi-commodity, each commodity being defined by its specific origin (carrier terminal), destination (a different terminal), as well as commodity (product) related physical characteristics (e.g., weight and volume) and service requirements in terms of delivery conditions, type of vehicle (e.g., refrigerated, multi-platform for containers or vehicles, etc.), and so on. Two additional attributes are usually associated to each commodity. First, a unit profit or (transportation) cost, the latter often related to the vehicle type used. Second, time-related requirements, i.e., a date when it is delivered to the origin terminal, as well as a due date (or time interval) to be delivered at destination. The latter is often linked to the level of service quality required; a unit penalty cost for late delivery or a unit cost for the total delivery time or delay is generally associated to this service-quality level.

Carriers respond to demand by offering a network with more or less scheduled services. Demand itineraries will move the corresponding loads through this service network, each *itinerary* being defined by the sequence of services used and the operations to be performed (e.g., transfer or re-classification and consolidation) and intermediary terminals. The service schedule could simply be a certain *frequency of service* or *number of departures*, i.e., the number of times the "same" service is run during the length of time the carrier uses to define its recurring operations (e.g., 1 day for LTL motor carriers, 1 week for railroads, longer for containership liners), also called *schedule length* in the following. A more precise service schedule gives the departure time from the origin terminal, arrival and departure times at each intermediary terminal, and the arrival time at destination. This information may be strict, as for most European and Canadian railroads and regular containership liners, or more of an "indicative" nature, the schedule being eventually modified to account for particular events (e.g., the need to pass a direct service for an important customer) or how much freight is already loaded. Often independent of its precision, a schedule is effective for a certain period of time, often related to seasonal variations in demand and operation conditions. We use in this chapter the term *season*, often found in the literature, to refer to this period of time during which the schedule and the services it contains are repeatedly performed.

2.2 Planning and Service Network Design Models

The planning activities consolidation-based carriers undertake may be broadly classified into three levels, similarly to most complex systems. Strategic planning involves long-term decisions on system design, operation strategies, and acquisition of major resources. Tactical planning is dedicated to building an efficient service network and schedule. Short term planning involves monitoring activities and performance, adjusting plans, managing resources and operations.

We focus on tactical planning in this chapter, as it involves the arguably strongest connection to network design, through the service network design modeling framework. We discuss at the end of this section the utilization of SND in varied contexts, including the other levels of planning.

A hub-and-spoke network concentrates the multi-commodity flows and allows a much higher frequency of service for the consolidated demand loads, while providing a more efficient utilization of resources, economies of scale for the carrier, and lower tariffs for the customers. The drawbacks of this type of organization are possibly increased delays for demand due to longer routes and more time spent going through terminals, which play a major role within consolidation-based transportation systems. This role is significantly broader than the loading and unloading of freight. Vehicle and freight sorting and consolidation, convoy make up and break down, and vehicle transfer between services are all time and resource-consuming operations performed in terminals. Indeed, if not planned properly, these additional activities and delays may cancel the benefits of the hub-and-spoke strategy.

Tactical planning aims to build a transportation plan and schedule to mitigate the drawbacks of consolidation, satisfy customer demand and service-quality requirements, and operate profitably and efficiently. It addresses the system-wide planning of operations to decide the selection and scheduling of services, the transfer and consolidation activities in terminals (as well as the convoy makeup and dismantling for railroads, and road and barge trains), the assignment and management of resources to support the selected services, and the routing of freight of each particular demand through the resulting service network. The goal is cost-efficient operation together with timely and reliable delivery of demand according to customer specifications and the service-quality targets of the carrier.

Such planning problems are difficult due to the strong interactions among system components and decisions and the corresponding trade-offs between operating costs and service levels that need to be achieved. Consider, for example, strategies based on re-consolidation and routing through intermediate terminals, which could be more efficient when direct services are offered rarely due to low levels of traffic demand. Such strategies would then probably result in higher equipment utilization and lower waiting times at the original terminals; hence, in a more rapid service for the customer. The same strategies would also result, however, in additional unloading, consolidation, and loading operations, creating larger delays and higher congestion levels at terminals, as well as a decrease in the delivery reliability of the shipment. Alternatively, offering more direct and frequent services would imply faster and more reliable service for the corresponding traffic and a decrease in the level of congestion at some terminals, but at the expense of additional resources, thus increasing the costs of the system.

Service network design is the methodology of choice to support tactical planning of consolidation-based carriers. A SND model integrates the issues discussed above and addresses them jointly at a network-wide level. It assumes a given physical system, infrastructure, resources, operation strategies, and it optimizes for an estimation for the season of the *regular demand* (e.g., 75–80% of the pick demand on a normal operating day). It integrates two major sets of decisions, the selection of the service network, that is, the routes—origin and destination terminals, physical route and intermediate stops—and schedules, or frequencies, on which services will be operated, and the itineraries, sequences of services, terminals, and operations, used to move the freight of each demand. Operating rules specifying, for example, how resources may be assigned and handled and how cargo and vehicles may be sorted and consolidated, are often specified as part of the service network. The SND model yields a *transportation plan* specifying operations for the given *schedule length*, to be repetitively applied for the next *season*.

Static problem settings, Sect. 3, assume that neither demand, nor any other problem characteristic varies during the schedule length considered. *Time-dependent* problem settings, Sect. 4, include an explicit representation of demand and activities in time and target the selection of *scheduled* services to support decisions related to *when* services leave and arrive at terminals on their routes. In all cases, the minimization of the total operating costs is the primary optimization criterion, reflecting the traditional objectives of freight carriers and expectations of customers to “get there fast at lowest possible cost”. Increasingly, however, customers not only expect low rates, but also high-quality service, measured by speed, flexibility, and reliability. Service performance measures reflecting these expectations and modeled, in most cases, by delays incurred by freight and vehicles or by the respect of predefined performance targets are then added to the objective function of the network optimization formulation. The resulting generalized cost function thus captures the trade-offs between operating costs and service quality.

The two sections that follow detail the issues and models associated to the two major problem settings for service network design described above (static and time-dependent). We then turn to problem settings and SND models addressing the needs and challenges of integrating resource-management concerns into tactical planning (Sect. 5). Section 6 continues this discussion addressing the issue of the explicit representation of the *uncertainty* inherent to any system and human endeavor.

We conclude this general presentation with a short discussion on the utilization of tactical planning SND methodology. One first must recognize that there are different contexts and mindsets with respect to service network design, from proposing a new plan yearly or twice a year (alternating between Summer and Winter) by railroads and shipping companies, to much shorter seasons of 3–4 months, or even solving a model weekly, as is typically performed by LTL motor carriers. The same model may then be used for a much shorter period, weekly (e.g., railroads) and daily (e.g., LTL trucking and City Logistics) to re-optimize and adjust the plan and operations to current conditions. What may be adjusted depends strongly on the application context. For example, while LTL motor carriers may quite freely cancel and add truck departures, such strategies are normally much more difficult for railroads, which will rather update the actual demands assigned to blocks and trains. Obviously, the scope of the SND model may be more focused when in plan-adjustment mode, parts of the system which should not be modified being fixed.

A different class of problem settings calls upon SND models to yield plans to be applied once only. Consider, for example, the case of City Logistics when one has little or no restrictions on calling up for duty on very short notice facilities, vehicles, and people. The planning of such systems is better performed close to operation-time. The so-called *day-before* SSND models, similar to those presented in this chapter, are then used before each operation period based on updated data. Note that, in this context, there are no impacts of today's decisions on the system status and capability for the next days. When this is not the case, e.g., when transport or storage activities require several periods, the time-dependent SND models may be used in a rolling-horizon approach. Then, the SND yields decisions for "now" (a somewhat limited number of periods) and for a number of following periods. The latter are not to be implemented, but bring to the model an evaluation of the consequences of today's actions on future capabilities. Then, today's proposed actions are implemented, time is advanced, information is updated, and the process repeats.

SND models may also be used as policy and performance-evaluation tools for strategic scenarios. Operational details need to be abstracted in such cases as well as, according to the planning horizon contemplated, the demand and cost figures. Governmental institutions and funding or control organizations, such as the World or Asian Development banks, may also use SND models as a simulation tool in the context of cost-benefit analyses, with appropriate approximation of carrier and shipper characteristics. Finally, generalized service network design models may be built to answer strategic-level decisions such as the number, locations, and characteristics of terminals to build, rent or use, the construction of dedicated infrastructure, the types of vehicles to use and the dimensions of the fleets, etc.

3 Static SND

Let $\mathscr{G}^{\mathrm{PH}} = (\mathscr{N}^{\mathrm{PH}}, \mathscr{A}^{\mathrm{PH}})$ represent the physical infrastructure network, where $\mathscr{N}^{\mathrm{PH}}$ stands for the set of facilities, hubs and regional terminals, connected by the the physical or conceptual links of set $\mathscr{A}^{\mathrm{PH}}$. The goal of such models is to select from a set of potential services $\Sigma = \{\sigma\}$ either the service network only or the services and their frequencies to satisfy the demand for transportation of a set $\mathscr{K}$ of origin-destination (OD) commodities, each $k \in \mathscr{K}$ requiring to move a quantity of freight d^k between its origin $O(k)$ to its destination $D(k)$.

A static SND model is built on a network $\mathscr{G} = (\mathscr{N}, \mathscr{A})$ that is defined in the static case on the physical nodes of the system, i.e., $\mathscr{N} = \mathscr{N}^{\mathrm{PH}}$. With respect to the arc set $\mathscr{A}$, its composition depends on whether the potential services are *single-leg*, with no intermediary stop between the origin and destination terminals, or *multi-leg*. In the former case, $\mathscr{A} = \Sigma$. In the latter, $\mathscr{A} = \mathscr{L} = \bigcup_{\sigma \in \Sigma} \mathscr{L}(\sigma)$, where each multi-leg service is defined by a sequence of n service legs, collected in set $\mathscr{L}(\sigma) = \{l_i(\sigma) \mid i = 1, \ldots, n\}$, each service leg being a path in the physical network connecting two consecutive terminals on the route of service σ. We let $\sigma_a \in \Sigma$ denote the service associated with arc $a \in \mathscr{A}$.

Associated with service σ is a capacity $u(\sigma)$, which can be leg specific $u(l_i(\sigma))$, $l_i(\sigma) \in \mathscr{L}(\sigma)$, representing the total volume of freight the service may load and haul, as well as the cost f_σ incurred when doing so. In terms of arcs $a \in \mathscr{A}$, the attribute u_a takes the value $u(l_i(\sigma))$ associated with the service leg, $l_i(\sigma)$, modeled by that arc. Note that capacity may be measured in volume, tonnage, length (particularly for railroads), and number of units (e.g., containers for intermodal navigation and rail), that more than one capacity measure may be active in any given problem setting, and that particular capacities for particular products can also be imposed. To simplify the presentation and if not otherwise indicated, however, we continue with a single capacity restriction in this chapter.

Note that with both single-leg and multi-leg services, $\mathscr{A}$ may contain multiple arcs that have the same origin and destination, but differ in one of these attributes. In a single-leg setting, the carrier may choose from a market of third party carriers for the execution of the same service, with each carrier offering a different cost and capacity. In a multi-leg setting, two services may involve different sequences of legs, but those sequences may overlap.

In applications wherein a service can be executed multiple times, the SND models the frequency with which a service is executed with the non-negative integer variable $y_\sigma \in \mathbb{Z}_+, \sigma \in \Sigma$. When the decision is whether a service should be executed, binary variables $y_\sigma \in \{0, 1\}$, $\sigma \in \Sigma$ are used. We note that adapting the SND to applications wherein vehicle capacity is measured along multiple dimensions (e.g., weight and volume) is straightforward.

The arc $a = (i, j) \in \mathscr{A}$ also models the opportunity to transport a commodity on the transportation leg (i, j), which incurs a per-unit cost c_a. In some applications, this cost can depend on the commodity being transported, and thus the cost parameter is also indexed by the commodity, k, yielding c_a^k. We can consider different types of *flow* variables to model the routing of commodities on such arcs. The first type of variable is of the form $x_a^k \geq 0,\ a \in \mathscr{A}, k \in \mathscr{K}$, and prescribes the amount of commodity k that travels on arc $a \in \mathscr{A}$. The second is named similarly, but is instead defined over the range [0, 1], and models the percentage of commodity $k's$ demand that flows on arc (i, j). Modifying the SND to accommodate one type of flow variable instead of another is an exercise in ensuring the model correctly calculates the total flow on each arc. Both sets of flow variables allow a commodity to be split, and then routed along multiple paths from its origin to its destination.

In settings wherein this is inappropriate or undesirable (e.g., breaking down a sealed pallet is not allowed), the model must restrict a commodity to travel on a single path from its origin to its destination. This can be done by restricting the x_a^k variables to be binary, wherein they model whether commodity k travels on arc a. In this chapter, we focus on the first form of flow variable, which represents the amount of a commodity that flows on a leg.

As noted in the description of the problem setting, shipments travel on itineraries, which in the context of our network $\mathscr{G} = (\mathscr{N}, \mathscr{A})$ can be represented by paths. As a result, flow variables can be defined in terms of paths. The same options (continuous, fractional, binary) for the domains of path-based flow variables exist as for arc-based flow variables with each option having an analogous modeling implication. The formulation we present next can be modified to prescribe decisions in terms of paths, similar to what was presented in Chap. 2.

Formally, the SND seeks to

$$\text{Minimize} \quad \sum_{\sigma \in \Sigma} f_\sigma y_\sigma + \sum_{k \in \mathscr{K}} \sum_{a \in \mathscr{A}} c_a^k x_a^k \tag{12.1}$$

Subject to

$$\textstyle\sum_{a \in \mathscr{A}_i^+} x_a^k - \sum_{a \in \mathscr{A}_i^-} x_a^k = \begin{cases} d^k, & \text{if } i = O(k), \\ -d^k, & \text{if } i = D(k), \\ 0, & \text{otherwise}, \end{cases} \qquad \forall i \in \mathscr{N}, \forall k \in \mathscr{K}, \tag{12.2}$$

$$\textstyle\sum_{k \in \mathscr{K}} x_a^k \leq u_a y_{\sigma_a}, \qquad \forall a \in \mathscr{A}, \tag{12.3}$$

$$y_\sigma \in \mathbb{Z}_+, \qquad \forall \sigma \in \Sigma, \tag{12.4}$$

$$x_a^k \geq 0, \qquad \forall a \in \mathscr{A}, \forall k \in \mathscr{K}. \tag{12.5}$$

where for each $i \in \mathscr{N}$ we define the sets $\mathscr{A}_i^+ = \{(i', j) \in \mathscr{A} : i' = i\}$ and $\mathscr{A}_i^- = \{(j, i') \in \mathscr{A} : i' = i\}$.

The objective of the SND is to minimize the sum of the fixed costs associated with selecting and executing transportation services (the first term in (12.1)) and the variable costs associated with transporting commodities on legs associated with those services (the second term in (12.1)). Constraints (12.2) are often referred to as *flow-balance* constraints and ensure that all of a commodity's demand departs from its origin (the first case), arrives at its destination (the second case), and departs any other locations at which it arrives (the third case). The expression on the left-hand side of the *linking* constraints (12.3) computes the total amount of demand that travels on arc $a \in \mathscr{A}$, whereas the expression on the right-hand side computes the total amount of capacity on that arc that is provided by the selected services. Thus, the constraint ensures that sufficient capacity is paid for. Constraints (12.4) define the domain of variables that indicate how often services are executed, while constraints (12.5) define the commodity routing decision variables, as well as their domains. .

Variants of the SND use commodity flow variables that represent the percentage (not the portion) of a commodity's demand that flows on an arc. This necessitates changing constraints (12.5) to require that $x_a^k \in [0, 1]$, replacing the expression on the left-hand-side of constraints (12.3) with $\sum_{k \in \mathscr{K}} d^k x_a^k$, dividing the right-hand-sides of constraints (12.2) by d^k, and multiplying the second expression in the objective (12.1) by d^k. Alternately, modeling a problem wherein the demand of each commodity flows along a single path necessitates changing Constraints (12.5) to instead require that $x_a^k \in \{0, 1\}$, in addition to the other changes noted above. Finally, for variants of the SND wherein the decision is whether a service should be executed, and not how many times it should be executed, constraints (12.4) should be changed to $y_\sigma \in \{0, 1\}$.

4 Time-Dependent SND

In *time-dependent* problem settings, demand $k \in \mathscr{K}$ is further characterized by an *availability time* $o(k)$ at origin $O(k)$ and a due date $d(k)$ at destination $D(k)$. Services are also characterized not only by their origin $O(\sigma)$, destination $D(\sigma)$, and set of legs $\mathscr{L}(\sigma) = \{l_i(\sigma) \mid i = 1, \ldots, n\}$, but also by a schedule indicating the departure and arrival times, $o(l_i(\sigma))$ and $d(l_i(\sigma))$, at the origin and destination terminals, respectively, of each of its legs $l_i(\sigma) \in \mathscr{L}(\sigma)$. Services are further characterized by a total duration $\tau(\sigma)$, that includes the time spent in terminals and the moving time associated to each leg $\tau(l_i(\sigma))$.

To capture these time-related characteristics of demand and service, SND models are generally defined on a time-space network $\mathscr{G} = (\mathscr{N}, \mathscr{A})$, that is typically built by extending the network $(\mathscr{N}^{\mathrm{PH}}, \mathscr{A}^{\mathrm{PH}})$ along the dimension of time for the fixed duration of the *schedule length*. The selected service network specifies in this case the movements through space and time of the vehicles and convoys of the various modes considered, while itineraries perform the same role for the transportation of time-dependent demand. When formulated on such a network, the SND is often referred to as a *Scheduled Service Network Design* (*SSND*) model.

The time-space network is built by partitioning the schedule length into non-overlapping periods of time, wherein all activities at terminals during a period will be modeled as occurring at the same time. Often, these periods are of the same length. As an example, a planning horizon consisting of seven 24-h days may be partitioned into 14 half day (of 12 h each when the terminal operate continuously) or 168 one hour time periods. Then, demand that arrives at a terminal during a period, e.g., between 04:01 and 05:00, is modeled as being there by the end of the preiod, e.g., 05:00, and hence can be consolidated and loaded on services that leave the terminal in the same or following periods.

The time periods represented at one terminal may differ, however, from those represented at another terminal. This is often due to different terminals serving different purposes within a network. For example, some terminals may primarily serve as the interface between customers and the transportation network. For these terminals, it may be sufficient to only represent the time periods during which shipments become available or are due. Other terminals may primarily serve as consolidation centers. At these terminals vehicles may arrive and depart throughout the day, and thus more time periods may need to be modeled. Similarly, the length of a time period modeled at a node may depend on both the physical terminal and the start of the time period itself.

More formally, for each terminal $i \in \mathscr{N}^{\mathrm{PH}}$, the network is based on a set, $\mathscr{T}_i = \{t_1^i, t_2^i, \ldots, t_{m_i}^i\}$ of periods of time during which activities may occur at that terminal. We let $\mathscr{T} = \cup_{i \in \mathscr{N}^{\mathrm{PH}}} \mathscr{T}_i$ denote the set of all such time periods. The node set $\mathscr{N}$ then consists of nodes of the form $(i, t_p^i), i \in \mathscr{N}^{\mathrm{PH}}, t_p^i \in \mathscr{T}_i$, that represent a terminal during a period in time.

The arc set $\mathscr{A}$ consists of multiple types of arcs. The first represents the execution of the service legs. Specifically, for service $\sigma \in \Sigma$ and leg $(i, j) \in \mathscr{L}(\sigma)$, $\mathscr{A}$ will contain an arc of the form $a = ((i, t_p^i), (j, t_q^j))$, which represents the departure of that service leg from terminal i at time t_p^i to arrive at location j at time t_q^j. As with the SND, we denote the underlying service associated with arc a by σ_a. The time point t_p^i is usually chosen so that $t_p^i \leq o(l_i(\sigma))$. Namely the arc models that the leg departs no later than its scheduled departure time. Similarly, the time point t_q^j is usually chosen so that $t_q^j \geq d(l_i(\sigma))$. Namely, the arc models that the service arrives no earlier than its scheduled arrival time. As with the SND, associated with a service, its legs, and the corresponding arcs are capacity and cost attributes.

Recall that we focus on a time-expanded network that enables the SSND to prescribe a plan that is repeatable. This is done by modeling activities (e.g., transportation) that would end after the end of the scheduling period as instead ending after the beginning. This is done by making the relevant arcs of $\mathscr{A}$ *wrap-around*. To be more precise, consider when the departure of service leg (i, j) at time t_p^i would imply arriving at j at a time period that is later than the last time period, $t_{m_j}^j$, modeled for terminal j. To model that the schedule is assumed to be repeated, an arc $a = ((i, t_p^i), (j, t_q^j))$ is then created wherein $t_q^j < t_p^i$. For example, consider a schedule length that is a business week and time periods that correspond to days. If service leg (i, j) has a 2 day duration, then a Friday departure from i

could be modeled with an arc that arrives at the node that models terminal j on Tuesday.

The second type of arc, often referred to as a *holding arc*, is of the form $a = ((i, t^i_p), (i, t^i_{p+1}))$ and represents the opportunity to hold goods or a resource at terminal i from period t^i_p to period t^i_{p+1}. As with transportation arcs, wrap-around arcs of the form $a = ((i, t^i_{m_i}), (i, 1))$ are created to represent holding a shipment (or allowing a resource to idle) from one schedule period to the next. The attribute u_a of a holding arc can be used to model the capacity terminal i has for holding goods. Similarly, the variable cost attribute c_a may be used to model the cost associated with holding goods for a period at terminal i (which may also depend on the commodity, k). The fixed cost attribute, f_a, may be used to model the cost associated with a resource idling at a location from one period to the next. However, we only consider variable costs in the model presented below.

Like the SND, the SSND considers two sets of decision variables. The first type of decision variable, $y_\sigma \in \mathbb{Z}_+, \sigma \in \Sigma$, models the number of times the transportation service, σ, is executed, which in turn implies the number of times its scheduled legs, $\mathscr{L}(\sigma)$, are executed. Selection-type decision variables are not typically associated with holding arcs in $\mathscr{A}$. However, situations wherein storage capacity at a location for a fixed period of time is paid for in fixed lot sizes (e.g., a storage cage) could be modeled with similarly-defined y variables. Like the SND, the domain of these variables, either service or holding, can be binary.

The second, $x^k_a \geq 0, a \in \mathscr{A}, k \in \mathscr{K}$, represents the amount of commodity k's demand that travels on arc $a \in \mathscr{A}$. Note that these commodity flow variables are defined over both types of arcs, those that represent transportation services, and those that represent the commodity being held at a terminal. As in the SND, these x variables can also be restricted to take on binary values. Alternately, and again like the SND, these x variables can instead be used to model the fraction, of k's demand that travels on the arc.

Thus, the SSND seeks to

$$\text{Minimize} \sum_{\sigma \in \Sigma} f_\sigma y_\sigma + \sum_{k \in \mathscr{K}} \sum_{a \in \mathscr{A}} c^k_a x^k_a \tag{12.6}$$

Subject to

$$\sum_{a \in \mathscr{A}^+_{(i,t^i_p)}} x^k_a - \sum_{a \in \mathscr{A}^-_{(i,t^i_p)}} x^k_a = \begin{cases} d^k, \text{ if } i = O(k), t^i_p = o(k), \\ -d^k, \text{ if } i = D(k), t^i_p = d(k) \\ 0, \text{ otherwise,} \end{cases}$$

$$\forall (i, t^i_p) \in \mathscr{N}, \forall k \in \mathscr{K}, \tag{12.7}$$

$$\sum_{k \in \mathscr{K}} x^k_a \leq u_a y_{\sigma_a}, \forall a \in \mathscr{A}, \tag{12.8}$$

$$y_\sigma \in \mathbb{Z}_+, \quad \forall \sigma \in \Sigma, \tag{12.9}$$

$$x_a^k \geq 0, \quad \forall a \in \mathscr{A}, \forall k \in \mathscr{K}, \tag{12.10}$$

where for each $(i, t_p^i) \in \mathscr{N}$ we define the sets $\mathscr{A}^+_{(i,t_p^i)} = \{a = ((i', t_p^{i'}), (j, t_q^j)) \in \mathscr{A} : i' = i, t_p^{i'} = t_p^i\}$ and $\mathscr{A}^-_{(i,t_p^i)} = \{a = ((j, t_q^j), (i', t_q^{i'})) \in \mathscr{A} : i' = i, t_p^{i'} = t_p^i\}$.

Each constraint set in the SSND has a direct analog in the SND. Note that the right-hand-side values of the flow balance equations (12.7) depend on the available and due times for the commodity. Also, note that constraints (12.8) and (12.9) are only defined for arcs in $\mathscr{A}$ that correspond to transportation services.

We complete this introduction to SSND models recalling that the time-space networks are a modeling tool and cannot capture all the temporal aspects of the problem. A continuous-time representation of the schedule length and associated events and decisions may be contemplated. It its most general setting, the SND model appears very complicated, however, as the time attributes of each service (and, thus, each itinerary) becomes part of the decision variable, including when several occurrences of the service are looked for (as in the model of this section). Such a model would also require a significant amount of constraints governing arrivals, departures, and activity synchronization at terminals. Moreover, such an approach does not fit well the applications where schedules are not strict and one only search for a number of departures within a given time interval.

Discretization of time, as described in this section, is thus the preferred methodology in time-dependent settings. Then, the question is what discretization granularity should one use. On the one hand, a fine granularity, yielding short time periods, provides the means to a detailed representation of time and time-related activities. But, it makes for huge time-space networks with dire consequences on the problem-solving capabilities of the current exact and metaheuristic methods, even when mathematical techniques and the restrictions of the application are used to reduce the network. A coarser granularity alleviates partially this problem, but may result in a poorer representation of decisions and operations in time. In most cases reported in the literature, the granularity is decided based on the application at hand, the experience of the researcher, and the power of the solver and computer available. We present in Sect. 7 the *Dynamic Discretization Discovery*, a new and very promising algorithmic strategy introduced recently to address this issue by an iterative generation the time-space network.

5 Broadening the Scope of SND: Integrating Resource Management

A number of additional considerations may characterize the carrier transportation system and its planning, enriching and challenging service network design methodology. Several such issues are particularly relevant for specific transportation modes (rail, trucking, navigation, public transport) or problem settings (City Logistics) and are discussed in the associated chapters. In this section, we focus on an important issue of general relevance, namely, the integration resource-management concerns into SND models for tactical planning.

Carriers need resources to execute services, including equipment and manpower. For example, even though automation is becoming more and more prevalent in terminals, human resources are still needed to load/unload freight into/from outbound/inbound vehicles or containers, as well as handle and classify freight, vehicles or containers. Transportation activities also require resources. All modes require some type of power unit (tractors or trucks in trucking, locomotive engines in rail, planes in air, and vessels in maritime), one or several carrying units (trailers for trucking, rail cars for rail), an operator (truck driver, railroad engineer also called engine or train driver outside North America, air pilot, and ship captain), and sometimes a whole crew (particularly in air and sea).

These resources are generally scarce. There are several reasons for this fact. First, carriers aim continuously to control and hopefully reduce their operating costs to improve their market share and profitability. Consequently, the number of the power units and vehicles a carrier maintains has been drastically reduced to fit the forecast level of activity. There are precious few units available in most air, rail, trucking, and navigation carrier systems in case "something happens". Most resources are also expensive to acquire (e.g., power units), or there are few available for renting or acquisition, the shortage of truck drivers in North America being a perfect illustration. Leasing the appropriate number of the "right" type of intermodal rail cars is also an increasingly serious issue, at least in North America.

A second phenomenon impacting the availability of resources where and when needed is the unbalance inherent in trade. Indeed, the very nature of why trade is initiated (the desire for something available somewhere else), makes the commercial exchanges among countries, regions, and cities unbalanced not only in monetary value, but also in the type and quantity of products exchanged. This results in an unbalanced resource distribution among the terminals of the system. Power units and vehicles at destination and unloaded, become empty and available for the next operation. Yet, very often, these vehicles are not of the appropriate type or not available at the appropriate moment to load the outgoing freight. There is therefore a shortage of the appropriate vehicles, while those on location are needed somewhere else. Moving power units and vehicles "empty" to re-balance the system is called *repositioning* (deadheading for crews, especially in the air industry), and may represent a significant cost item for the carrier.

Not surprisingly, carriers have always aimed to minimize such operations and costs. Traditionally, however, the literature identified this problem as operational and addressed it, through more or less sophisticate network flow models, over rather short planning horizons, given the tactical plan. A somewhat more integrative approach computed an origin-destination matrix of empty vehicles, based on the OD-demand matrices, and distributed over the network jointly with the regular demand. None of these approaches works directly on the design of the service network.

The first integrative approaches focused on the most expensive resources, planes, ships, and rail engines. It assumed one unit of resource for each service and required that the number of selected services entering a terminal equals the number exiting the terminal. The corresponding *design-balanced SND* formulations extends the previous models by adding the set of node-degree constraints

$$\sum_{(i,j)\in\mathscr{A}_i^+} y_{ij} - \sum_{(j,i)\in\mathscr{A}_i^-} y_{ji} = 0, \quad \forall\, i \in \mathscr{N}. \tag{12.11}$$

Notice that, adding design-balanced constraints to SND formulations greatly complicates the search for high-quality solutions as, for example, even finding an initial solution is no longer straightforward (the rounding of the linear relaxation no longer guarantees a feasible solution). Moreover, the size of the formulation is increased, as is the computational effort to address arc-based models. On the other hand, note that such constraints naturally imply that resources move on cycles. The cycles may be of different time lengths (controlling cycle duration requires appropriate constraints) and may start at different periods during the schedule length. They are all, however, anchored at the terminal to which the resource is assigned. *Cycle-based* formulations thus appear natural.

Let $\Theta = \{\theta\}$ stand for the set of feasible cycles the units of the resource considered may perform, f_θ the "fixed" cost of selecting and operating the resource cycle $\theta \in \Theta$, and δ_θ^σ the cycle-to-service assignment indicator, where $\delta_\theta^\sigma = 1$ if the resource performing cycle $\theta \in \Theta$ may support service $\sigma \in \Sigma$, and 0 otherwise. Define the binary decision variable $y_\theta = 1$, if cycle $\theta \in \Theta$ is selected, and 0 otherwise,. The SSND with single resource management then becomes (to simplify the presentation, we display the formulation for the single-leg service case):

$$\text{Minimize} \sum_{\sigma\in\Sigma} f_\sigma y_\sigma + \sum_{\theta\in\Theta} f_\theta y_\theta + \sum_{k\in\mathscr{K}} \sum_{a\in\mathscr{A}} c_a^k x_a^k \tag{12.12}$$

subject to constraints (12.7)–(12.9) enriched with

$$y_\sigma \leq \sum_{\theta\in\Theta} \delta_\theta^\sigma y_\theta, \quad \forall \sigma \in \Sigma, \tag{12.13}$$

$$y_\theta \in \mathbb{Z}_+, \quad \forall \theta \in \Theta, \tag{12.14}$$

where the objective function aims to minimize the selection and operation costs of services and resources, plus the cost of moving the demand flows, while constraints (12.13) link the existence of services and the resources required to operate them.

A more general *Scheduled Service Network Design with Resource Acquisition and Management*, *SSND-RAM*, problem includes not only several types of resources, but also integrates tactical, service network design-related decisions, and strategic, resource-acquisition and allocation decisions. In the SSND-RAM model presented herein, resources are differentiated by relevant characteristics, e.g., capacity, traction power, speed, energy and emission, scheduling rules, etc. The model also considers the additional "resource" of executing a service by a third party rather than by a resource owned or leased. Calling on such a resource incurs costs that are greater than executing the service with an owned resource, but may be valuable when, for example, moving a resource into and out of a somewhat remote region is costly. Moreover, the carrier does not have to worry about how the utilization of the third-party resource outside the execution of the designated service. Tactical planning is thus selecting services with costs and capacities that can be influenced by the type of resource supporting them, including the outsourcing possibility. To simplify the presentation, services in the following model require one unit of resource only to operate. Resources, on the other hand, are assigned to specific terminals and must return to their home terminals at least once during the tactical planning horizon.

The model also addresses strategic decisions related to fleet acquisition and management, e.g., how many resources of each type should be acquired (or rented, depending on the resource type and supplier), to what terminal new resources should be assigned, and between which terminals currently existing resources should be reassigned. Costs associated with these strategic decisions include, e.g., the unit purchase or renting cost, the additional salary or signing bonus associated with hiring the required personnel to operate the resource, and the transportation costs associated with re-allocating a resource from a home terminal to another.

The problem and decisions may be represented schematically as in Fig. 12.1. An integrated SSND-RAM formulation captures those decisions through a two-layer time-space network, illustrated in Fig. 12.2 for the decisions related to a single resource type. The SSND layer, on the right of the figure, corresponds to the tactical-planning decisions on service choice and commodity transportation. It is similar to the SSND models of the previous sections. The resource acquisition and allocation decisions are modeled on the strategic RAM layer, on the left of the figure.

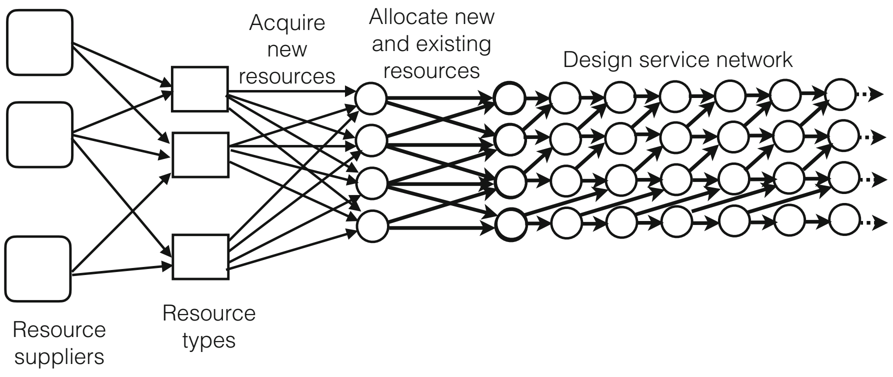


Fig. 12.1 Network model of SSND-RAM strategic and tactical decisions

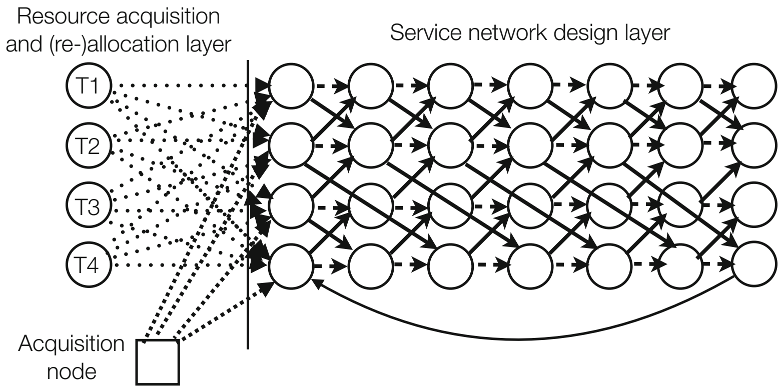

Resource acquisition and (re-)allocation layer
Service network design layer
T1
T2
T3
T4
Acquisition node

Fig. 12.2 SSND-RAM network model of strategic and tactical decisions

The SSND layer and notation is mostly similar to that of Sect. 4, adjusted for multiple resources. Let $\mathscr{R}$ stand for the set of available resources, f_i^r the fixed cost (salaries, maintenance, etc.) of operating a unit of resource of type $r \in \mathscr{R}$ that is assigned to terminal $i \in \mathscr{N}^{\text{PH}}$, and I_i^r the quantity of resources of type r initially assigned to terminal i. Let also Θ_i^r be the set of potential cycles a resource of type r assigned to terminal i can execute, $\Theta^r = \cup_{i \in \mathscr{N}} \Theta_i^r$ and $\Theta = \cup_{r \in \mathscr{R}} \Theta^r$. The cycle-to-service assignment indicator δ_θ^σ, links services and resources as previously. When service costs and capacities vary according to the assigned resource, the notation becomes f_σ^r and $u(\sigma, r), \sigma \in \Sigma,\ r \in \mathscr{R}$, respectively. Notice that a resource-independent fixed service selection cost, f_σ, may still be associated to a service modeling, e.g., the salaries of the officers of a liner ship. Finally, F_σ^r represents the fixed cost of operating service σ with a third party-owned resource of type r.

The RAM layer adds a few nodes, $\mathscr{N}'$, and arcs, $\mathscr{A}'$, to the time-space network, together with associated parameters and decision variables. There are two types of nodes in this layer, which are (1) symbolically defined at period 0, before the first period of the schedule length, and (2) connected to all first representations of the terminal nodes in the SSND layer. To simplify the presentation, and without loss of generality, we do not indicate the period 0, unless necessary.

A unique node, A, represents the acquisition of new resources. The corresponding arcs $(A,\ i, t_1^i),\ (i, t_1^i) \in \mathscr{N}$, represent the allocation of newly acquired resources to terminal i at the first period of activity at that terminal. Let h_i^r be the total cost of acquiring a new unit of resource $r \in \mathscr{R}$ and allocating it to terminal $i \in \mathscr{N}^{\text{PH}}$.

The second type of node is used to model the re-allocation of existing resources. A node i' is added at period 0 for each terminal $i \in \mathscr{N}^{\text{PH}}$, the arcs $(i',\ (j, t_1^j)), (j, t_1^j) \in \mathscr{N}$, connecting that node to each terminal representing the re-allocation of the resources initially at terminal i to terminal $j \in \mathscr{N}^{\text{PH}}$.

The corresponding cost of repositioning a unit of resource $r \in \mathscr{R}$ from terminal $i' \in \mathscr{N}^{\text{PH}}$ to terminal $j \in \mathscr{N}^{\text{PH}}$ is noted $h^r_{i'j}$ (with $h^r_{i'j} = 0$ for $i' = j$).

The cycle definition is extended over the RAM layer to capture the acquisition and re-allocation activities within the resource-routing decisions. Cycles are thus associated to nodes in $\mathscr{N}'$ and include the arcs of $\mathscr{A}'$, yielding the set $\Theta^r_{i'}$ of potential cycles a resource of type r can execute out of each respective terminal.

The decision variables of the SSND, $y_\sigma, \sigma \in \Sigma$, and $x^k_a \geq 0, a \in \mathscr{A}, k \in \mathscr{K}$, are also defined for the SSND-RAM. Define the additional decision variables

$y^r_\sigma = 1$	if service $\sigma \in \Sigma$ is operated with a third party-owned resource $r \in \mathscr{R}$ and 0, otherwise;
$z^r_\theta = 1$	if cycle $\theta \in \Theta^r,\ r \in \mathscr{R}$, is selected and 0, otherwise;
w^r_i:	The number of new units of resource $r \in \mathscr{R}$ acquired and assigned to terminal $i \in \mathscr{N}^{\text{PH}}$;
$w^r_{i'j}$	The number of units of resource $r \in \mathscr{R}$ positioned from terminal $(i' \in \mathscr{N}^{\text{PH}}$ to terminal $j \in \mathscr{N}^{\text{PH}}$.

The *Scheduled Service Network Design with Resource Acquisition and Management* formulation for the single-leg-service case may be then written as follows:

$$\text{Minimize} \sum_{r\in\mathscr{R}} \left(\sum_{i\in\mathscr{N}} h^r_i w^r_i + \sum_{i'\in\mathscr{N}} \sum_{j\in\mathscr{N}} h^r_{i'j} w^r_{i'j} \right) + \tag{12.15}$$

$$+ \sum_{\sigma\in\Sigma} \left(f_\sigma y_\sigma + \sum_{r\in\mathscr{R}} f^r_\sigma \sum_{\theta\in\Theta^r} \delta^\sigma_\theta z^r_\theta \right) + \sum_{\sigma\in\Sigma} \sum_{r\in\mathscr{R}} F^r_\sigma y^r_\sigma$$

$$+ \sum_{r\in\mathscr{R}} \sum_{i\in\mathscr{N}} f^r_i \sum_{\theta\in\Theta^r} z^r_\theta + \sum_{k\in\mathscr{K}} \sum_{a\in\mathscr{A}} c^k_a x^k_a$$

Subject to

$$\sum_{i'\in\mathscr{N}'} w^r_{i'j} = I^r_i, \ \ \forall r \in \mathscr{R}, \ \forall (j, t^j_1) \in \mathscr{N}, \tag{12.16}$$

$$\sum_{\theta\in\Theta^r_{i'}} z^r_\theta \leq \sum_{(j,t^j_1)\in\mathscr{N}} h^r_{i'j}, \ \ \forall r \in \mathscr{R}, \ \forall i' \in \mathscr{N}', \tag{12.17}$$

$$\sum_{a\in\mathscr{A}^+_{(i,t^i_p)}} x^k_a - \sum_{a\in\mathscr{A}^-_{(i,t^i_p)}} x^k_a = d^k, \ \ \forall (i, t^i_p) \in \mathscr{N}, \ \forall k \in \mathscr{K}, \tag{12.18}$$

$$\sum_{k\in\mathscr{K}} x^k_a \leq \sum_{r\in\mathscr{R}} u(\sigma, r) \left(\sum_{\theta\in\Theta^r} \delta^\sigma_\theta z^r_\theta + y^r_\sigma \right), \ \ \forall a \in \mathscr{A}, \tag{12.19}$$

$$y_\sigma \leq \sum_{r \in \mathscr{R}} \sum_{\theta \in \Theta^r} \delta_\theta^\sigma z_\theta^r, \quad \forall \sigma \in \Sigma, \tag{12.20}$$

$$y_\sigma + y_\sigma^r \leq 1, \quad \forall \sigma \in \Sigma, \tag{12.21}$$

$$w_i^r,\ w_{i'j}^r \in \mathbb{Z}^+, \quad \forall r \in \mathscr{R},\ i \in \mathscr{N}', \tag{12.22}$$

$$z_\theta^r \in \{0, 1\}, \quad \forall r \in \mathscr{R},\ \forall \theta \in \Theta^r, \tag{12.23}$$

$$y_\sigma^r \in \{0, 1\}, \quad r \in \mathscr{R},\ \forall \sigma \in \Sigma, \tag{12.24}$$

$$x_a^k \geq 0, \quad \forall a \in \mathscr{A},\ \forall k \in \mathscr{K}. \tag{12.25}$$

The objective minimizes the total cost of the system. The first term models the cost of acquiring new and re-allocating existing resources. The second term computes the cost of selecting services and operating them with owned resources on particular cyclic routes. The third term models the costs incurred to secure third-party resources. The fourth term represents the costs associated with putting a resource into use, while the fifth and last term models shipment transportation costs.

Constraints (12.16) ensure that all resources of type r that are initially allocated to terminal i are either left at i or re-allocated. Constraints (12.17) link the strategic resource acquisition and allocation/re-allocation decisions that determine the number of resources available at each terminal with the tactical decision of how many resources from that terminal are to be used to execute services. Note the summation over $\mathscr{N}'$ in constraint s(12.17) enables the use of resources that are newly acquired.

Constraints (12.18) and (12.19) enforce classical network design relations. The former are commodity-specific flow conservation constraints. The latter link the existence of flow on owned or outsourced services to the corresponding service-selection decision. Constraints (12.20) indicate that at most one resources is used for each owned service, while constraints (12.21) specify that each service cannot be selected more than once, either supported by the carrier's resources or outsourced. Finally, constraints (12.22)–(12.25), define the domains of the variables in the formulation.

6 Managing Uncertainty

The SND and SSND are parameterized mathematical models of consolidation-based transportation systems. Using the methodology for planing and management purposes requires not only the model to accurately represent the system, but also the values of the model parameters to adequately predict the variations in the state of the system over the contemplated planning horizon. Of course, in reality, the validity of this assumption is rarely certain. Accounting explicitly for uncertainty in SND and SSND models aims to address this issue. An in-depth discussion of uncertainty and network design may be found in Chap. 9. We briefly recall the fundamental concepts in this section, focusing on their application to service network design.

In general, researchers have classified uncertainty into one of three types based upon their likelihood and impact. The first type, *randomness*, refers to events whose likelihood can be described and is reasonably high, but whose impacts can usually be mitigated within normal operations. The classic example of such uncertainty in SND contexts is fluctuations in the shipment volume between a given origin and destination. The second type, *hazards*, refers to events whose likelihood can be described, but are quite rare. An example in SND contexts is vehicle failure. The third type, *deep uncertainty*, refers to events whose likelihood can not be described and is extremely impactful. An example in SND contexts is a maritime port closing down due to a threat of terrorist attack.

Much (if not all) of the research on SND problems has focused on the first type of uncertainty, randomness, and specifically uncertainty with respect to model parameter values. This uncertainty is modeled by extending one of these deterministic models to a two-stage stochastic program . Such an optimization model presumes that some decisions must be made and implemented at a time when information regarding instance parameter values is incomplete. Specifically, that some decisions must be made at a time when only statistical distributions are known for the values of some parameters. In the context of a two-stage stochastic program, these decisions are referred to as *first stage* decisions. Then, at some point after the first stage decisions are implemented, the realizations of the uncertain parameter values is revealed. At that point, the remaining decisions can be made, in light of both the realized parameter values and the first stage decisions. These remaining decisions are often referred to as *second stage*, or, *recourse* decisions. As the second stage decisions are functions of random variables, they are random variables as well. Thus, the objective of such a model is to minimize the sum of the costs associated with the first stage decisions and the expected costs associated with second stage decisions.

In the context of service network design, most stochastic models prescribe the selection of services in the first stage and the routing of commodities, given those services and the realized parameter values, in the second stage. It is important to note that with two-stage stochastic programs in general, as well as those for service network design, the presumption behind these models is that from a practical planning perspective only the first stage decisions must be determined. The second stage decisions are not expected to be implemented. They may be used as guidelines (e.g., the itineraries and terminals for the main demand flows) when repeatedly applying the plan during the planning horizon. They primarily serve, however,as a means of approximating the impact of the first stage decisions on the performance of the system over the planning horizon. Specifically, the second stage approximates the expected cost of transporting demand loads given a network design.

To that effect, much of the research involving stochastic service network design models includes in the second stage the option to *outsource* all, or a part of, the delivery of a commodity from its origin to its destination, wherein the cost of outsourcing is proportional to the amount of the commodity's demand that is outsourced. Outsourcing may mean calling on an external service provider, or using an owned service which is not within the scope of the current problem and SND model. Thus, e.g., empty and loaded container-dedicated rail cars that cannot be accommodated on intermodal train services when the intermodal SND is being built, can be moved by general trains not in the scope of the planning problem. Then, by outsourcing a commodity, its delivery does not require the carrier designing a transportation plan to execute transportation services. Thus, in total, the stochastic SND formulation seeks to minimize the cost of executing services together with the expected cost of routing commodities and calling on external resources.

In addition, most research involving stochastic service network design models presumes that the joint probability distribution for uncertain parameter values can be approximated with a finite set of scenarios, wherein each scenario contains a realization of each uncertain parameter value and has a probability of occurring. With these scenarios, the expectation in the objective function can be expressed as a linear function, and the stochastic program can be formulated as a deterministic mixed integer program. In this section, we first focus on what types of stochastic programs have received the most attention for the SND. Namely, models that explicitly recognize uncertainty in shipment volumes due to randomness. We then discuss other potential sources and types of uncertainties that can be modeled.

6.1 Uncertainty in Shipment Volumes

The most commonly modeled source of uncertainty is demand. This is in part because it is the most prevalent in practice. Fundamentally, the SND and SSND presume that the size of a commodity is known and constant over the planning horizon during which the transportation plan prescribed by the model is implemented. In many logistics settings, a commodity models the orders of some customer (or the aggregation of multiple customers' orders). Thus, one source of uncertainty in commodity demand is due to variation in customer orders during that horizon. Another source of uncertainty has to do with the actual amount of vehicle capacity required by a customer order. The commodity demand value derived from a customer order is often just an estimate that is based on physical dimensions that are communicated by the customer to the transportation carrier. Thus, the actual amount of vehicle capacity required by an order may not be known with certainty until the order is picked up. We next present a stochastic programming variant of the SND above that is based on the premise that there is uncertainty in shipment volumes.

Uncertainty in shipment volumes is typically represented by treating the demand quantities, d^k, as random variables. A joint probability distribution for those random variables is presumed known, and is represented with a finite set of scenarios, $\mathscr{S}$. Each scenario $s \in \mathscr{S}$ represents a realization of the values, d^{ks}, of each of the random variables d^k. In addition, associated with scenario s is a probability, p_s, that it occurs.

The service network design under uncertainty (SND-U) problem is typically formulated on the same network, $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, as the SND and considers the same set of services, Σ. As service selection is determined in the first stage, before demand information is completely known, the SND-U involves the same y variables, $y_\sigma, \sigma \in \Sigma$, as the SND. Like the SND, the domains of these y_σ variables are usually either binary or integer numbers.

The SND-U models that commodity routing decisions occur after demand information has been fully revealed and design decisions are made. As a result, these decisions may depend on the scenario observed, and are modeled by indexing commodity flow variables by scenario, x_a^{ks}. Like the service selection variables, y_σ, these variables have the same possible domains as in the SND. Thus, the SND-U seeks to

$$\text{Minimize} \sum_{\sigma \in \Sigma} f_\sigma y_\sigma + \sum_{s \in \mathcal{S}} p_s \sum_{k \in \mathcal{K}} \sum_{a \in \mathcal{A}} c_a^k x_a^{ks} \tag{12.26}$$

Subject to

$$\sum_{a \in \mathcal{A}_i^+} x_a^{ks} - \sum_{a \in \mathcal{A}_i^-} x_a^{ks} = \begin{cases} d^{ks}, \text{ if } i = O(k), \\ -d^{ks}, \text{ if } i = D(k), \quad \forall i \in \mathcal{N}, \forall k \in \mathcal{K}, \forall s \in \mathcal{S}, \\ 0, \text{ otherwise}, \end{cases} \tag{12.27}$$

$$\sum_{k \in \mathcal{K}} x_a^{ks} \le u_a y_{\sigma_a}, \quad \forall a \in \mathcal{A}, s \in \mathcal{S}, \tag{12.28}$$

$$y_\sigma \in \mathbb{Z}_+, \quad \forall \sigma \in \Sigma. \tag{12.29}$$

$$x_a^{ks} \ge 0, \quad \forall a \in \mathcal{A}, \forall k \in \mathcal{K}, \forall s \in \mathcal{S}. \tag{12.30}$$

The objective of the SND-U seeks to minimize the cost associated with executing services along with the expected cost of routing commodities given the services selected. As the SND-U models commodity routing decisions that vary by scenario, constraints (12.27)–(12.30) enforce the same logical conditions as constraints (12.2)–(12.5) of the SND, albeit with a set of constraints for each scenario and demands that depend on the scenario. However, note that the right-hand side of constraints (12.28) represents that the same design is used to route commodities in each scenario.

As noted above, the SND-U is sometimes formulated under the assumption that the transportation of a commodity from its origin to its destination may be outsourced, and at a cost that is proportional to the amount outsourced. This is often modeled by adding the arc $(O(k), D(k))$ to $\mathcal{A}$ for each $k \in \mathcal{K}$. For these arcs, the cost c_a^k represents the outsourcing costs. As the transportation options modeled by these arcs do not involve a service executed by the carrier, constraints (12.28) are not formulated for such arcs.

6.2 Other Uncertainties in SND

We next discuss models that recognize other uncertainties that can be present in service network design. However, we note that many of these models have received little academic attention and some none at all. On the supply side, there can be uncertainty regarding the capacity to route commodities provided by a service that first stage decisions indicate should be executed. In practice, there are two sources for this uncertainty. In the first, unforeseen events (i.e., hazards) such as equipment failures can prevent the execution of a service that the first stage decisions prescribe. Thus, the capacity of the service effectively becomes zero. The second is similar, in that the capacity is different from what was anticipated in the first stage of the model, but less dramatic. Such uncertainty can occur, e.g., when a service is executed by a third party transportation carrier and the capacity provided by that service is shared with other carriers. As a result, when other carriers use more capacity than anticipated, the capacity available to the organization solving the SND is reduced. Such a drop in capacity may occur even with owned resources, such as partial equipment failure, e.g., cars on trains or compartments on liner ships.

Both sources can be modeled by treating the quantities u_a as random variables. However, the distributions used to model the two different sources are likely different. Regardless, given a set of scenarios to approximate the joint distribution of arc capacities (and potentially other random variables such as commodity demands), a SND-U similar to the one presented above can be formulated wherein u_a^s represents the capacity of arc a in scenario s. Then, the right-hand-side of constraints (12.28) is replaced with the term $u_a^s y_{\sigma_a}$.

There can also be uncertainty related to the costs incurred, either when routing a commodity or executing a service. Regarding routing a commodity, the SND may model the opportunity to use services that are executed by a third-party carrier that charges on a per-unit-of-demand basis (e.g., per pallet). In such a situation, there may be variability in the variable costs due to market forces. Modeling such variability can be done by treating the quantities c_a as random variables, which can be easily done as the variables associated with those cost coefficients are already in the second stage. By again presuming a set of scenarios representing the joint distribution of random variables, and c_a^s representing the variable cost on arc a in scenario s, a SND-U similar to the one above can be formulated, albeit with a slightly modified second term in the objective.

Regarding executing a service, as the associated cost is generally a function of transportation, variability from what was estimated can be driven by variability in the resources needed for transportation (e.g., fuel). Alternately, when a service is executed by a third party that provides transportation services to multiple customers, but charges on a per-service basis, variability may be driven by market forces. Such variability can be modeled by treating the costs f_{ij} as random variables. As these coefficients are associated with first stage decisions, calculating the total, expected fixed cost is not as straightforward as treating the variable costs c_a as random variables. No known research considers models that recognize this source of uncertainty.

Finally, and specific to the SSND, there may be uncertainty in the timings of activities. For example, there may be uncertainty related to the time, e_k, at which a commodity is available or to the time, l_k, at which it is due for delivery. In constraints (12.7), the SSND presumes these times are known with certainty, when in fact both may vary from what is expected. Issues with a manufacturing process may mean that goods to be transported are not always available by the time e_k. Alternately, a customer may sometimes need to rush an order, requiring the goods to be delivered before the time l_k. The SSND can be easily extended to a stochastic program that models both these uncertainties. Specifically, the right-hand-side values of constraints (12.7) can be modeled as random variables, d_{it}, with a distribution that is approximated by scenario. Then, in each scenario s, there must be a single t such that the $d^s_{O(k)t} = d^k$, a single t' such that $d^s_{D(k)t'} = -d^k$, and for all other i, t'', $d^s_{it''} = 0$.

Lastly, there may be uncertainty in the departure and arrival times of services. Note that time-dependent service travel times may be accommodated in the construction of the time-expanded network, $\mathscr{G}_{\mathscr{T}}$. Variability in service departure and arrival times may occur due to traffic congestion, weather conditions, or unforeseen events in terminal operations. Recently, there has been research that seeks to design transportation networks that meet a "service quality" target, wherein service quality refers to the probability of a commodity reaching its destination on time.

7 Bibliographical Notes

There is a broad and extensive literature on the Service Network Design problem. General surveys of the literature can be found in Crainic and Laporte (1997); Crainic (2000, 2003) and Wieberneit (2008). There are also surveys that focus on the use of SND models in specific contexts. Examples include intermodal freight transportation (Crainic and Kim 2007; Bektaş and Crainic 2008), City Logistics (Bektaş et al. 2017), and several chapters of this book. In the remainder of this section we review some of the most significant contributions to the literature. As this chapter was focused primarily on modeling up to now, we pay particular attention to solution approaches. Many ideas proposed for more general network design problems have been successfully adapted or applied to service network design problems. However, we focus our discussion on ideas that were primarily proposed in the context of service network design.

Some of the earliest work, both in terms of modeling and algorithmic development, can be found in Crainic et al. (1984); Crainic and Rousseau (1986) and Crainic and Roy (1988). The static path-based SND formulation minimizes a non-linear generalized objective function combining operating and time-related costs for services and shipments, as well as penalty costs for non compliance with service targets (e.g., market-specific delivery times) or the capacity limitations of terminals and services. The latter are cast as quadratic functions of the excess flow or duration. Moreover, the duration of terminal activities is modeled through convex approximations of average (and standard deviation) delays derived from queuing models accounting for the capacity and operation characteristics of the terminal. A similar approach is used for inter-terminal travel times when vehicles are captive of the infrastructure (e.g., rail and barges) or congestion phenomena are considered.

Many of the early solution methods proposed for SND problems are iterative local improvement heuristics. Examples of such methods can be found in the previous papers as well as in Powell (1986); Farvolden and Powell (1994). These methods search for an improving solution at an iteration by first adding or dropping services from the current network, and then routing the flows on the resulting network. Adding/dropping services from a network in the context of searching for an improving solution continues to be an effective algorithm strategy (Pedersen et al. 2009).

Kim et al. (1999) study a service network design problem in the context of express package delivery via a transportation network that connects ground and air movements. They leverage the structure of this transportation network and the nature of potential vehicle routes to derive a reduced time-space network on which they formulate an integer programming based on service and package flow route variables. Due in part to the scale of the delivery operation they solve this reduced formulation with column and row generation techniques. This exact, integer programming-based method is also used as the basis of a computationally effective heuristic. Armacost et al. (2002) study a similar problem and also approach the problem with integer programming methodology. However, motivated in part by the notoriously weak linear programming relaxations of service network design problems, they propose a formulation that does not model package flows directly. Instead, they propose a formulation based solely on design variables that represent aircraft routes, and show that with the right constraint set such a formulation can ensure sufficient capacity to transport all package demands, even though those demands are not explicitly modeled. They further strengthen the formulation by defining a specific type of design variable called a *composite variable*, which encodes the selection of multiple aircraft routes.

Jarrah et al. (2009) studies the service network design problem in the context of the less-than-truckload freight transportation industry. They leverage the single-path per shipment policy desired by carriers to propose a new formulation to the problem. Specifically, because the paths for shipments destined for the same terminal must induce a directed in-tree rooted at that terminal, the problem can be formulated with variables that represent flows on such trees. The proposed solution approach generates destination in-trees in a column generation-fashion in the context of a heuristic scheme. This in-tree structure was also exploited in Erera et al. (2013) in the context of a matheuristic scheme which at each iteration chooses a destination terminal and then solved an integer program to route freight destined for that terminal, holding fixed the routes for freight destined for other terminals.

Crainic et al. (1984); Crainic and Rousseau (1986); Crainic and Roy (1988); Powell (1986); Armacost et al. (2002); Jarrah et al. (2009), and Erera et al. (2013), to name but a few, consider models wherein the need to reposition empty vehicles is explicitly modeled. This has also been more generally referred to as *asset* *management* or *design-balance* (Andersen et al. 2009b,a; Pedersen et al. 2009; Chouman and Crainic 2015; Vu et al. 2013). Generally speaking, these models seek to ensure that the number of services that arrive at a node in a network equal the number of services that depart. This requirement introduces a challenge to linear programming-based heuristics for the SND as rounding up a fractional solution to the linear programming relaxation is no longer guaranteed to yield a feasible solution to the original problem. However, it also induces a structure to solutions. Specifically, that a design can be decomposed into cycles. Andersen et al. (2011) exploit this structure in a branch-and-price-based scheme for the problem wherein vehicles flow on cycles and commodities flow on paths, with both cycles and paths generated dynamically via column generation.

Many of the earliest service network design models do not consider assets at all. They seek to ensure there is sufficient capacity dispatched to transport shipments. Models that incorporate asset management constraints recognize that resources are needed to transport shipments and thus may have to move empty to be positioned for future moves. However, these models do not recognize that there may be a fixed fleet of resources, or that resources may need to periodically return to a specific "home" terminal. These types of issues are studied in models that incorporate *resource management* considerations (Crainic et al. 2014, 2018; Hewitt et al. 2019). The solution methods proposed in these papers combine a column generation scheme for generating resource cycles with another scheme for choosing cycles and routing shipments given the capacity created by those cycles. Crainic et al. (2018); Hewitt et al. (2019) also consider resource acquisition, allocation, and re-allocation decisions. Unlike the papers discussed so far, Hewitt et al. (2019) considers a model that explicitly recognizes uncertainty. Specifically, shipment volumes are presumed to be uncertain and resource and service network design decisions are made before complete demand information is known.

SND and SSND problems that recognize uncertainty have been studied. Lium et al. (2007, 2009) analyze the value of recognizing uncertainty in such models as well as how doing so leads to different structures in solutions. Both papers consider models that recognize uncertainty in shipment volumes, with the first focusing specifically on situations where there are correlations between those volumes. Turning to algorithms for such problems, Hoff et al. (2010) proposes a metaheuristic for a SND problem wherein there is uncertainty in shipment volumes. Wang et al. (2019) consider a different algorithmic approach to stochastic service network design problems and instead focus on the potential of creating a solution to a stochastic SND from a solution to its deterministic counterpart.

Stochastic programming-based approaches to SND that recognize uncertainty typically prescribe design decisions in the first stage and shipment flow decisions in the second. Thus, most of these models presume that the design remains unchanged after demands are revealed. Some (e.g., Crainic et al. 2016; Hewitt et al. 2019) model the opportunity to slightly adjust, in departure times, for example, or augment the chosen design after demands are revealed. Bai et al. (2014) model the opportunity to instead change the design (albeit at a penalty). Lastly, we note that while the vast majority of stochastic service network design models presume that statistical distributions exist for demands, robust optimization-based approaches have recently been proposed (Wang and Qi 2019, 2020).

Other sources of uncertainty have received attention as well. Specifically, Lanza et al. (2018, 2021) study a model wherein there is uncertainty in travel times. Such uncertainty introduces an additional component to the objective of the model that measures quality of service. Namely, the objective incorporates penalty factors based upon the total expected lateness of services and shipments. Demir et al. (2016) also study a model that recognizes uncertainty in travel times. Instead of general SND, they focus on intermodal transportation wherein fluctuations in travel times can interfere with the need to synchronize different transportation modes. One source of variability in travel times is the potential for vehicles and shipments to be delayed at a terminal. Estimating the lengths of these delays has received some attention (Crainic and Gendreau 1986).

Most SNDs consider a single level of consolidation. Namely, the consolidation of shipments into a container that is transported by a vehicle. However, some modes of transportation (e.g., rail, sea liners, and intermodal barges) necessitate multiple levels of consolidation as a vehicle may transport many containers, vehicles may be grouped into so-called *blocks* or *convoys* (rail and barge trains), or both. Such multi-layer models are considered in Kazemzadeh et al. (2019) and Zhu et al. (2014). We defer a deeper discussion of this topic to Chap. 12 of this book that focuses on rail network design. However, we note that a similar phenomenon has been considered in papers on SND that model motor-carrier platooning for long-haul movements (Albinski et al. 2020), and autonomous vehicles that can only travel autonomously in certain geographic regions (Scherr et al. 2018, 2019). The autonomous vehicles instead have to be pulled (called *platooning*) by a manned vehicle to such regions wherein they can then operate autonomously.

One of the computational challenges associated with solving instances of SSND models inspired by real-world operations is that the time-space networks on which these instances are based end up being very large. As a result, the numbers of variables that model shipments and vehicles moving through that network in those instances are very large as well, leaving mathematical programs that are too large to be solved in reasonable run-times. The network reduction techniques proposed in Kim et al. (1999) leveraged the specifics of that logistics context in an attempt to mitigate this issue. However, the size of these networks is due in part to the enumerative nature in which they are created and the process by which they are used.

First, the node set of such a network is created by enumerating each physical location at every time point when operations can occur at that location. Second, one portion of the arc set is created by enumerating each physical transportation move (the service) at every time point when it can depart. The other portion of the arc set consists of arcs that connect two nodes that represent the same location at different time points. Then, the instance is formulated on this network and solved. In such a static approach, much of the network that is created may not be needed by high-quality solutions.

Motivated by this observation, Boland et al. (2017) propose a different algorithmic strategy, named *Dynamic Discretization Discovery* (DDD), for using time-space networks in the context of SSND models. Specifically, they propose an iterative approach that begins with a time-space network wherein each location is represented at a small subset of the time points wherein operations may occur. Similarly, each physical transportation move is represented at a small subset of the time points at which it may depart. Boland et al. (2017) refer to such a network as a *partially time-expanded network* and formulate it in such a way that a SSND formulated on such a network is a relaxation of the SSND formulated on the time-space network derived from complete enumeration. To ensure that it is a relaxation given that not all locations are represented at all potential times, the network may need to contain arcs that underestimate actual travel times.

Thus, at an iteration of DDD, a SSND is solved on the current partially time-expanded network and the solution is examined to see if it can be converted to an optimal solution to the SSND formulated on a time-space network derived from complete enumeration. If it can be converted, the algorithm stops. If it can not, the current partially time-expanded network is refined and the algorithm continues. While Boland et al. (2017) present DDD for general SSNDs, it has also been adapted to other SSND-related problems. Medina et al. (2019) and He et al. (2019) propose adaptations of the algorithm to SSND problems that also determine local delivery routes. Hewitt (2019) proposes speed-up techniques for DDD when used to solve instances of SSNDs inspired by the less-than-truckload freight transportation industry. Marshall et al. (2021) propose a variant of DDD based on a differently-formed partially time-expanded network.

Another computational challenge that is often encountered when solving either SNDs or SSNDs inspired by real-world operations is that the number of shipments to be transported can be very large. This in turn can yield a large number of shipment flow variables and a mathematical program that is too large to be solved in reasonable run-times. One way to mitigate this issue is by defining shipment flows on paths instead of arcs (e.g., Crainic and Rousseau 1986; Crainic et al. 2009; Andersen et al. 2011; Hewitt et al. 2019). The downside to this approach is that it typically necessitates a scheme for dynamically generating paths as there are usually far too many to enumerate a priori. Another approach is a Benders decomposition-based method, wherein design decisions are made by a master problem based on estimates of the resulting shipment routing costs. These estimates are reflected in a constraint set present in the master problem that is iteratively added to as new designs are discovered. The downside of this type of approach is that these estimates are typically very poor in the early stages of the algorithm. As a result, Belieres et al. (2020) propose strengthening the master problem with the need to route a single, aggregated, *super-product*. Fontaine et al. (2016, 2021) take advantage of the problem structure to propose a different Benders decomposition, which includes tailored partial-decomposition technique for deterministic mixed-integer linear-programming formulations and specialized valid inequalities. In this approach, the master problem selects the services to generate a lower bound, while the slave problem solves a multiple knapsack problem with precedence constraints.

8 Conclusions and Perspectives

The chapter presented an overview and synthesis of the main classes of Service Network Design models aimed at supporting decision-making in planning the activities and managing the resources of consolidation-based freight carriers and systems. Applications of SND models and associated solution methods to particular transportation modes and system organizations are described in many chapters of this book, notably Chaps. 12–17. The chapter focused rather on issues and model structures of general interest and relevance. We continue this approach in identifying a number of challenging research perspectives of importance for both service network design and its applications and the broader network design field.

Extending the scope of SND, and the related modeling challenges, makes up a first research field. The aim is (1) to enhance the representation power and relevance of our models and solution methods, and (2) to extend the applicability of SND methodology beyond planning, to the short-term adjustment of plans to today's or this week's environment in terms of demand, state of the physical network, weather, and so on and so forth. Identifying the relevant issues and modeling them adequately requires a significant research effort, similar to the ones evoked in the following.

We have discussed the integration of resource-management concerns in Sect. 5, but many challenges remain. Different services and resources have different requirements and limitation. Thus, for example, North American trains are generally long and heavy and require traction power which can be provided only by combining two or three engines. Several such combinations are possible and each engine type has particular operational, maintenance, and fleet-size characteristics. Human resources also come with particular qualifications and work rules, including limitations on working hours and the types of vehicle individuals are authorized to operate. And, irrespective of mode and setting, transportation services require resources of several types, governed by particular compatibility rules to operate. Integrating the management of several heterogeneous interlinked resources into SND and SSND formulations challenges modeling and solution-method development alike.

Similar challenges characterize a better, more refined representation of terminal activities within tactical and strategic-level formulations. Most contributions so far model terminals through "simple" single node or arc representations and associated unit cost measures. Global capacity and unit time-related measures are appended to the node or arc representation in some cases. Yet, most terminals are complex infrastructures performing several operations on vehicles, power units, and loads in various parallel or sequential activity and waiting/queuing combinations. Average node or arc measures per unit of flow or service do not adequately represent this complexity. Obviously, one cannot integrate into a network-wide tactical or strategic model the full detailed representation of an operating terminal through, e.g., a network of queues. A few authors explored more detailed terminal representations replacing the node or arc with a small network capturing the main activities and waiting times of the terminal (e.g., Andersen et al. 2009b; Pedersen and Crainic 2007). These models provided more accurate estimations of the performance of the terminal, in terms of time and cost.

Explicitly addressing time and delay-related issues enlarges and refines the scope of SND models while raising significant modeling and algorithmic challenges. Consider, for example, the congestion one frequently observes in terminals and the resulting delays to vehicles and freight, which have to, first, enter the terminal and, then, go through the sequence of operations. These congestion conditions and delays are the result of high volumes of vehicles and freight "competing" for the terminal resources, i.e., its "capacity", within more or less the same time interval. As briefly mentioned above, the first challenge is to adequately represent these delays in terms both of model representativity and algorithmic efficiency. Working with a more refined terminal representation is part of the response to this challenge. Then, there is the issue of approximating the delays with linear or non-linear, ideally convex, functions. The former makes for an easier algorithm development, while the latter offers a more refined and adequate representation.

Adopting a non-linear formulation provides opportunities for modeling a broader range of issues and criteria compared to linear formulations with fixed capacities. Two cases to illustrate the point. First, representing infrastructure or service capacity in tactical-planning models through traditional constraints ignores the flexibility of translating plans into daily operations and of adjusting the former to the reality of the second. Thus, depending on the mode and carrier, extra freight one cannot load on a service either waits for the next departure or forces the dispatch of an extra vehicle. Both actions come at a cost best represented through a non-linear penalty, which may increase with the volume waiting, the length of the delay, etc. Consider, second, the quality targets carriers set and often publicize, e.g., A to B in X days. One may refine the selection of services and freight itineraries by representing potential deviations from target in the SND objective function. Non-linear penalties accounting for deviations from schedule for services and from due dates for commodity paths model such situations and guide the SND solution method. Standard deviations of activity, waiting, and travel times may also be included in computing the penalties, as well as the generally unpublicized percentage of error in attaining the targets the carrier allows for itself. Research is needed into SND formulations with non-linear objective functions and the associated solution methods (e.g., Bektaş et al. 2010).

Addressing uncertainty in SND models and methods for transportation and logistics planning constitutes a broad and important research area, challenging modeling and algorithmic development alike. As discussed in Sects. 6 and 7, SND models and solution models have already been proposed to address a number of uncertainty-related issues. One may state, however, that research in this area is still in its infancy. Research is still required in adequately representing demand uncertainty in the various problem settings evoked in this chapter and the other chapters of the book. Almost totally overlooked, although of great operational and economic importance, is the uncertainty in travel and terminal-activity times. The solution often adopted in practice of adding large buffers to the planned delivery times is not only scientifically unsatisfactory, but also less and less economically viable and impracticable in many cases (City Logistics to name but one example). Moreover, one should not overlook that both demand and time uncertainty (and heavy correlations) characterize operations and their simultaneous presence, and interactions, should be reflected in the planning models proposed. Research on this challenging topic is needed.

The previous issues, the discussion and the model of Sect. 6 refer to what is known as business-as-usual cases, when uncertainty can be somewhat easily represented with probability distributions. Other sources of uncertainty exist, however, and should be studied. Reliability and robustness are two such issues, as is resilience, i.e., the capability to rebound following an incident, and the operation plans to perform the recovery and return to a desired state of system and operation behavior. Advancing in this direction would also lead to a broader exploration of information-revelation mechanisms and multi-stage formulations.

We complete this "modeling" discussion noticing that most SND models, including those discussed in this chapter, assume that the behavior of customers, that is, of demand, is known with respect to economic, e.g., tariffs, and service-level criteria. This is true even when uncertainty in these elements is explicitly represented. Simply put, customers react to tariffs and quality-of-service levels and, consequently, so is the demand the carrier will ultimately service and the revenues it can potentially earn. Extending the SND to address such issues requires considering not only a profit-maximizing objective, but also modeling in mathematical terms the behavioral relations between tariffs, service-quality levels, and the willingness of customers to give a carrier their business. The revenue management literature is the starting point of this line of research noticing, however, that most of it targets people-servicing industries and that one cannot simply transpose those results to the freight transport environment (Bilegan et al. 2021). Bi-level SND programming, modeling the interactions between the carrier setting of tariffs and service levels and its self-interested customers, appears promising for a strategic-type of decision making on service level and pricing. Equally promising are the developments related to aggregated but accurate customer-behavior representations that could be integrated into carrier system-optimization SND models. An initial approach could define the response, through weights or probabilities, of customer types (e.g., regular, occasional, ad-hoc) to the carrier's discriminative service and tariff classes. The approximations, and their consequence in terms of demand volumes and revenues, would then become part of deterministic or stochastic SND formulations.

Addressing large SND models, particularly when several layers of design decisions are present and scheduling is involved, makes up another important research area. A first research direction in this area concerns the Dynamic Discretization Discovery (DDD) approach, which has shown its value when applied to standard SSND settings. More research is required, however, to refine and accelerate the method. We also need to extend it to the cases involving particular scheduling rules and patterns for some or all services according to, e.g., operation practices, modes or geographic/administrative zones. Another important extension, including for the DDD, concerns the problem settings with several design layers as one encounters when management of resources is considered or when, as in the case of railroads where one consolidates freight into cars, cars into blocks, and blocks into trains, each with potentially different temporal characteristics.

A second algorithmic research direction focuses on the dynamic generation of services (paths), resource work assignments (cycles), blocks (paths), and demand-flow itineraries (paths). With a few exceptions for resource management, this area has received little attention so far. Recall that, as illustrated in the models of this chapter, services are selected out of a set of potential services; the same discussion is relevant for the other system components as well. Authors rarely elaborate on the construction of the potential set. Obviously, it may correspond to all possible services, of all possible types, at all possible time instances. Two main issues with building a priori such a set. First, it would be of dimensions one could not address in most practical cases. Moreover, many of the potential services would be totally useless. But, which ones? How to avoid "bad" ones? The research on crew scheduling has clearly demonstrated that ad-hoc rules are not appropriate even when based on a company's own policies. The second issue is that, in practice, such a set would mean that the complete service structure and schedule of the carrier is to be built from scratch each time. This is generally not the case. Indeed, the demand structure of the next season is not totally different in most cases from the one at the last similar (e.g., Summer or Winter) season. Carriers then aim to update their previous schedule to adapt it to changes in demand patterns without imposing dramatic changes to their customers. Part of the service network is thus more or less fixed and a set of potential services must be built to reflect the changes in demand. The same question as previously stated arises with respect to building such a set. Research efforts have thus to be dedicated to extending the column-generation methodology to the SND and SSND cases with simultaneous generation of several types of paths and cycles.

With respect to solution-method approaches, recall that network design problems are NP-Hard in most cases of interest, and service network design ones are not different. Consequently, heuristic-type solution methods must be constructed. Yet, the research in this area is still not sufficiently developed. Particularly promising, and challenging, are matheuristics combining exact and meta-heuristic solution principles, ideally coupled with parallel optimization strategies, such as the Integrative Cooperative Search (Crainic 2019).

The development of efficient solution methods for stochastic SND and SSND is particularly challenging, even for the two-stage formulations of business-as-usual demand uncertainty case, which has been studied the most. The adequate representation of the "future", through sets of scenarios for example, is one the aspects contributing to this challenge. It raises issues regarding, the required number of scenarios, the purpose of scenario generation (represent the solution space or the optimal-solution neighborhood?), and how to generate the scenarios to serve this purpose. These questions are even more challenging when correlations and uncertainty in several problem parameters are considered. A scenario-based representation generally yields deterministic formulations of very large dimensions, very challenging to address as discussed above. The contributions mentioned in Sect. 6 and Chap. 9 make up the starting point of what should be a significant research effort on exact and matheuristic solution methods for stochastic service network design.

References

Albinski, S., Crainic, T. G., & Minner, S. (2020). The day-before truck platooning planning problem and the value of autonomous driving. Publication CIRRELT-2020-04, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et les transports, Université de Montréal, Montréal, QC, Canada.

Andersen, J., Crainic, T. G., & Christiansen, M. (2009a). Service network design with asset management: formulations and comparative analyzes. *Transportation Research Part C: Emerging Technologies, 17*(2), 197–207.

Andersen, J., Crainic, T. G., & Christiansen, M. (2009b). Service network design with management and coordination of multiple fleets. *European Journal of Operational Research, 193*(2), 377–389.

Andersen, J., Christiansen, M., Crainic, T. G., & Grønhaug, R. (2011). Branch-and-price for service network design with asset management constraints. *Transportation Science, 46*(1), 33–49.

Armacost, A. P., Barnhart, C., & Ware, K. A. (2002). Composite variable formulations for express shipment service network design. *Transportation science, 36*(1), 1–20.

Bai, R., Wallace, S. W., Li, J., & Chong, A. Y. L. (2014). Stochastic service network design with rerouting. *Transportation Research Part B: Methodological, 60*, 50–65.

Bektaş, T., & Crainic, T. G. (2008). A brief overview of intermodal transportation. In G. D. Taylor (Ed.). *Logistics engineering handbook*, chap 28 (pp. 1–16). Boca Raton, FL: Taylor and Francis Group.

Bektaş, T., Chouman, M., & Crainic, T. G. (2010). Lagrangean-based decomposition algorithms for multicommodity network design with penalized constraintsm. *Networks, 55*(3), 272–280.

Bektaş, T., Crainic, T. G., & Van Woensel, T. (2017). From managing urban freight to smart city logistics networks. In K. Gakis, & P. Pardalos (Eds.), *Networks design and optimization for smart cities, series on computers and operations research* (Vol. 8, pp. 143–188). Singapore: World Scientific Publishing.

Belieres, S., Hewitt, M., Jozefowiez, N., Semet, F., & Van Woensel, T. (2020). A Benders decomposition-based approach for logistics service network design. *European Journal of Operational Research, 286*(2), 523–537.

Bilegan, I. C., Crainic, T. G., & Wang, Y. (2021). Scheduled service network design with revenue management considerations for intermodal barge transportation. Publication CIRRELT-2021-23, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Université de Montréal.

Boland, N., Hewitt, M., Marshall, L., & Savelsbergh, M. W. F. (2017). The continuous-time service network design problem. *Operations Research, 65*(5), 1303–1321.

Chouman, M., & Crainic, T. G. (2015). Cutting-plane matheuristic for service network design with design-balanced requirements. *Transportation Science, 49*(1), 99–113.

Crainic, T. G. (2000). Network design in freight transportation. *European Journal of Operational Research, 122*(2), 272–288.

Crainic, T. G. (2003). Long-haul freight transportation. In R. W. Hall (Ed.), *Handbook of transportation science* (2nd ed., pp. 451–516). Norwell, MA: Kluwer Academic Publishers.

Crainic, T. G. (2019). Parallel metaheuristics and cooperative search. In M. Gendreau, & J.-Y. Potvin (Eds.), *Handbook of metaheuristics* (3rd ed., pp. 419–451). Berlin: Springer

Crainic, T. G., & Gendreau, M. (1986). Approximate formulas for the computation of connection delays under capacity restrictions in rail freight transportation. In *Research for Tomorrow's Transport Requirements, Fourth World Conference on Transport Research* (Vol. 2, pp. 1142–1155). Vancouver.

Crainic, T. G., & Kim, K. H. (2007). Intermodal transportation. In C. Barnhart, & G. Laporte (Eds.), *Transportation, Handbooks in Operations Research and Management Science*, chap 8 (Vol. 14, pp. 467–537). Amsterdam: North-Holland.

Crainic, T. G., & Laporte, G. (1997). Planning models for freight Transportation. *European Journal of Operational Research, 97*(3), 409–438.

Crainic, T. G., & Rousseau, J. M. (1986). Multicommodity, multimode freight transportation: A general modeling and algorithmic framework for the service network design problem. *Transportation Research Part B: Methodological, 20*, 225–242.

Crainic, T. G., & Roy, J. (1988). O.R. tools for tactical freight transportation planning. *European Journal of Operational Research, 33*(3), 290–297.

Crainic, T. G., Ferland, J. A., & Rousseau, J. M. (1984). A tactical planning model for rail freight transportation. *Transportation Science, 18*(2), 165–184.

Crainic, T. G., Ricciardi, N., & Storchi, G. (2009). Models for evaluating and planning city logistics transportation systems. *Transportation Science, 43*(4), 432–454.

Crainic, T. G., Hewitt, M., Toulouse, M., & Vu, D. M. (2014). Service network design with resource constraints. *Transportation Science, 50*(4), 1380–1393.

Crainic, T. G., Errico, F., Rei, W., & Ricciardi, N. (2016). Modeling demand uncertainty in two-tier city logistics tactical planning. *Transportation Science, 50*(2), 559–578.

Crainic, T. G., Hewitt, M., Toulouse, M., & Vu, D. M. (2018). Scheduled service network design with resource acquisition and management. *EURO Journal on Transportation and Logistics, 7*(3):277–309

Demir, E., Burgholzer, W., Hrušovskỳ, M., Arıkan, E., Jammernegg, W., & Van Woensel, T. (2016). A green intermodal service network design problem with travel time uncertainty. *Transportation Research Part B: Methodological, 93*, 789–807.

Erera, A., Hewitt, M., Savelsbergh, M., & Zhang, Y. (2013). Improved load plan design through integer programming based local search. *Transportation Science, 47*(3), 412–427.

Farvolden, J. M., & Powell, W. B. (1994). Subgradient methods for the service network design problem. *Transportation Science, 28*(3), 256–272.

Fontaine, P., Crainic, T. G., Jabali, O., & Rei, W. (2016). The impact of combining inbound and outbound demand in city logistics systems. In *41st IEEE Annual Computer Software and Applications Conference (COMPSAC)* (Vol. 2, pp. 766–770). Piscataway, NJ: IEEE.

Fontaine, P., Crainic, T. G., Jabali, O., & Rei, W. (2021). Scheduled service network design with resource management for two-tier multimodal city logistics. *European Journal of Operational Research, 294*(2), 558–570.

He, Y., Péton, O., Lehuédé, F., Hewitt, M., Medina, J. (2019) A continuous-time service network design and routing problem. In Program ROADEF 2019. On-line at: roadef2019.univ-lehavre.fr/programme/ROADEF2019_submissions/ROADEF2019_paper_195.pdf

Hewitt, M. (2019). Enhanced dynamic discretization discovery for the continuous-time load plan design problem. *Transportation Science, 53*(6), 1731–1750.

Hewitt, M., Crainic, T. G., Nowak, M., & Rei, W. (2019). Scheduled service network design with resource acquisition and management under uncertainty. *Transportation Research Part B: Methodological 128*, 324–343.

Hoff, A., Lium, A. G., Løkketangen, A., & Crainic, T. G. (2010). A metaheuristic for stochastic service network design. *Journal of Heuristics, 16*(1), 653–679.

Jarrah, A. I., Johnson, E., & Neubert, L. C. (2009). Large-scale, less-than-truckload service network design. *Operations Research, 57*(3), 609–625.

Kazemzadeh, M. R. A., Crainic, T. G., & Gendron, B. (2019). A survey and taxonomy of multilayer network design. Publication CIRRELT-2019-11, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Université de Montréal, Montréal, QC, Canada.

Kim, D., Barnhart, C., Ware, K., & Reinhardt, G. (1999). Multimodal express package delivery: A service network design application. *Transportation Science 33*(4), 391–407.

Lanza, G., Crainic, T. G., Rei, W., & Ricciardi, N. (2018). A study on travel time stochasticity in service network design with quality targets. *Lecture Notes in Computer Science, 11184*, 401–416.

Lanza, G., Crainic, T. G., Rei, W., & Ricciardi, N. (2021). Service network design problem with quality targets and stochastic travel times. *European Journal of Operational Research, 288*(1), 30–46.

Lium, A. G., Crainic, T. G., & Wallace, S. W. (2007). Correlations in stochastic programming: A case from stochastic service network design. *Asia-Pacific Journal of Operational Research 24*(2), 161–179.

Lium, A. G., Crainic, T. G., & Wallace, S. W. (2009). A study of demand stochasticity in service network design. *Transportation Science, 43*(2), 144–157.

Marshall, L., Boland, N., Savelsbergh, M., & Hewitt, M. (2021). Interval-based dynamic discretization discovery for solving the continuous-time service network design problem. *Transportation Science 55*(1), 29–51.

Medina, J., Hewitt, M., Lehuédé, F., & Péton, O. (2019). Integrating long-haul and local transportation planning: The service network design and routing problem. *EURO Journal on Transportation and Logistics, 8*(2), 119–145.

Pedersen, M. B., & Crainic, T. G. (2007). Optimization of intermodal freight service schedules on train canals. Publication CIRRELT-2007-51, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Montréal, QC, Canada.

Pedersen, M. B., Crainic, T. G., & Madsen, O. B. G. (2009). Models and Tabu search metaheuristics for service network design with asset-balance requirements. *Transportation Science, 43*(2), 158–177.

Powell, W. B. (1986) A local improvement heuristic for the design of less-than-truckload motor carrier networks. *Transportation Science, 20*(4), 246–257.

Scherr, Y. O., Neumann-Saavedra, B. A., Hewitt, M., & Mattfeld, D. C. (2018) Service network design for same day delivery with mixed autonomous fleets. *Transportation Research Procedia, 30*, 23–32.

Scherr, Y. O., Saavedra, B. A. N., Hewitt, M., & Mattfeld, D. C. (2019). Service network design with mixed autonomous fleets. *Transportation Research Part E: Logistics and Transportation Review, 124*, 40–55.

Minh, V., Crainic, T., & Toulouse, M. (2013). A three-stage matheuristic for the capacitated multicommodity fixed-cost network design with design-balance constraints. *Journal of Heuristics, 19*, 757–795.

Wang, X., Crainic, T. G., & Wallace, S. W. (2019). Stochastic network design for planning scheduled transportation services: The value of deterministic solutions. *INFORMS Journal on Computing, 31*(1), 153–170.

Wang, Z., & Qi, M. (2019). Service network design considering multiple types of seervices. *Transportation Research Part E, 126*, 1–14.

Wang, Z., & Qi, M. (2020) Robust service network design under demand uncertainty. *Transportation Science, 54*(32), 676–689.

Wieberneit, N. (2008). Service network design for freight transportation: A review. *OR Spectrum 30*(1), 77–112.

Zhu, E., Crainic, T. G., & Gendreau, M. (2014) Scheduled service network design for freight rail transportations. *Operations Research, 62*(2), 383–400.

Chapter 13
Freight Railroad Service Network Design

Mervat Chouman and Teodor Gabriel Crainic

1 Introduction

Rail transportation supports our social and economic life, providing economically-priced, environmentally-friendly, and timely transportation services for people and freight at the urban, regional, national, and international levels. A freight train, cargo train, or goods train is a group of freight cars (US) or goods wagons (International Union of Railways) hauled by one or more locomotives on a railway infrastructure network, transporting cargo all or some of the way between the shipper and the consignee. Railroads move large quantities of products, bulk materials (e.g., grains. minerals, petroleum and chemical products), intermodal containers and trailers loaded on flat cars, general freight, or specialized freight (e.g., automobiles and heavy machinery) in purpose-designed cars. Railroads are particularly efficient for long-haul movements in terms of per ton-km monetary, energy-consumption, and pollutant-emission costs. They are faster and more direct than ocean freight, which lead to setting up transcontinental land bridges, e.g., the North-American Landbridge linking the West and East coasts, and the Eurasian Landbridge between China and Western Europe. We focus on freight rail transportation in this chapter, while passenger rail transportation is discussed in Chap. 17.

Freight rail transport makes up an essential link in intermodal transportation and supply chains, supporting national and international trade. The efficiency of railroads, in terms of cost and reliable on-time delivery, thus directly impacts the

M. Chouman
Effat University, Jeddah, Saudi Arabia
e-mail: mchuman@effatuniversity.edu.sa

T. G. Crainic (✉)
CIRRELT and AOTI, Université du Québec à Montréal, Montréal, QC, Canada
e-mail: TeodorGabriel.Crainic@cirrelt.net

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_13

availability and cost of goods for the final customer, be it a private citizen, an institution, or a company. To achieve this efficiency, railroads operate mostly as *consolidation*-based carriers, similarly to less-than-truckload trucking (Chap. 14), liner shipping (Chap. 15), and city logistics (Chap. 16), for example.

The fundamental idea of consolidation is to take advantage of economies of scale and reduced handling in terminals, by grouping loads from different shippers, with possibly different origins and destinations, and loading them into the same vehicles for efficient long-haul transportation. Railroads generally implement a more complex double consolidation policy, however, as cars are grouped into blocks, which are then grouped into trains. Thus, loaded and empty cars, with different origins and destinations, being present simultaneously in the same terminal, are sorted and grouped into a block, which is then moved as *a single unit* by a series of trains until its destination, where it is broken down, the cars being either delivered to their final consignees or sorted for inclusion into new blocks. The performance and profitability of such a system depend on an offer of services meeting the cost and quality criteria of its potential customers, but also, for a large part, on efficient and coordinated terminal and long-haul transport operations.

Tactical, medium-term, planning for freight rail carriers aims to address this challenge at the network and system-wide level, through a transportation plan specifying the train services to operate over the contemplated schedule length (e.g., the week), together with their frequencies or schedules (timetables), the blocks that will make up each train, the blocks to be built in each terminal, and the routing of the cars, empty and loaded with the customers' freight, using these services, blocks, and terminal operations. As detailed in Sect. 2, tactical planning makes up a very complex problem, with many facets and decisions linked in a web of economic, resource utilization, and time-performance objectives, limitations, and trade-offs. Operations Research provides the *Service Network Design* (*SND*) methodology to build the railroad tactical plan making the most efficient use of the railroad's resources to achieve its economic and customer-service performance objectives. The chapter reflects this important relation between railroad planning and network design. It focuses on SND models for railroad tactical planning, both for particular activities, e.g., car blocking and train makeup, and for integrated planning processes.

The chapter is organized as follows. Section 2 briefly describes the rail transportation system, the associated tactical planning issues and the utilization of tactical-planning SND models, and concluding with the general notation used in the chapter. Section 3 is dedicated to SND formulations, which do not integrate the time dimension explicitly, for three problem settings: service selection and train makeup (Sect. 3.1), car classification and blocking (Sect. 3.2), and integrated planning (Sect. 3.3). Section 4 focuses on the case where the time characteristics of the problem components and decisions are explicitly addressed, and introduces the *Scheduled Service Network Design* (*SSND*) problem and model for the integrated planning of freight railroads. The SSND modeling framework is extended in Sect. 5 to account for existing schedules, the container-to-car loading rules of intermodal traffic, and resource management. Bibliographical notes are presented in Sect. 6 and we conclude with a number of research directions in Sect. 7.

2 Rail Transportation System and Planning

We initiate the section with a brief description of freight rail transportation, with its main objectives, system components, operations, and decision and planning challenges. Tactical planning issues and their complex interactions are discussed next, introducing the *Service Network Design* (*SND*) methodology generally proposed to address them and which is the object of this chapter. We conclude with a discussion on the various utilization modes of SND models, and the general notation used throughout the chapter.

2.1 Rail Transportation System

Railroads are complex transportation systems where several major components interact and compete for resources. The infrastructure of the system is made up of a large number of terminals and rail tracks linking them. Most of these terminals are stations where demand originates and terminates. A much smaller number are denoted *yards* and are specially equipped to handle large quantities of cars, sorting and grouping them for long-haul transportation, as well as to make up and disassemble trains. The term *classification* (*marshaling* is also found) yard is used to emphasize the major car-handling role of these facilities. Terminals are linked by a physical network of tracks. The backbone component of this network is made up of main lines connecting the yards of the system. The network is completed by a large number of secondary, branch lines connecting most stations to the backbone network. Even when stations are located on a main line, the movements of loaded and empty cars between stations and their respective designated yards are generally performed by local, so-called feeder trains.

Customer *demand* takes the form of a number of cars (the special case of intermodal transportation is discussed later in this section), of a type appropriate to the commodity that needs to be moved, to be shipped from an origin station to a destination one. The appropriate number of empty cars is delivered for loading by the railroad to the customer site, assuming it is connected to the rail network, or to a designated station, otherwise. The empty cars are generally delivered from a designated yard by a feeder train. Once loaded, the cars are moved back to the same yard or to a different one as appropriate for the long-haul movement on the main-line network toward the destination. Since the scheduling of feeder trains is usually not within the scope of the network-wide tactical planning process designing the long-haul service network, we assume in this chapter that demands are defined among origin and destination yards. Each demand is also characterized by a *volume* in terms of number of loaded cars of given physical and operational attributes, as well as by an *availability time* (and date) at the origin yard and a *due time* at the destination yard.

Trade is unbalanced among countries and regions and, consequently, so is the demand for particular car types in the case of railroads. Moving empty cars is costly and railroads aim to minimize such balancing flows. Yet, they cannot be entirely avoided and, thus, empty cars are often part of train composition. These movements must be accounted for when planning services and resources, to avoid underestimating traffic, resource utilization, and costs. Origin-destination "empty-car" volumes are thus often part of the demand definition.

Movements of freight on the rail network are performed by *train services*. A train is composed of one or more locomotives providing power and a series of cars (which may be loaded or empty; sometimes, locomotives are repositioned in the network and are part of a train without providing power). Each train has a particular origin yard where it is made up and a destination yard where it completes its journey, delivers all the cars currently hauled, and liberates the locomotives. The route may encompass a number of intermediary stops where the train delivers or picks up cars, eventually grouped into blocks as described below (locomotives and crews may also be changed, added and dropped off, at intermediary yards). Other than its route, the train service or, simply, the *service*, is also characterized by time-related information. In its simplest version, this information takes the form of a *frequency of service*, i.e., the number of times the "same" train is run during the length of time the railroad uses to define its recurring operations (e.g., 1 week), also called *schedule length*. A more precise definition is given by a service schedule indicating the departure time from the origin yard, arrival and departure times at each intermediary yard, and the arrival time at destination. This information may be strict, as for most European, Canadian, and a few U.S. Class 1 railroads, or relative (e.g., most U.S. Class 1 railroads), indicating time intervals for their departures which may be modified to account for particular events, e.g., the need to pass a direct train for an important customer. (Note that, there are still railroads around the world with schedules of an "indicative" nature, the train leaving when full and ready.) The railroad may operate a single type of service, e.g., dedicated intermodal shuttle trains between main yards. Alternatively, services of different types, e.g., general cargo, bulk, intermodal, may be defined and operated, often on the same infrastructure. Priority with respect to the other service types (often linked to the speed and capacity allowed on each section of track) is often used to define the service type.

Railroads aim to maximize revenue, which often translates into achieving the best balance between the operational cost of operating resources and services, on the one hand, and the quality of the service according to the customer expectations in terms of tariffs, speed, flexibility, and reliability, on the other hand. Dedicated and direct non-stop services from origins to destinations (so-called "unit" trains) would achieve high customer satisfaction, reducing delivery time and the risk of delay (providing train congestion on rail tracks is avoided), and eliminating the risk of damage related to car handling at intermediate yards. This would also, however, imply high operational costs, particularly for the very large numbers of origin-destination demands with low and medium numbers of car to move. Railroads therefore operate direct trains only for particularly important customers or when

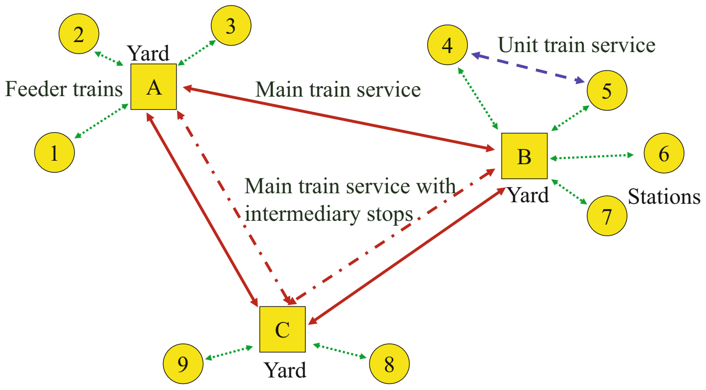


Fig. 13.1 Hub-and-spoke rail service network

the volume of demand between two stations is significant (i.e., the equivalent of at least a full train) and regular. Railroads rather aim for economies of scale for most of their operations through consolidation of freight from different demands, that is, cars with different origins and destinations, into blocks and blocks into trains.

Schematically, cars at their origin yard are sorted, *classified* is the term generally used, to be then grouped with other cars, with potentially different origins and destinations, into particular blocks. The *block* is then handled as a single unit from its origin yard, where it is formed, to its destination yard, where it is taken apart, its cars at their final destination being delivered to the respective consignees, the others being reclassified and blocked with other cars present at the yard for the next part of their trip. This classification and blocking operation contributes significantly to the economy-of-scale provided by rail transportation. Trains are thus made up of blocks and, when appropriate, it is blocks that are picked up and delivered at intermediate stops. Blocks may thus be *transferred* (*switched*) from one train to another.

Figure 13.1 illustrates this hub-and-spoke service organization for a network with three yards and nine stations. Dotted lines indicate feeder services moving cars and, eventually, blocks, between stations and main yards. The dash line illustrates a direct-train service between two stations, while the solid and dash-dotted lines represent the non-stop and one-intermediary-stop, respectively, long-haul train services moving on the main line network between yards.

Demand is moved along itineraries. Each *itinerary* for a particular demand specifies the sequence of blocks and trains, and thus the sequence of classification and transfer activities, between its origin and destination yards. The volume of freight of certain demands must be moved together, while for others, it can be split among several itineraries, as agreed between the railroad and the customer. From the railroad perspective, the possibility to split demand flows allows to better fill up blocks and trains, increasing the economies-of-scale, but requires additional care in monitoring the flows and making sure everything arrives in time to the final yard, for on-time delivery of the complete shipment. From an optimization perspective, the model must include integer-valued flow variables when demand cannot be split, which increases the algorithmic challenge. To simplify the presentation, we assume in this chapter that flows may be split for all demands.

Operations are constrained by the physical characteristics of the infrastructure and the operational policies of the railroad and, thus, "capacity" is a multi-facet concept in rail transport. Consider, for example that, the car classification capacity of a yard, for a given time period, may be defined in terms of the maximum numbers of cars that may be handled, blocks that may be built (number and length of tracks on which the blocks are composed), trains that may be made up or serviced, and so on. Similarly, the capacity of the rail tracks limits operations with respect to the number of trains that may operate "simultaneously" on a given track segment (meeting or overtaking), as well as to the total weight, length or both a train may haul on the track. The length and weight of trains are thus limited and translate into lower and upper limits on the length and weight of blocks. Representing the operational characteristics and limits at a level appropriate for the network-wide nature of system and planning is one of the challenges of developing Operations Research-based methods for railroad freight transportation.

We conclude this section with a short discussion of an important and growing component of rail transportation, namely, intermodal traffic and operations. In its general sense, intermodality means that different transport modes are combined to seamlessly move containerized freight from a point of origin to a point of destination. A sequence involving a truck or rail (or a sequence of both) movement to a port, ocean navigation to another port, and a truck or rail sequence to destination is typical of intermodal transport and makes up the backbone of international trade. Rail plays a major role in this context as illustrated by the European Commission policy on intermodality, the new rail services being set up between China and Europe, and the intermodal-rail divisions of North American railroads linking the continental ports to the industrial and heavily populated regions of the continent.

Intermodal traffic is often handled separately from the general one, being moved on dedicated intermodal trains (attaching intermodal traffic to regular main-line trains may be viewed as a recourse operation to mitigate variations in forecast demand). Moreover, even when intermodal and regular cars and trains are handled in the same yards, the classification of intermodal cars is performed separately.

The most important difference, however, concerns the loading and unloading operations of intermodal traffic, which is actually taking place in particular zones of the railroad's yards. There is a large variety of container types, e.g., 20-, 40- and 53-feet long, and railroads use fleets of cars of various types, each with one or several platforms and slots on the platforms. Single- and double-stack platforms have one and two slots, respectively. The containers are delivered at yards (or maritime port facilities) and the railroad must determine the matching/loading of containers to available cars and types. This is an important but complex issue since not all combinations are legal or suitable, a very large number of loading alternatives exist, and decisions taken at any given yard impact the availability of cars at later periods at the yard and the other yards, as well as the performance of the railroad operations.

The double consolidation organization of freight railroads provides the sought-after economies of scale in operation costs and resource utilization, reduces car handling activities at yards, and fosters a timely service for markets (origin-destination pairs of cities or regions) with low traffic volumes. It also implies more complex operations in terminals, with potentially higher possibilities for delays and incidents. Which translate into more complex decision-making problem settings. The complexity is even larger for intermodal transportation which implies a third consolidation operation, of containers on multi-platform cars.

Network design-based models and methods are proposed to address these challenges and support decision making at various levels of planning. We now briefly recall these planning issues and the links to network design.

2.2 Tactical Planning and Network Design

The planning activities undertaken by railroads may be broadly classified into three levels, similarly to most other consolidation-based transportation systems. Strategic planning involves long-term decisions on system design, operation strategies, and acquisition of major resources (e.g., buy or rent locomotives or cars and enhance track or yard capabilities). Tactical planning is dedicated to building an efficient service and resource-utilization network and schedule. Short term planning, monitoring, and adjustment of operations make up the so-called operational planning (e.g., running the trains, crew and locomotive scheduling, repositioning crews, locomotives and cars for the next operations, and maintenance of infrastructure and rolling stock). We focus on tactical planning in this chapter, as it involves arguably the strongest connection to network design. We discuss at the end of this section the utilization of the related network design methodology in varied contexts, including the other levels of planning. Section 6 points to general references addressing railroad challenging planning activities and problems at the three levels.

Tactical planning is performed over a medium-term planning horizon, e.g., 6 months, called *season* in the following. Planning generally takes place some time before the beginning of the season. It aims to select and schedule services, together with the demand itineraries used to move the freight from origins to destinations using the resulting service network. Determining strategies for managing important resources supporting the selected services, as well as activity profiles for terminals, in terms of car, block, and train-handling policy for example, is also increasingly part of tactical planning. The goal is to satisfy the forecast regular demand in the most efficient way possible with respect to costs (profits) and resource-utilization, while satisfying the service-quality levels set by the carrier to answer customer requirements. Notice that, even though some part of demand, e.g., long-term contracts with customers, may be known at planning time, most is forecast using history, customer-relation representatives knowledge, and customer input, among other data sources. The service network and plan is determined for a rather short schedule length and it is repeatedly applied over the season.

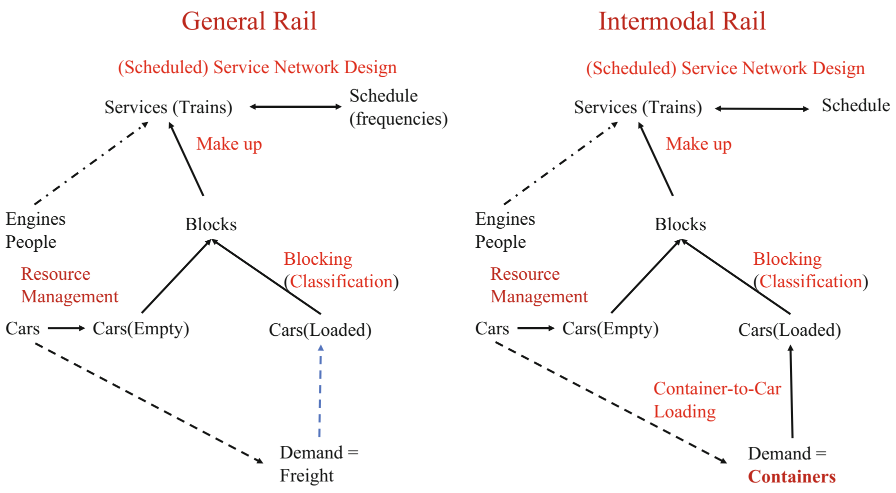


Fig. 13.2 Main activities and tactical planning decisions

A number of main decisions/issues make up the tactical planning process and are addressed through SND models and methods. These problems and their relations are briefly defined in this section and schematically illustrated in Fig. 13.2 for the general and the intermodal cases.

Service selection is concerned with choosing among a set of possible services the ones to operate the next season to service demand efficiently, profitably, and on time. The set of possibilities could represent a complete yard-to-yard network with intermediate stops for all service types, or the last-season network enriched with additional potential services to address changes in demand profile and railroad policies. The resulting service network specifies the movements through space and time of trains and cars, demand itineraries corresponding to paths in this network. The problem is defined as *static* when one assumes that neither demand nor the other problem characteristics vary during the schedule length considered. The time dimension of the service network is then implicitly considered through the definition of services and the inter-service operations at terminals, and one generally addresses the issue of *service frequency* assuming a more or less uniform distribution of departures over the schedule length. *Time-dependent* problem settings and formulations address the cases when the moments demands become available and are due at destinations are explicitly considered, which implies an explicit representation of demand and activities in time. Time-dependent formulations thus usually target the planning of *schedules* to support decisions related to *when* services and freight (demand itineraries) arrive at and leave from yards.

The *blocking* and *classification* problem addresses the issue of how cars are grouped in yards yielding the blocks to be moved by trains. It encompasses several strongly interrelated decisions: (1) select the blocks to build at each yard; (2) specify for each block its origin and destination terminals, the path through the infrastructure network, the sequence of intermediate yards on the path where it will be transferred from one train to another (when relevant), the sequence of train services performing the transportation; and (3) define how cars, empty and loaded, are classified and assigned to blocks at their origin and at the intermediary (when brought in by blocks being dismantled at destination) yards on their journeys. The decisions concerning the empty cars are generally linked to car-fleet management concerns about providing the appropriate cars to yards given the particular associated demand. Loaded-car assignment to blocks, on the other hand, is related to the freight-routing objective of delivering shipments efficiently and on time.

The problem includes an additional dimension for intermodal services, namely, the assignment and consolidation of containers of various types and dimensions to multi-platform cars of different types and dimensions. The first challenge is how to reflect the differentiation of the many types of containers, cars, and loading rules, and how to represent the container-to-car assignment and loading in a way appropriate for the level of aggregation proper of tactical planning models and solution methods. Second, the containers-to-car consolidation adds a third combinatorial dimension and design decision to the blocking problem, yielding different SND formulations harder to address than for the regular-traffic case. This difference in illustrated in Fig. 13.2 by including the demand loading component in the intermodal-rail box on the right of the figure.

Train makeup yields the list of blocks, and cars of particular origin-destination demands, each train service hauls out of its origin yard, it drops and picks up at intermediary stops, and delivers at its destination yard.

Freight routing determines the *itinerary*, or itineraries when splitting of demand is allowed, used to transport the cars of each particular demand from its origin to is destination through the selected service network.

Resources, e.g., locomotives and cars, are required to operate services. *Resource management* addresses the issue of, on the one hand, assigning the appropriate resources to services to support the planned activities while, on the other hand, determining the general rules dictating the economically and operationally-efficient resource movements over the schedule length. Resource management is generally considered an operational-level managerial activity and its impact on tactical-level decisions was often limited to the somewhat simple case of accounting for the need to reposition empty cars for the next cycle of operations. The situation is evolving, however, and more comprehensive problem settings are detailed later in the chapter.

These problems may be, and have often been, addressed individually echoing a tactical-planning process decomposed into a series of sequential decisions. Increasingly, however, the strong interconnections among decisions, in particular in their cost and service quality consequences, lead to integrated approaches addressing several of the problems identified above jointly. Such approaches do not make problems easier, however, as planning must be performed network-wide aiming for the best trade off among the not necessarily convergent operational and economic characteristics of the individual problems and decisions. Thus, for example, one could increase the level of service by increasing the frequency of services, but this could result in higher levels of congestion in yards and on the tracks, resulting in increased delays and, thus, lower quality service (increased costs as well, of course).

Service network design (SND) models are generally proposed to address freight railroad planning problems. SND models take the form of *fixed cost, capacitated, multicommodity network design* formulations. Minimization of the total operating costs is the primary optimization criterion in most cases, reflecting the traditional objectives of railroads and expectations of customers to "get at destination fast but at the lowest possible cost". As customer expectations for high-quality service and environmental concerns rise, however, service performance measures are increasingly included in tactical planning and SND formulations. Service performance measures are generally modeled through delays incurred by freight and resources or the amplitude of violation of predefined performance targets (e.g., delivery within a given time length). Constraints may then be imposed on the values of the service-performance measures, or one may add them as costs and penalties to the objective function of the SND optimization formulation. The resulting generalized cost function then captures the trade offs between operating costs and service quality. The sections that follow present the main classes of railroad service network design models proposed, in increasing degree of problem and decision integration.

We conclude this general presentation with a short discussion on the utilization of SND methodology developed for tactical planning. To start, notice that, although network design models may be built to address strategic-planning issues, SND formulations may be used to evaluate the impact of strategic scenarios, relative, for example, to economic (e.g., fuel prices or changing production and consumption levels of certain goods) and regulatory (trade restrictions or speed and weight limits when carrying hazardous goods) variations on operations, resources, and system performance. One may thus use SND as a simulation tool in the context of cost-benefit analyzes, with appropriate approximation of railroad and demand characteristics. Clearly, generalized service network design models may be built to answer strategic-level decisions such as the number of each resource type to buy or rent, and the capital-intensive enhancement of infrastructure.

The service network design formulations may also be used to review, weekly for example, the tactical plan built for the season. One would then re-optimize and adjust the plan and operations to current conditions. What may be adjusted depends strongly on the railroad application context. Canceling or adding services on a short notice is not easily performed by railroads and SND models may assist in selecting the best alternative and determining the network-wide impacts. Updating the actual demands or resources, or both, assigned to blocks and trains is taking place quite often within railroad management and, again, SND models are appropriate. Obviously, the scope of the SND model has to be more focused when in plan-adjustment mode, parts of the system which should not be modified being fixed.

2.3 Notation

This section is dedicated to the definitions and general notation used throughout the chapter. It is summed up in Table 13.1 (together with notation proper to particular problem settings and defined in the next sections).

Let $\mathscr{G}^{\mathrm{PH}} = (\mathscr{N}^{\mathrm{PH}}, \mathscr{A}^{\mathrm{PH}})$ represent the physical network on which the railroad operates, where $\mathscr{N}^{\mathrm{PH}}$ stands for the set of terminals (yards and, possibly, main stations), connected by the physical track arcs of set $\mathscr{A}^{\mathrm{PH}} = \{(\eta_i, \eta_j), \eta_i, \eta_j \in \mathscr{N}^{\mathrm{PH}}\}$.

Yards $\eta \in \mathscr{N}^{\mathrm{PH}}$ are characterized by several capacity measures, defined for a given time period (which can be the schedule length or shorter for multi-period, time-dependent formulations), namely, the *classification capacity* u^{C}_{η}, for the number of cars that can be sorted and assigned to blocks, the *blocking capacity* u^{B}_{η}, for the number of blocks which can be built, the *block-transferring capacity* u^{T}_{η}, for the number of block which may be transferred from one service to another, and u^{M}_{η}, for the number of trains which can be made up at the yard during the period.

Train services run on this network to answer demand. Following the general Service Network Design (Chap. 12) and rail-planning literature, SND models select these services out of a set of *potential services* $\varSigma$, given an estimated *regular demand* for transportation represented by set $\mathscr{K}$ of origin-destination (OD) commodities, each commodity $k \in \mathscr{K}$ standing for the request to move a quantity d^k of freight from its origin terminal $O(k)$ to its destination terminal $D(k)$.

Each service $\sigma \in \varSigma$ is characterized by a path in the physical network between its origin and destination yards, $O(\sigma)$ and $D(\sigma)$, respectively. *Single-leg* services operate non-stop between their respective origins and destinations, while *multi-leg* services stop at one or several yards on their routes to drop and pick up blocks and cars. Let $\mathscr{N}^{\mathrm{PH}}(\sigma) = \{O(\sigma) = \eta_0, \eta_1, \eta_2, .., \eta_{n-1}, D(\sigma) = \eta_{n(\sigma)}\}$ be the sequence of yards visited by service $\sigma \in \varSigma$, and $\mathscr{L}^{\mathrm{PH}}(\sigma) = \{l_i(\sigma) = (\eta_{i-1}, \eta_i) \mid i = 1, \ldots, n(\sigma)\}$ be the sequence of service legs of the service, with $n(\sigma) = 1$ for single-leg services. Let $\mathscr{L}^{\mathrm{PH}} = \bigcup_{\sigma \in \varSigma} \mathscr{L}^{\mathrm{PH}}(\sigma)$. Several "cost" and "capacity" measures may be associated to services depending on the particular problem addressed. In almost all cases, however, one finds the fixed cost to select the service, f_σ, the leg unit transportation costs $c^k_{l_i(\sigma)}$, and the leg-service capacity $u_{l_i(\sigma)}$.

Each of the SND tactical planning models described in the following sections is defined on a network $\mathscr{G} = (\mathscr{N}, \mathscr{A})$ built out of the physical network and the set of potential services, enriched to address problem-specific characteristics, the modeling of time, in particular. Static SND problem settings, discussed in Sect. 3, do not include an explicit representation of time and, thus, $\mathscr{G}$ has $\mathscr{N} = \mathscr{N}^{\mathrm{PH}}$ and $\mathscr{A} = \mathscr{L}^{\mathrm{PH}}$ in those cases.

Scheduled service network design, *SSND*, targets time-dependent problem settings, where time and service schedules are explicitly considered. SSND formulations are built on *time-space* networks $\mathscr{G} = (\mathscr{N}, \mathscr{A})$, using a discrete or continuous representation of the schedule length $\mathbf{T}$. Let t stand for a time instant within the schedule length, i.e., $0 \leq t \leq \mathbf{T}$. A discrete representation of the schedule length is

Table 13.1 SND and SSND main notation

$\mathscr{G}^{\mathrm{PH}} = (\mathscr{N}^{\mathrm{PH}}, \mathscr{A}^{\mathrm{PH}})$	Physical network
$\mathscr{N}^{\mathrm{PH}} = \{\eta\}$	Set of terminals (yards and main stations)
$\mathscr{A}^{\mathrm{PH}} = \{(\eta_i, \eta_j)\}$	Set of rail-track arcs
$\mathscr{G} = (\mathscr{N}, \mathscr{A})$	Potential (service) network for the SND formulations
$\mathscr{N}, \mathscr{A}$	Set of terminals and arcs in $\mathscr{G}$
$\mathscr{H} = \{(\eta_t, \eta_{t+1})\}$	Set of holding arcs
u_a	Capacity of arc $a \in \mathscr{A}$
$\Sigma = \{\sigma\}$	Set of potential services
$O(\sigma), D(\sigma)$	Origin and destination terminals of service $\sigma \in \Sigma$
$\mathscr{L}^{\mathrm{PH}}(\sigma) = \{l_i(\sigma)\}$	Set of legs of service $\sigma \in \Sigma$
f_σ	Fixed selection cost for service $\sigma \in \Sigma$
$c^k_{l_i(\sigma)}$	Unit transportation cost for commodity $k \in \mathscr{K}$ on leg $l_i(\sigma) \in \mathscr{L}^{\mathrm{PH}}(\sigma)$ of service $\sigma \in \Sigma$
$u_{l_i(\sigma)}$	Capacity of service $\sigma \in \Sigma$ on its leg $l_i(\sigma) \in \mathscr{L}^{\mathrm{PH}}(\sigma)$
$o(l_i(\sigma))$	Scheduled departure time of service $\sigma \in \Sigma$ from the origin of its leg $l_i(\sigma) \in \mathscr{L}^{\mathrm{PH}}(\sigma)$
$d(l_i(\sigma))$	Scheduled arrival time of service $\sigma \in \Sigma$ at the destination of its leg $l_i(\sigma) \in \mathscr{L}^{\mathrm{PH}}(\sigma)$
$\mathscr{K} = \{k\}$	Set of origin-destination demands
d^k	Quantity of demand $k \in \mathscr{K}$
$O(k)$	Origin terminal of demand $k \in \mathscr{K}$
$D(k)$	Destination terminal of demand $k \in \mathscr{K}$
$o(k)$	Availability time of demand $k \in \mathscr{K}$ at its origin terminal
$d(k)$	Due date of demand $k \in \mathscr{K}$ at its destination terminal
$\mathscr{P}^k$	Set of itineraries in the service network for demand $k \in \mathscr{K}$
c^k	Unit service-quality cost, per unit of time, for demand $k \in \mathscr{K}$
u^{C}_η	Classification capacity, in number of cars, at yard $\eta \in \mathscr{N}$
u^{B}_η	Blocking capacity, in number of blocks, at yard $\eta \in \mathscr{N}$
u^{T}_η	Block-transfer capacity, in number of blocks, at yard $\eta \in \mathscr{N}$
u^{M}_η	Service make-up capacity, in number of services starting at yard $\eta \in \mathscr{N}$
τ^{C}_η	Expected delay to transfer a car at yard $\eta \in \mathscr{N}$
c^{T}_η	Unit car-transfer cost at yard $\eta \in \mathscr{N}$
c^{C}_η	Unit car-classification cost at yard $\eta \in \mathscr{N}$
f_b	Fixed building, transferring, and dismantling cost of block $b \in \mathscr{B}$
c^k_b	Unit transport cost for commodity $k \in \mathscr{K}$ on block $b \in \mathscr{B}$
$\mathscr{B} = \{b\}$	Set of blocks
$O(b), D(b)$	Origin and destination yards of block $b \in \mathscr{B}$
$\mathscr{N}(b) \subseteq \mathscr{N}$	Sequence of yards making up the route of block $b \in \mathscr{B}$
$\mathscr{L}(b)$	Sequence of service legs making up the route of block $b \in \mathscr{B}$
u_b	Capacity of block $b \in \mathscr{B}$
$\mathscr{B}(l_i(\sigma))$	Set of blocks assigned to service leg $l_i(\sigma) \in \mathscr{L}^{\mathrm{PH}}(\sigma), \forall\sigma \in \Sigma$
$\Theta = \{\theta\}$	Set of resource cycles
u^{R}	Quantity of resources available in the railroad system
$\mathscr{L}(\theta)$	Set of service legs of the resource cycle $\theta \in \Theta$
f_θ	Fixed cost of selecting and operating resources on cycle $\theta \in \Theta$
$\mathbf{T}$; $\mathscr{T}$	Schedule length; Set of discrete time periods

obtained by defining a sequence of time instances $t = 0, \dots, \mathbf{T}$, grouped in set $\mathscr{T}$. The time period t in this representation corresponds to the length of time between instances t and $t+1$, $t = 0, \dots, \mathbf{T}-1$, grouped in set $\mathscr{T}$. $\mathscr{N} = \{\eta_t, \eta \in \mathscr{N}^{\text{PH}}, t = 0, \dots, \mathbf{T} - 1\}$ includes copies of all the yards in the physical network at all the time periods defined. This definition may be specific to each yard. To simplify the presentation, however, and without loss of generality, we assume the same time definition for all yards in this chapter, and time periods of equal length.

Each SSND potential service $\sigma \in \Sigma$ is defined on the time-space network $\mathscr{G}$. It is characterized by a *schedule* indicating arrival and departure times at the yards where it originates, stops, and terminates. The sets of yards and service legs identifying the service then become $\mathscr{N}(\sigma)$ and $\mathscr{L}(\sigma)$ (with $\mathscr{L} = \bigcup_{\sigma \in \Sigma} \mathscr{L}(\sigma)$), respectively, with departure time from origin, $o(l_i(\sigma))$, and arrival time at destination, $d(l_i(\sigma))$, for each service leg $l_i(\sigma) \in \mathscr{L}(\sigma)$. Note that, the schedule may also be described in terms of arrival and departure times at the yards in $\mathscr{N}(\sigma)$. Similarly, each demand $k \in \mathscr{K}$ is characterized by an *availability time* $o(k)$ at origin $O(k)$ and a due date $d(k)$ at destination $D(k)$.

The network is completed by the set of arcs $\mathscr{A}$, which includes *moving* and *holding* arcs. The former correspond to the legs of the potential services $\mathscr{L}$, an arc being defined for each service leg, while the latter are arcs connecting two time-consecutive representations of each terminal in $\mathscr{N}$. The attributes of the moving arcs $a \in \mathscr{A}$, the capacity u_a and the unit commodity-transportation cost c_a^k, inherit the values of the corresponding service legs for both SND and SSND cases. Holding arcs $\mathscr{H} = \{(\eta_t, \eta_{t+1}), \eta_t, \eta_{t+1} \in \mathscr{N}\}$ represent the possibility for equipment and freight to wait at a terminal for a time period.

Several measures are used in the industry and the literature for the amplitude of demand and the capacity of the system components, e.g., number of cars, number of containers, tonnage, and length. Moreover, more than one may be used simultaneously to constrain decisions and operations. Thus, the characteristics of a track segment may limit both the total tonnage a train may haul on the track, its total length (and this, independently of the locomotive power assigned to the train). To simplify the presentation, but with no loss of generality, we use a single and same unit to measure demand and service capacity, the latter being specific for each of the legs of the service.

3 Static SND

The section is dedicated to static service network design formulations. The general hypothesis of this class of planning problems and SND formulations is that neither demand nor the other problem characteristics vary during the schedule length and, thus, they do not integrate the time dimension explicitly. Time may still be accounted, however, through the selection of services. Instead of a simple yes or no decision, the formulations may select a service and its *frequency* of operation over the schedule length. In the literature, one generally assumes that frequencies are uniformly distributed over the schedule length. We follow this trend in this chapter.

Models addressing particular components of tactical planning are presented in the first two subsections. Models integrating several tactical decisions are discussed in the third. We complete this part with a general discussion on the issue of generating the sets of potential services and blocks.

3.1 Service Selection and Train Makeup

The problem of selecting services and determining the cars their will haul arises when there is no blocking performed at yards (e.g., in most European railroads), or blocking is performed once the main service network is decided (see, e.g., the intermodal case described in Sect. 5). Given the set of possible services, the problem aims to (1) select the services to run and their frequencies over the planning period, and (2) assign cars to trains and determine the associated freight routing to accommodate all demand at minimum cost.

As in all problem settings considered in this chapter, freight routing may involve single-train itineraries from origin to destination and itineraries with service-to-service transfers at intermediary yards. Car transfers require time and resources. They generate costs and may cause delays related to many factors, including but not limited to, the number of cars to be transferred, the number of trains involved in transfers, and the capacity of the yard. Such delays not only increase the costs of the system, but may also decrease the service quality to customers.

The *Service Selection and Train Make-up Network Design (SMND)* problem thus aims to address these issues and decisions by minimizing the total operating cost, including penalties $\Phi(\cdot,\cdot)$ representing the interplay among delays and service quality standards. The model is built on a static network with $\mathscr{N} = \mathscr{N}^{\mathrm{PH}}$ representing the physical yards of the system, and $\mathscr{A} = \mathscr{L}^{\mathrm{PH}} = \bigcup_{\sigma \in \Sigma} \mathscr{L}^{\mathrm{PH}}(\sigma)$ standing for the legs of the potential service set. Time is implicitly considered through (1) the possibility to define services with different travel times between the same pairs of yards, (2) the frequencies of the selected services, and (3) the cost-penalty associated to the delays.

Let us define the decision variables

- $y_\sigma \in \mathbb{Z}_+$: Frequency of service $\sigma \in \Sigma$;
- $x_a^k \geq 0$: Flow of commodity $k \in \mathscr{K}$ traveling on arc $a \in \mathscr{A}$, with $x_{l_i(\sigma)}^k = x_a^k$, for $a = l_i(\sigma), l_i(\sigma) \in \mathscr{L}^{\mathrm{PH}}(\sigma), \sigma \in \Sigma$;
- $z_\eta^k = 1$ if commodity $k \in \mathscr{K}$ is transferred at yard $\eta \in \mathscr{N}^{\mathrm{PH}}(\sigma), \sigma \in \Sigma$, and 0, otherwise.

Let $\mathscr{A}_\eta^+ = \{a = (\eta, j) \in \mathscr{A}, j \in \mathscr{N}\}$ and $\mathscr{A}_\eta^- = \{a = (j, \eta) \in \mathscr{A}, j \in \mathscr{N}\}$ be the sets of outward and inward arcs (service legs) of node $\eta \in \mathscr{N}$. The SMND is then formulated as

$$\text{Minimize} \quad \sum_{\sigma \in \Sigma} f_\sigma y_\sigma + \sum_{k \in \mathscr{K}} \sum_{\sigma \in \Sigma} \sum_{l_i(\sigma) \in \mathscr{L}^{\text{PH}}(\sigma)} c^k_{l_i(\sigma)} x^k_{l_i(\sigma)}$$

$$+ \sum_{k \in \mathscr{K}} \sum_{\sigma \in \Sigma} \sum_{\eta \in \mathscr{N}^{\text{PH}}(\sigma)} \sum_{a \in \mathscr{A}^+_\eta} \Phi(x^k_a, z^k_\eta) \tag{13.1}$$

Subject to

$$\sum_{a \in \mathscr{A}^+_\eta} x^k_a - \sum_{a \in \mathscr{A}^-_\eta} x^k_a = \begin{cases} d^k, \text{ if } \eta = O(k), \\ -d^k, \text{ if } \eta = D(k), \ \forall \eta \in \mathscr{N}, k \in \mathscr{K}, \\ 0, \text{ otherwise}, \end{cases} \tag{13.2}$$

$$\sum_{k \in \mathscr{K}} x^k_{l_i(\sigma)} \le u_{l_i(\sigma)} y_\sigma, \ \forall l_i(\sigma) \in \mathscr{L}^{\text{PH}}(\sigma), \sigma \in \Sigma, \tag{13.3}$$

$$x^k_{l_i(\sigma)} - x^k_{l_i(\sigma)n} \le d^k z^k_{\eta_i},$$

$$k \in \mathscr{K}, i = 1, \dots, n(\sigma) - 1, \ l_i(\sigma) = (\eta_{i-1}, \eta_i) \in \mathscr{L}^{\text{PH}}(\sigma), \sigma \in \Sigma, \tag{13.4}$$

$$y_\sigma \in \mathbb{Z}_+, \ \forall \sigma \in \Sigma, \tag{13.5}$$

$$x^k_{l_i(\sigma)} = x^k_a \ge 0, \ \forall k \in \mathscr{K}, a \in \mathscr{A}, \tag{13.6}$$

$$z^k_\eta \in \{0, 1\}, \ \forall k \in \mathscr{K}, \eta \in \mathscr{N}^{\text{PH}}(\sigma), \sigma \in \Sigma. \tag{13.7}$$

Constraints (13.2) represent the usual flow conservation and demand satisfaction requirements. Linking constraints (13.3) ensure that the total load on any service leg cannot exceed the capacity of the service on that leg, provided the service is selected. Constraints (13.4) make sure that the transfer (and classification, eventually) costs are paid whenever such an operation is performed, by setting the transfer variable z^k_η to 1 whenever the flow of commodity k is transferred from service σ to a different at node η, except at the origin and destination of the demand.

The objective function (13.1) represents the total cost of the system computed as the sum of selecting, i.e., making up, operating, and dismantling, services at determined frequencies, and moving demand shipments on the selected services, plus a monetary evaluation of customer-service satisfaction. The later is captured through a penalty term $\Phi(x^k_a, z^k_\eta)$, which is application specific and may take various forms.

To illustrate, consider that transfers not only require time and resources, generating costs and delays, but also increase the possibility of missed connections and late arrival at destination of certain demand flows. Railroads thus aim to reduce the number of transfers and may also pay particular attention to commercially sensitive customers. Let c^{T}_η be the unit car-transfer cost at yard $\eta \in \mathscr{N}$, and τ^{C}_η the expected delay to transfer a car at the same yard. (τ^{C}_η may be defined to account for congestion in the yard and for the type of rail car or commodity involved, but, for simplicity of presentation, we use a linear term here.) Let also c^k be the service-quality cost per unit of time for demand $k \in \mathscr{K}$. We may then define for each arc (service leg)

$$\Phi(x^k_a, z^k_\eta) = \Phi(x^k_{l_i(\sigma)}, z^k_\eta) = (c^{\mathrm{T}}_\eta + c^k)\tau^{\mathrm{C}}_\eta x^k_{l_i(\sigma)} z^k_\eta, \tag{13.8}$$

which captures the yard and demand-specific costs of transferring cargo between services and potential loss of service quality. This modeling approach yields non-linear objective functions, however, increasing the computational challenges.

3.2 Car Classification and Blocking

As described in Sect. 2, car classification and grouping into blocks is central to aiming for the goal of efficiency and revenue maximization railroads. Recall that, cars with possibly different origins and destinations are classified and grouped into blocks; blocks are moved by trains and are handled as single unit from their origins to their destinations, where they are broken up. On the other hand, cars at their origin yard may follow itineraries where they are classified and assigned to a block which brings them to their destination yard, from where they are to be delivered to their consignees. Alternative itineraries may also be used where the initial block brings cars to an intermediate yard, where they are reclassified into new blocks, and continue their trip towards the final destination or another intermediate yard and reclassification. More than one reclassification may make up the itinerary.

The objective is to minimize cost. Decisions are highly constrained by the operating policies of the railroad (e.g., what blocks may be put on particular services), as well as by the resource and physical limitations of the yards in terms of, e.g., yard type and layout, numbers and characteristics of the yard equipment such as yard locomotives, personnel, and number and length of the classification tracks to which sorted cars are directed and where blocks are built. This translates, for each yard $\eta \in \mathcal{N}$, into unit car classification cost, c^{C}_η, as well as limits on the total number of cars which may be classified and blocked during a certain period of time, u^{C}_η, the total number of blocks one may build, u^{B}_η, or transfer, u^{T}_η, during the same time, the number of trains one may make up, u^{M}_η, etc.

Two approaches have been proposed to address this challenging car classification and blocking problem: (1) Develop the block plan first, then devise the set of train services (and schedule, possibly) to accommodate the blocks; (2) Select first the set of services and, second, build the block plan on the resulting service network. Both problems are challenging and SND formulations have been proposed to address them. Notice that, although it is the former which is mainly found in the literature, there is no methodological difference between the two in a static setting, except for the network on which the SND model is built, physical or service, respectively. We present the blocking problem in the block-first context in this subsection, together with a general formulation. The second case, often encountered when intermodal services are planned, is further detailed in Sect. 5.

The problem setting considers the physical (first case above) or the designed service network (second case). It is defined on a network $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, with $\mathcal{N} =$

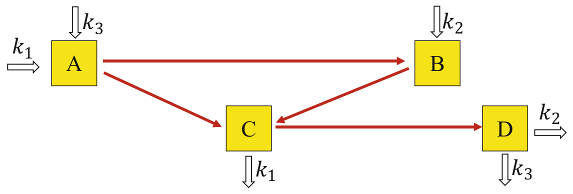


Fig. 13.3 Blocking SND network

$\mathscr{N}^{\text{PH}}$, $\mathscr{A} = \mathscr{B}$, the set of *potential blocks* linking the yards in $\mathscr{N}$, and the OD demand represented by set $\mathscr{K}$.

Similarly to the service definition, let $\mathscr{N}(b) \subseteq \mathscr{N}$ be the sequence of yards where the block $b \in \mathscr{B}$ is formed (origin $O(b)$), is dismantled (destination $D(b)$), and transferred from one service to a different one. Then, let $\mathscr{L}(b)$ be the sequence of service legs, $l_i(\sigma) \in \mathscr{L}^{\text{PH}}$, transporting the block from its origin to its destination, through the transfer yards, when relevant. Let $\delta_\sigma^b = 1$, if at least a service leg of service $\sigma \in \Sigma$ moves block $b \in \mathscr{B}$, and 0, otherwise, and let $\mathscr{B}(l_i(\sigma))$ be the set of blocks assigned to service leg $l_i(\sigma) \in \mathscr{L}^{\text{PH}}$, $\forall \sigma \in \Sigma$. Finally, let f_b be the block (fixed) building, transferring, and dismantling cost, c_b^k the unit cost of transporting commodity $k \in \mathscr{K}$ from the origin to the destination of the block, and u_b the block *capacity*, measured in the same units used for services.

To illustrate, consider the simple four-yard network displayed in Fig. 13.3, with four directed rail tracks, three OD commodities, and eight potential blocks $b_1 = (A, B), b_2 = (A, C), b_3 = (A, B, C), b_4 = (A, C, D), b_5 = (A, B, C, D), b_6 = (B, C), b_7 = (B, C, D), b_8 = (C, D)$, the intermediate yard labels identifying the block route not transfers. The possible commodity itineraries are: three for k_1: b_2, b_3, and (b_1, b_6) with reclassification at B; two for k_2: b_7, (b_6, b_8) with reclassification at C; and six for k_3: two direct, blocks b_4 and b_5, or with reclassification at yards B or C, or both, via the block paths (b_1, b_6, b_8), (b_1, b_7), (b_2, b_8), and (b_3, b_8), respectively.

The goal is to select the blocks to build from within $\mathscr{B}$, and to assign OD demand commodities to them, at minimum total cost, computed as the sum of the fixed cost of building, transferring, and dismantling the blocks, the cost of car classification, and the car transportation cost. Notice that, even though $\mathscr{A} = \mathscr{B}$, we write the formulation in terms of $\mathscr{B}$ to emphasize the classification and blocking scope of the model. Define the decision variables

- y_b = 1, if block $b \in \mathscr{B}$ is built, and 0, otherwise;
- x_b^k, continuous flow variable representing the volume of commodity $k \in \mathscr{K}$ assigned to block $b \in \mathscr{B}$.

The car classification and blocking service network design formulation takes then the following form:

$$\text{Minimize} \quad \sum_{b\in\mathscr{B}} f_b y_b + \sum_{k\in\mathscr{K}} \sum_{b\in\mathscr{B}} c_b^k x_b^k + \sum_{k\in\mathscr{K}} \sum_{\eta\in\mathscr{N}} \sum_{b\in\mathscr{B}_\eta^+} c_\eta^{\mathrm{C}} x_b^k \tag{13.9}$$

Subject to

$$\sum_{b\in\mathscr{B}_\eta^+} x_b^k - \sum_{b\in\mathscr{B}_\eta^-} x_b^k = \begin{cases} d^k, \text{ if } \eta = O(k), \\ -d^k, \text{ if } \eta = D(k), \\ 0, \text{ otherwise}, \end{cases} \quad \forall\, \eta \in \mathscr{N}, k \in \mathscr{K}, \tag{13.10}$$

$$\sum_{k\in\mathscr{K}} x_b^k \le u_b y_b, \ \forall\, b \in \mathscr{B}, \tag{13.11}$$

$$\sum_{k\in\mathscr{K}} \sum_{b\in\mathscr{B}_\eta^+} x_b^k \le u_\eta^{\mathrm{C}}, \ \forall\, \eta \in \mathscr{N}, \tag{13.12}$$

$$\sum_{b\in\mathscr{B}_\eta^+} y_b \le u_\eta^{\mathrm{B}}, \ \forall\, \eta \in \mathscr{N}, \tag{13.13}$$

$$y_b \in \{0, 1\}, \ \forall\, b \in \mathscr{B}, \tag{13.14}$$

$$x_b^k \ge 0, \ \ \forall\, b \in \mathscr{B}, k \in \mathscr{K}, \tag{13.15}$$

where $\mathscr{B}_\eta^+ = \{b = (\eta, j) \in \mathscr{B}, j \in \mathscr{N}\}$ and $\mathscr{B}_\eta^- = \{b = (j, \eta) \in \mathscr{B}, j \in \mathscr{N}\}$ are the sets of outward and inward arcs of node $\eta \in \mathscr{N}$.

The objective function (13.9) represents the total cost measured as the total fixed cost of building, transferring, and dismantling blocks, total cost of moving cars on blocks, and total car classification cost at yards. Constraints (13.10) and (13.11) are the classical flow conservation and block linking and capacity constraints, respectively. Constraints (13.12) and (13.13) enforce the yard capacity limits in terms of the number of cars that can be classified and the number of blocks that can be built during the planning period. Decision-variable ranges are defined by constraints (13.14) and (13.15).

3.3 *Integrated Planning SND*

Sections 3.1 and 3.2 addressed the issues of selecting and making-up trains, and classifying cars and building blocks separately. Yet, the solution to one problem is affecting the planning and solution of the other, no matter which problem is considered first. To emphasize the strong relations among these issues, consider, on the one hand, that the availability and frequency of a service determine the possibility of building and transporting blocks using that service while, on the other hand, the usefulness of a train service depends on the amount of traffic, in terms of blocks and cars, the train may service. Integrated-planning SND formulations address these issues simultaneously to select the service and the block networks and, thus, define the classification strategy, as well as to determine the demand itineraries, establishing how freight is to be routed through the service and block network.

We start with the arc and path formulations of the SND model for this problem in Sects. 3.3.1 and 3.3.2, respectively. Section 3.3.3 presents a path formulation when one extends the SND model to account for more advanced features such as non-additive costs/tariffs and congestion phenomena. We conclude the section with a short discussion of service and block generation issues in Sect. 3.4.

3.3.1 Arc-Based Integrated SND

The integrated SND model is built on the network G of potential services. Then, $\mathscr{N} = \mathscr{N}^{\mathrm{PH}}$ and $\mathscr{A} = \mathscr{L}^{\mathrm{PH}}$. The components and notation of Sects. 3.1 and 3.2 apply directly to the integrated context, in particular the yard, service, and block definitions and characteristics. We recall the decision-variable definitions allowing the formulation to address simultaneously the selection of services with their frequencies, the selection of blocks to build at each yard, and the itineraries of demand determining the routing of the flows within the service and block networks and, thus, the classification strategy at each yard:

- $y_\sigma \in \mathbb{Z}_+$: Frequency of service $\sigma \in \Sigma$;
- $y_b = 1$, if block $b \in \mathscr{B}$ is built, and 0, otherwise;
- $x_b^k \geq 0$, continuous flow variable representing the volume of commodity $k \in \mathscr{K}$ assigned to block $b \in \mathscr{B}$; as the cars grouped within a block are the same over all the route of the block, that is, on all the service legs of the services carrying it, $x_b^k = x_{l_i(\sigma)}^k$, $l_i(\sigma) \in \mathscr{L}^{\mathrm{PH}}(\sigma)$, $\sigma \in \Sigma$ (and equal to x_a^k as $a = l_i(\sigma)$).

The integrated service and block selection with classification model is formulated as mixed integer SND:

$$\text{Minimize} \quad \sum_{\sigma \in \Sigma} f_\sigma y_\sigma + \sum_{b \in \mathscr{B}} f_b y_b + \sum_{k \in \mathscr{K}} \sum_{b \in \mathscr{B}} c_b^k x_b^k + \sum_{k \in \mathscr{K}} \sum_{\eta \in \mathscr{N}} \sum_{b \in \mathscr{B}_\eta^+} c_\eta^{\mathrm{C}} x_b^k \tag{13.16}$$

Subject to constraints (13.5), (13.10)–(13.15), and

$$y_b \leq y_\sigma, \quad \forall b \in \mathscr{B}(l_i(\sigma)), l_i(\sigma) \in \mathscr{L}^{\mathrm{PH}}(\sigma), \sigma \in \Sigma, \tag{13.17}$$

$$\sum_{k \in \mathscr{K}} \sum_{b \in \mathscr{B}(l_i(\sigma))} x_b^k \leq u_{l_i(\sigma)} y_\sigma, \quad \forall l_i(\sigma) \in \mathscr{L}^{\mathrm{PH}}(\sigma), \sigma \in \Sigma, \tag{13.18}$$

where the objective function (13.16) computes the total system cost of selecting and operating services, building and hauling blocks and cars, and classifying cars. (Note that the costs related to car handling in yards captured by $\Phi(x_a^k, z_\eta^k)$ in the service-selection case, Sect. 3.1, are included in the classification and blocking costs.) Constraints (13.17) link the building of blocks to the selection of the services which move them, while constraints (13.18) enforce the service capacity limits in terms of cars hauled on each service leg given the blocks that can be moved on that leg.

3.3.2 Path-Based Integrated SND

It is well-known that one may write network design models in arc and path forms (see, e.g., Chap. 2), each with its own pros and cons. For example, path formulations generally involve a huge number of variables, but are amenable to decomposition and the utilization of solution techniques based on column generation. For service network design, services are paths in the physical network. The same is true for blocks in the freight railroad SND case. Hence the "arc" or "path" qualification in this context refers generally to the representation of the freight flows on the service network, that is, to the modeling of the demand itineraries.

We define, for the path-version of the model of Sect. 3.3.1, the set $\mathscr{P}^k$ of itineraries, *paths through the service network* $G = (\mathscr{N}^{\text{PH}}, \mathscr{L}^{\text{PH}})$, with potential set of blocks $\mathscr{B}$, which may be used to transport all or some part of the volume of demand $k \in \mathscr{K}$. Indicators detail the definition of each itinerary, linking the arc and path flow variables on blocks and at classification yards. Let $\delta_\eta^p = 1$ when the cars following the itinerary $p \in \mathscr{P}^k$ (re-)classify at yard $\eta \in \mathscr{N}$, and 0, otherwise. Similarly, let $\delta_{l_i(\sigma)}^p$ and δ_b^p to equal 1 when the itinerary p includes the service leg $l_i(\sigma) \in \mathscr{L}^{\text{PH}}$ and the block $b \in \mathscr{B}$, respectively, and 0, otherwise.

Let us assume that the unit itinerary (path) cost may be computed as the sum of the unit transportation and classification costs associated to the services and yard classification activities making it up. This is a wide-spread hypothesis in the literature and practice and it does correspond to many actual problem settings. The unit cost of itinerary $p \in \mathscr{P}^k$ then becomes $c_p^k = \sum_{\sigma \in \Sigma} \sum_{l_i(\sigma) \in \mathscr{L}^{\text{PH}}(\sigma)} c_{l_i(\sigma)}^k \delta_{l_i(\sigma)}^p + \sum_{\eta \in \mathscr{N}} c_\eta^{\text{C}} \delta_\eta^p$.

With respect to decision variables, the service and block selection variables defined previously are also part of this model. Flow variables, however, are defined as h_p^k, standing for the quantity of commodity $k \in \mathscr{K}$ assigned to its itinerary $p \in \mathscr{P}^k$. The path formulation of the integrated service design & block selection with classification model may be written as:

$$\text{Minimize} \quad \sum_{\sigma \in \Sigma} f_\sigma y_\sigma + \sum_{b \in \mathscr{B}} f_b y_b + \sum_{k \in \mathscr{K}} \sum_{p \in \mathscr{P}^k} c_p^k h_p^k \tag{13.19}$$

Subject to constraints (13.5), (13.11)–(13.15), (13.17)–(13.18), and

$$\sum_{p \in \mathscr{P}^k} h_p^k = d^k, \qquad \forall k \in \mathscr{K}, \tag{13.20}$$

$$h_p^k \geq 0, \qquad \forall p \in \mathscr{P}^k,\ k \in \mathscr{K}, \tag{13.21}$$

$$x_b^k = \sum_{p \in \mathscr{P}^k} \delta_b^p h_p^k, \qquad \forall b \in \mathscr{B}, k \in \mathscr{K}. \tag{13.22}$$

Constraints (13.20) ensure all demand is moved to its final destination (and enforces the flow conservation at the nodes of the network), while constraints (13.21) define the domain of the path flow variables. Finally, relations (13.22) are definitional constraints linking the arc and path flow variables on blocks.

3.3.3 Advanced Path-Based Integrated SND

Most planning models in the literature and this chapter, including the previous path-based one, assume strict capacity restrictions, no waiting due to congestion, additive path characteristics with the composing arcs and nodes, and linear cost (and time) functions in the decision-variable values. While reasonable in many cases, and making solving somewhat easier, such hypotheses limit the scope of the planning models. We discuss these limitations in the following, together with a modeling framework addressing them. Although presented for the static SND case, the discussion and modeling framework are general, including for the time-dependent case.

Capacity constraints are ubiquitous in practice and OR models. They obviously apply at operation time. One cannot load more containers on a car than it can physically accommodate. At the tactical planning however, one is generally less concerned with how the capacity of each individual car, train or yard is filled up, and much more interested in identifying the service network and flow distribution for an optimal usage of those resources and capacities. Thus, the assignment of some quantity of freight to a particular service resulting in exceeding its capacity may indicate either that the frequency of the service should be increased, or that some less important (in terms of priority or delay costs) traffic should pass to another service. The formulation of strict capacity constraints would prevent, however, the detection and handling of such a situation by the solution method. Moreover, it is also known that assigning more flow to a service or a yard does not result in stopping the system activities. It rather translates, in practice, either in delays for the respective freight, which will wait for the next departure, or in additional resources being brought on line. Increased costs and, possibly, delays, occur in both cases.

Treating such limits as *utilization targets* rather than strict constraints, and including in the objective function penalties for the over utilization of the capacity, addresses these issues. Consider, to illustrate, the service-leg capacity constraints (13.18). Let α_σ be the unit penalty cost of overloading service $\sigma \in \Sigma$. A rather simple utilization-target penalty may be written for each service leg of the service as

$$\Psi_{l_i(\sigma)}(y_\sigma)=\alpha_\sigma \left(\max \left\{ 0, \sum_{k\in\mathscr{K}} \sum_{b\in\mathscr{B}(l_i(\sigma))} x_b^k - u_{l_i(\sigma)} y_\sigma \right\} \right)^n, \ \forall l_i(\sigma) \in \mathscr{L}^{\mathrm{PH}}(\sigma), \sigma \in \Sigma, \tag{13.23}$$

where n represents a certain degree of unwillingness of letting the tactical plan overloading the resource too much.

A similar approach may be used to model resource availability limitations with respect, e.g., to locomotives or railroad cars of particular types. Global limits for the network or targeted by yard may be handled in this way. *Service-quality targets* for demand, specifying, for example, the total duration of the origin to destination activity chain, may also be addressed in this way. Such targets may have been publicized or promised to specific customers only. Delay (time) measures are associated in such problem settings to the yard and long-haul movement activities and, thus, to services and itineraries. A capacity-like constraints may then be imposed on the itinerary duration with respect to the service target. But, again, such a constraint would provide the opportunity to trade off a penalty on some ODs against a more significant reduction in costs in other regions, generated by a more cost-or time-efficient deployment of resources. An itinerary-specific penalty may then be computed as the difference between the itinerary duration and the service target, weighted by a demand-specific penalty cost, which may represent the penalty the railroad must pay when delivering late or an estimation of the potential market-share loss.

Penalties defined according to (13.23) represent a rather strict translation of the capacity and target constraints, which does not account for the well-known fact that getting close to the capacity limits is not suitable in several cases. Consider, for example, yard classification capacities. Trains bring cars in batches, each according to its more or less followed schedule. These cars are then handled by a limited number of resources, with varying characteristics and performance measures, proper to the yard type. Queuing phenomena and congestion are a direct consequence of such situations, which may be observed for various yard activities (e.g., classification, container loading/unloading, and interservice block transfer), as well as for long-haul movements when several freight and, possibly, passenger trains share a single or double-track with restricted capacity (due, e.g., to too few or too short sidings). Models based on queuing theory have been proposed in the literature to account for these phenomena. Queuing models or networks of queues were proposed and used mostly to simulate operations. Such models are very detailed, however, and generally yield non-continuously differentiable functions, which is very hard to handle, particularly for large-scale formulations. Consequently, continuous non-linear functions were proposed to approximate such congestion behavior within network-wide SND formulations addressing tactical-planning issues.

Let the decision-variable vectors $\mathbf{y}$ and $\mathbf{h}$ indicate a given level of service in Σ and flow distribution in $\mathscr{P}$, respectively. Let then

$F_\sigma(\mathbf{y}, \mathbf{h})$:	Total (fixed) cost of operating service $\sigma \in \Sigma$;
$F_b(\mathbf{y}, \mathbf{h})$:	Total (fixed) cost of building, hauling, and dismantling block $b \in \mathscr{B}$;
$C^k_p(\mathbf{y}, \mathbf{h})$:	Total unit cost for itinerary $p \in \mathscr{P}^k$;
$\Psi(\mathbf{y}, \mathbf{h}))$:	Penalty terms capturing various relations and restrictions, such as the limited service capacity.

The objective function of the path-based integrated SND formulation may then be written in a general form

$$\text{Minimize} \sum_{\sigma \in \Sigma} F_\sigma(\mathbf{y}, \mathbf{h}) y_\sigma + \sum_{b \in \mathscr{B}} F_b(\mathbf{y}, \mathbf{h}) y_b + \sum_{k \in \mathscr{K}} \sum_{p \in \mathscr{P}^k} C^k_p(\mathbf{y}, \mathbf{h}) h^k_p + \Psi(\mathbf{y}, \mathbf{h}), \tag{13.24}$$

where costs depend on the complete status of the network as given by the **y** and **h** vectors at the corresponding iteration of the solution method. It may be very difficult, in practice, to develop and calibrate such general functions for railroads of realistic dimensions and complexity. The impact of distant activities on a given service, block, or yard may be hard to evaluate and might not be very important. Consequently, most models adopting this approach consider nearby interactions only, within each individual yard, for example.

Notice that service-quality targets and time-related measures open a number of possibilities for more flexible modeling and accounting for the cost of time, even in a so-called static formulation. Thus, one may model frequency (or connection) delays encountered when one must transfer between two services with different frequencies and, thus, with differences in their presence at the same yard. One may also define time-related costs for services and demands, and use them to weight the total time required to go from origin to destination through the various system activities. The delay cost for demand usually represents the penalties in case of late delivery. It may also be used to model priorities, the time sensitiveness for certain commodities, and customer-service classes, a higher cost pushing the corresponding demand flows more rapidly through the system. For services, these costs may represent depreciation values and inventory costs as well as, according to the railroad's accounting practices manpower or energy-consumption costs.

The objective function thus computes a generalized cost, in the sense that it may include a broad range of productivity measures related to terminal and transportation operations, in terms of time, cost, and quality and reliability of the service offered. This enhances modeling refinement and flexibility, providing the opportunity for enhanced trade-off analyses among cost and service-quality objectives, as well as among the impact and value of activities and resource utilization. The gains come, however, at the price, the SND formulations taking the form of nonlinear integer multicommodity network design problems.

3.4 Service & Block Generation and SND Models

The static and time-dependent models described in this chapter proceed as most network design models do, by selecting from a set of candidate, potential, arcs. More precisely, for the railroad case, from potential sets of services Σ and blocks $\mathscr{B}$. How these sets are generated is a valid question, which is relevant for most SND applications (see, e.g., Chap. 12), and more so for railroad transport with its several levels of consolidation and combinatorial complexity. We briefly discuss the topic in the context of static SND, but everything applies to time-dependent settings too. In fact, the latter case presents even greater challenges, the time dimension of the problem setting exacerbating the combinatorial multiplication of the number of potential services and blocks.

The cardinality of these sets may be very large. Consider, for example, that a service may, in theory, be defined between every pair or yards in the network, on every possible physical path, with every possible combination of stops at the yards on that physical path, as well as for every type of service in terms of power, capacity, speed, priority, and so on and so forth. Blocks may then be defined similarly but on the network made up of all those potential service legs. Obviously, full enumeration of all potential services and blocks is not more realistic for railroads than for the other modes or other situations of a similar nature, e.g., crew scheduling in passenger and freight transportation. On the one hand, full enumeration yields problem dimensions extremely difficult to manipulate and solve, even when stringent feasibility checks are enforced. On the other hand, trying to generate “good” services and blocks only, with respect to limits on costs and time, for example, generally eliminates elements contributing to very good or optimal solutions. Hence, a systematic service and block generation procedure tightly linked to or part of SND formulations is needed.

Partial targeted enumeration is appropriate in many practical cases when the plan for the next season is based on the previous one, adjusted for the trends and predictions in demand, prices, and the regulatory environment identified by management. The past service and block networks are then enriched with a number of additional possibilities reflecting these trends and predictions. Yet, even in such situations, one faces the problem of missing elements required for very good solutions, and a more systematic procedure is required.

The goal is thus to include the generation of the service and block sets into SND formulations. We illustrate the difficulty of arc-based formulations focusing on the case when one starts with the set of potential services, the blocks are to be generated together with the tactical plan, at most one block is created for each pair of yards.

The problem description and notation of Sect. 3.3.1 (and previous ones) apply except for the block definition, which is reduced to the origin and destination yards, $O(b)$ and $D(b)$, respectively, of block $b \in \mathscr{B}$. The path in the service network $\mathscr{L}^{\mathrm{PH}}$ is thus not part of the input, but is an output of the optimization problem. Thus, at most $|\mathscr{B}| = |\mathscr{N}^{\mathrm{PH}}|^2 - |\mathscr{N}^{\mathrm{PH}}|$, which is relatively small. This gain in problem dimensions and number of integer block-selection variables is paid for, however, in increasing numbers and complexity of constraints, as shown in the following.

Given the updated definition of $\mathscr{B}$, the fixed cost f_b includes only the cost relative to building the bloc at the origin yard and dismantling it at destination. The inter-service transfer costs must be identified and, then, computed separately. We model this through a function Φ, which can be of any form but accounts for the characteristics and operating policies of the yard and the number of blocks to transfer. Other than the $y_\sigma, \sigma \in \Sigma$, $y_b, b \in \mathscr{B}$, and $x_b^k, b \in \mathscr{B}, k \in \mathscr{K}$, decision variables of Sect. 3.3.1, we define

- $y_{bl_i(\sigma)} = 1$ if block $b \in \mathscr{B}$ is moving on service leg $l_i(\sigma) \in \mathscr{L}^{\mathrm{PH}}(\sigma), \sigma \in \Sigma$, and 0 otherwise;
- $z_\eta^b = 1$ if block $b \in \mathscr{B}$ is transferred at yard $\eta \in \mathscr{N}^{\mathrm{PH}}$ from one service to another, and 0 otherwise.

The SND formulation with block generation minimizes the total system cost (13.25), computed as the service- and bloc-selection costs, plus the cost of moving cars on blocks given the service leg used to haul the block, the car classification cost at yards where blocks are generated, and the cost of transferring blocks between services.

$$\begin{aligned}\text{Minimize} \quad & \sum_{\sigma\in\Sigma} f_\sigma y_\sigma + \sum_{b\in\mathscr{B}} f_b y_b + \sum_{k\in\mathscr{K}} \sum_{b\in\mathscr{B}} \sum_{l_i(\sigma)\in\mathscr{L}^{\mathrm{PH}}} c^k_{l_i(\sigma)} x^k_b y_{bl_i(\sigma)} \\ & + \sum_{k\in\mathscr{K}} \sum_{\eta\in\mathscr{N}} \sum_{b\in\mathscr{B}^+(\eta)} c^{\mathrm{C}}_\eta x^k_b + \sum_{\eta\in\mathscr{N}} \sum_{b\in\mathscr{B}} \Phi(z^b_\eta) \qquad (13.25)\end{aligned}$$

Subject to (13.10)–(13.13), and

$$\sum_{\sigma\in\Sigma} \sum_{l_i(\sigma)\in\mathscr{A}^+_\eta} y_{bl_i(\sigma)} - \sum_{\sigma\in\Sigma} \sum_{l_i(\sigma)\in\mathscr{A}^-_\eta} y_{bl_i(\sigma)} = \begin{cases} y_b, \text{ if } \eta = O(b), \\ -y_b, \text{ if } \eta = D(b), \\ 0, \text{ otherwise}, \ \forall\, \eta \in \mathscr{N}, b \in \mathscr{B}, \end{cases} \qquad (13.26)$$

$$\begin{aligned} y_{bl_i(\sigma)} - y_{bl_{i+1}(\sigma)} &\le z^b_{\eta_i}, \\ \forall b \in \mathscr{B}, i &= 1, \ldots, n(\sigma) - 1,\ l_i(\sigma) = (\eta_{i-1}, \eta_i) \in \mathscr{L}^{\mathrm{PH}}(\sigma), \sigma \in \Sigma, \qquad (13.27) \\ \sum_{k\in\mathscr{K}} \sum_{b\in\mathscr{B}} x^k_b y_{bl_i(\sigma)} &\le u_{l_i(\sigma)} y_\sigma,\ \forall l_i(\sigma) \in \mathscr{L}^{\mathrm{PH}}, \sigma \in \Sigma, \qquad (13.28) \\ y_{bl_i(\sigma)} &\in \{0, 1\},\ \ \forall\, b \in \mathscr{B}, l_i(\sigma) \in \mathscr{L}^{\mathrm{PH}}, \sigma \in \Sigma, \qquad (13.29) \\ y_\sigma &\in \mathbb{Z}_+,\ \ \forall\, \sigma \in \Sigma, \qquad (13.30) \\ y_b &\in \{0, 1\},\ \ \forall\, b \in \mathscr{B}, \qquad (13.31) \\ z^b_\eta &\in \{0, 1\},\ \ \forall\, b \in \mathscr{B}, \eta \in \mathscr{N}, \qquad (13.32) \\ x^k_b &\ge 0,\ \ \forall\, b \in \mathscr{B}, k \in \mathscr{K}. \qquad (13.33) \end{aligned}$$

Constraints (13.26) and (13.27) enforce the building of blocks conditions. The former are the block-building constraints ensuring that a single path is selected in $\mathscr{L}^{\mathrm{PH}}$ for each block from its origin to its destination. The latter, (13.27), are linking relations ensuring that block transfers are accounted for, and that the corresponding costs will be paid, by setting the transfer decision variable z_{η}^{b} to 1 whenever block b is transferred from a service to a different one at yard η, except at the origin and destination of the block. Constraints (13.28) are the flow-service linking and capacity constraints, given the service legs moving the block transporting the cars. Restrictions on the decision variables are enforced by constraints (13.29)–(13.33).

It is noteworthy that a sleeker set $\mathscr{B}$ and, thus, fewer y_b selection variables, translates into a large number of constraints, namely (13.26) and (13.27), and decision variables, $y_{bl_i(\sigma)}$, required to build the blocks out of service legs and transfers. It is also worth noticing that both the objective function (13.25) and constraints (13.28) are non linear. This is not surprising and the issue can be addressed, but it does not make the problem dimensions smaller nor the formulation easier to address. These observations are not unique to railroad planning, but have been made in many other settings, e.g., crew scheduling and vehicle routing. Dynamic path generation techniques, based on Column Generation techniques, have been applied in such settings and appear promising for service network design and railroad tactical planning. Most work has still to be undertaken in this field, which constitutes a challenging but interesting research direction, particularly when the time dimension is explicitly considered as in the models of the following sections.

4 Time-Dependent SND and Integrated Planning

We now turn to time-dependent SND formulations, also known as *Scheduled Service Network Design* (*SSND*) models. As discussed in Sect. 2.3, SSND targets time-dependent problem settings, explicitly representing the time-related characteristics of demand, in terms of availability time at origin and due time at destination. To answer the requirements of time-dependent demand, the time characteristics of the service the railroad offers is also explicitly represented, in terms of a schedule stating the departure and arrival times at each of the yards on the route of each individual service. The aim is thus not only to select the service network, but also the schedule of the selected services to address the time-dependent demand.

Most SSND models address a broader set of planning issues than selecting services only, and are generally qualified as *integrated-planning methods*. The general SSND modeling framework presented herein for the integrated freight-railroad planning problem addresses the main tactical-planning issues: service selection and scheduling, blocking and classification, train makeup, and freight routing. The goal is to minimize the total cost of the system, while satisfying demand with the available resources. The framework is extended in Sect. 5 to address exiting schedules, intermodal traffic, and resource management at the tactical planning level.

Most of the notation is introduced in Sect. 2.3 and Table 13.1. It is briefly recalled and completed in the following. SSND formulations are built on time-space networks, defined over the total duration of the schedule length, most of them using a discrete representation of time. The mixed-integer network design formulation presented in this section follows this classical approach, adapted for the multiple interrelated decisions involved in the problem setting. The model is thus built on a *multi-layer time-space* network $\mathscr{G} = (\mathscr{N}, \mathscr{A})$, using a discrete representation of the schedule length $\mathbf{T}$. The nodes are representations of the physical nodes (yards, mainly) at all time periods. Two nodes, $\eta_t^{\text{IN}}, \eta_t^{\text{OUT}} \in \mathscr{N}$, are created for each yard $\eta \in \mathscr{N}^{\text{PH}}$ and time period $t \in \mathscr{T}$, to capture all the traffic coming into the node and going out of the node, respectively.

Arcs represent movement on service legs and holding activities at nodes. Recalling that the scheduled service plan is to be applied repeatedly over the tactical-planning horizon, $\mathscr{G}$ takes on a *cyclic* nature. The network thus reflects the fact that activities and decisions do not stop with the end of the schedule length, but rather involve the next application of the scheduled plan. Thus, for example, when building a week-long schedule, a service may start on Friday and arrive at destination the following Tuesday. We model these situations by having the corresponding arcs *wrap around*. In modeling terms, this means that the destination for an arc with origin at time t and a duration which would make it arrive at a time $> \mathbf{T}$ is defined at a time $t' < t$ through a *modulo* computation.

We present the SSND model on a *three-layer* time-space network, schematically illustrated in Fig. 13.4. Multi-layer networks make up a general methodology with applications in transport and telecommunications. Integrated railroad planning offers a very good illustration. Each layer represents the activities and decisions which focus on the particular type of flow, cars, blocks, and services. The arcs

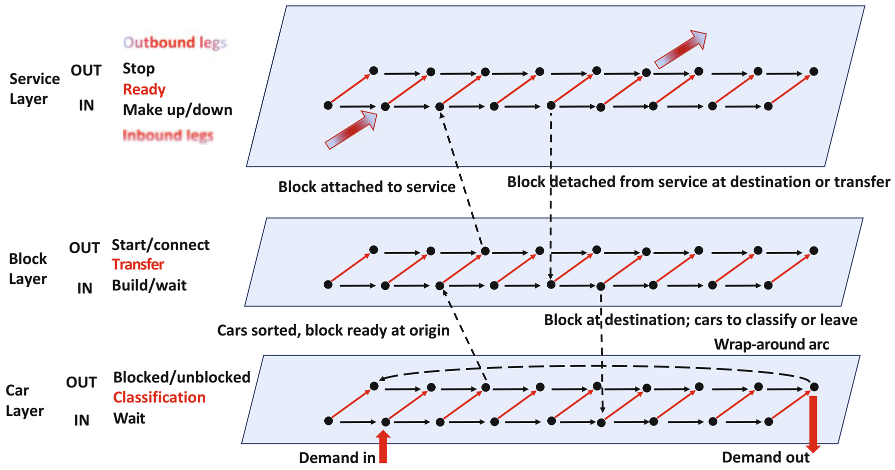


Fig. 13.4 Three-layer time-space SSND network

in each layer stand for the operations taking place in terminals, impacting principally the corresponding flows. The inter-layer arcs model the assignment and consolidation/de-consolidation of these flows, from cars to blocks to services and vice-versa, the classification, transfer, and makeup activities being modeled within each layer. Each layer is thus a “complete” time-space network. In the model we present in this chapter, the time and schedule length definitions are the same for all layers. Consequently, the node set $\mathscr{N}$ is the union of the node sets in all layers. Similarly, the arc set $\mathscr{A}$ is the union of all arcs, movement and holding, in all layers, plus the inter-layer arcs supporting the flow movements between layers. To simplify the presentation, we do not detail the notation based on layers, except when needed to avoid confusion (Sect. 6 points to literature with detailed notations).

Cars, loaded and empty, enter the network through an IN node within the *car layer*. They exit the network through an OUT node in the same layer. Section 5 adds a container loading/unloading layer within the context of intermodal rail transportation. The holding arcs between IN nodes represent the waiting time for classification. In this formulation, the yard classification capacity is defined by time period and is associated to the classification links. The interplay between this limit and the flow of cars requiring classification determines for how long (given by the number of waiting links) cars have to wait before being sorted. The blocked/unblocked links capture the time waiting, once classified, on the appropriate block track for the appropriate number of cars to accumulate and the block to be ready. They also represent the possible waiting at the final destination when arrived too early. The car layer illustration also shows a wrapped-around (blocked/unblocked) arc (for clarity of illustration we do not show such arcs on the other layers).

The car and the block layers are linked through two types of arcs. The first moves the sorted and blocked cars, i.e., the block, to the block origin in the block layer. Symmetrically, block-to-car arcs move the cars on a block at destination back to the car layer, to be either re-classified (not at their destination, yet) and put on a new block, or to exit the yard for final distribution (cars at their final destination).

The *block layer* focuses on selecting the blocks, the block-to-service assignment, and the associated operations of attaching a block to a train or detaching the block from a train. The attach-to-train operation involves the new blocks at their origins, and blocks transferring at intermediary yards. The detach-from-train operation applies to blocks at their destinations, and blocks requiring transfer to a different train. The build/wait arcs in the block layer capture the blocks ready to be attached/transferred, while the start/connect arcs capture the waiting for the departure service. Transfer arcs stand for the physical operations of moving blocks to or between trains and may model limited capacity and force waiting.

Block-to-service and service-to-block arcs link the block and service layers. The former connect OUT nodes in the block layer to IN nodes in the service layer and represent adding the blocks, new or transferred, to trains at the given period. The latter connect IN service nodes to IN block nodes, taking the blocks off their current services for transfer or dismantling (when at destination).

Arcs in the *service layer* represent service selection, schedules, and operations (Sect. 2.3). Services start from an OUT node at the specified starting time period. Direct services terminate their routes at an IN node at the period specified in its schedule. Multi-stop services, arrive at the first stop at an IN node, stop at the yard for the specified number of periods, leave from the corresponding OUT node, move to the IN node of the next stop, and so on and so forth until the final destination. Differently from the two previous layers, the service layer thus includes explicitly moving arcs representing the service legs of the potential services according to their schedules. Two such arcs only are sketched in Fig. 13.4. The figure also shows the make up/down arcs on which trains stay while taking out blocks at destination or being transferred or taking in blocks for the outbound move. The stop arcs complete the stop length until the departing service leg. The ready arcs complete the modeling of the service yard activity and may be used to model yard capacity limitations.

A few notes before introducing the SSND formulation. A demand *itinerary* is then a path in the three-layer time-space network between the IN and OUT nodes in the car layer representing the corresponding origin and destination yards at the availability and due dates, respectively. When the cars are delivered before the due date, they wait on the blocked/unblocked arcs (the model may be easily modified to account for late deliveries and penalties). The itinerary then includes the wait arcs, a classification arc, reaching an OUT node, where they wait for accumulation of cars on blocked/unblocked arcs. Once the block is formed, the traffic goes up to the block layer, where the block journeys to its destination yard, where the block is dismantled and the cars return down to the car layer at an IN node for final delivery or re-classification. The journey continues as described in the latter case until the final destination of the demand.

A block journey may be similarly described, from an IN node in the block layer, through build/wait arcs, a transfer arc, and start/connect arcs until the OUT node when the block is ready to be put on the train through an inter-layer arc. Once in the service layer, the block journeys through a sequence of service legs interspersed with movements down to the block layer, the arcs involved in the transfer operation, and then back up to the service layer and the next segment on the block route.

It is noteworthy that "moving arcs" may be shown in the car layer by projecting on it the appropriate blocks making up each itinerary. Similarly, moving arcs in the block layer are obtained by projecting the corresponding service legs. Notice, finally, that parallel arcs may exist in the service layer standing either for train movements following different physical routes between the two yards with the same departure and transit times, or for services of different types (e.g., regular cargo and intermodal) sharing the same infrastructure.

The parameter and decision-variable definitions follow the pattern of all other models in this chapter, with the provision that, all service, block, and itinerary sets follow the time-space network definition with IN and OUT nodes for each yard. Thus

- f_σ: Fixed selection cost for service $\sigma \in \Sigma$;
- $y_\sigma \in \mathbb{Z}_+$: Frequency of service $\sigma \in \Sigma$. The selection decision and its fixed cost concern, as usual, the complete service definition in the service layer. For representation purposes, they may be associated to the moving arc of the first leg of the service;
- f_b: Fixed cost of building and moving block $b \in \mathscr{B}$;
- $y_b = 1$, if block $b \in \mathscr{B}$ is built, and 0, otherwise. Similarly to service selection, block-selection decision and fixed cost apply to the complete block definition in the block and service layers. For representation purposes, they may be associated to the arc out of the $O(b)$ IN node on the block layer;
- $x_a^k \geq 0$, continuous flow variable representing the volume of commodity $k \in \mathscr{K}$ on arcs $a \in \mathscr{A}$, becoming x_b^k, when $a = b \in \mathscr{B}$, and $x_{l_i(\sigma)}^k$, when $a = l_i(\sigma) \in \mathscr{L}, \sigma \in \Sigma$;
- c_a^k: Hauling and time unit cost for commodity $k \in \mathscr{K}$ on the service legs, i.e., on the moving arcs of the service layer; the handling cost on car-classification (car layer) and block-transfer (block layer) arcs; and the time cost on the holding arcs of the car, block and service layers;
- u_a: Capacity of classification (u_η^{C}, car layer) and transfer arcs (u_η^{B}, block layer).

The integrated scheduled service network design model may be formulated as:

$$\text{Minimize} \quad \sum_{\sigma \in \Sigma} f_\sigma y_\sigma + \sum_{b \in \mathscr{B}} f_b y_b + \sum_{k \in \mathscr{K}} \sum_{a \in \mathscr{A}} c_a^k x_a^k \tag{13.34}$$

Subject to

$$\sum_{a \in \mathscr{A}_\eta^+} x_a^k - \sum_{a \in \mathscr{A}_\eta^-} x_a^k = \begin{cases} d^k, \text{ if } \eta = O(k), \\ -d^k, \text{ if } \eta = D(k), \\ 0, \text{ otherwise}, \ \forall \eta \in \mathscr{N}, k \in \mathscr{K}, \end{cases} \tag{13.35}$$

$$\sum_{k \in \mathscr{K}} x_b^k \leq u_b y_b, \qquad \forall\, b \in \mathscr{B}, \tag{13.36}$$

$$\sum_{k \in \mathscr{K}} \sum_{b \in \mathscr{B}(l_i(\sigma))} x_b^k \leq u_{l_i(\sigma)} y_\sigma, \qquad \forall l_i(\sigma) \in \mathscr{L}, \sigma \in \Sigma, \tag{13.37}$$

$$\sum_{k \in \mathscr{K}} x_a^k \leq u_a^{\mathrm{C}}, \qquad \forall a \in \text{classification arcs} \subset \mathscr{A}, \tag{13.38}$$

$$y_b \leq \delta_\sigma^b y_\sigma, \qquad \forall b \in \mathscr{B}, \sigma \in \Sigma, \tag{13.39}$$

$$\sum_{b \in \mathscr{B}} y_b \leq u_a^{\mathrm{T}}, \qquad \forall a \in \text{transfer arcs} \subset \mathscr{A}, \tag{13.40}$$

$$\sum_{b \in \mathscr{B} \mid O(b) = \eta} y_b \leq u_\eta^{\mathrm{B}}, \qquad \forall \eta \in \text{in nodes on block layer} \subset \mathscr{N}, \tag{13.41}$$

$$\sum_{\sigma \in \Sigma \mid O(\sigma)=\eta} y_\sigma \leq u_\eta^{\mathrm{M}}, \qquad \forall \eta \in \text{in nodes on service layer} \subset \mathcal{N}, \tag{13.42}$$

$$y_\sigma \in \mathbb{Z}_+, \qquad \forall \sigma \in \Sigma, \tag{13.43}$$

$$y_b \in \{0, 1\}, \qquad \forall b \in \mathcal{B}, \tag{13.44}$$

$$x_a^k \geq 0, \qquad \forall a \in \mathcal{A}, k \in \mathcal{K}, \tag{13.45}$$

where the objective function (13.34) computes the total cost of selecting and operating services, building, transferring, and hauling blocks, and classifying, blocking, transferring, and hauling cars. Constraints (13.35) enforce flow conservation at all nodes on all layers. Constraints (13.36) and (13.37) limit the loads of blocks and service legs in terms of cars hauled, respectively, while constraints (13.38) perform the same task on the yard classification arcs on the car layer. Constraints (13.39) link the building of blocks to the selection of the services moving them, while constraints (13.40) limit the number of blocks which can transfer simultaneously at a yard on the block layer. Finally, constraints (13.41) and (13.42) limit the number of blocks and trains, respectively, which can be built at each yard during the schedule length (extending the formulation to enforce capacities by time periods is straightforward).

5 Extending the SSND

The previous model is general and may be extended to account for additional railroad features and planning issues. It may be extended, for example, to path-based models, integrating the handling of non-additive characteristics and penalty or congestion representations of capacity limits. We focus in this section on three extensions, the first handling given service schedules and continuous-time representation, the second addressing the intermodal railroad case, while the third is concerned with the integration of resource-management concerns into tactical SSND.

It is not unusual for tactical railroad planning to be performed by two different teams within the railroad, one focusing on the service design, with somewhat rough blocking concerns, the other starting from the service network selected and focusing on the detailed classification, blocking, and final train makeup decisions. This case is particularly observed when intermodal traffic is concerned. The service design still addresses the entire system and all traffic classes, while only the services dedicated to intermodal freight are within the scope of classification and blocking planning.

A given service network and schedule induces a continuous time discretization of the schedule length, corresponding to the departure and arrival time instants of each service at each of the yards in its route. Figure 13.5 illustrates the service layer of a multi-layer SSND, at a particular yard, for two services stopping at that yard for different lengths of time. The network is greatly simplified, as the only IN and OUT nodes in the layer correspond to the arrival and departure time instances,

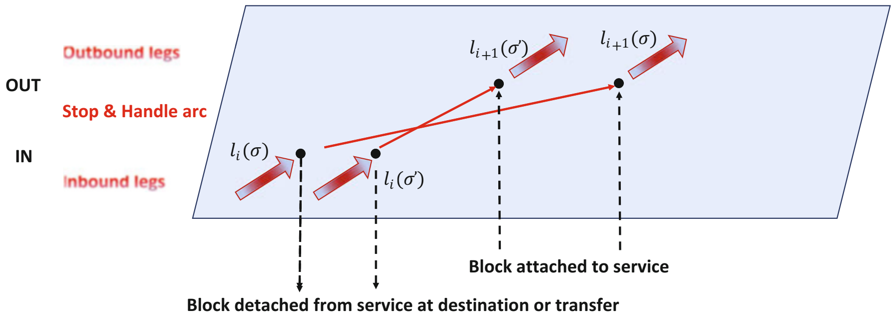


Fig. 13.5 Service layer with given SSND service schedule

respectively, of each service. The block-to-train attach and detach activities are concentrated in the IN and OUT nodes of the service. This defines the time instances in all the other layers. The network is further simplified as the activities related to each service while in the yard are associated to a unique Stop & Handle service-specific arc, the arcs in the other layers being simplified in a similar way. Note that holding arcs capturing waiting periods on the car or block layer are not eliminated. The duration and costs of the inter- and intra-layer arcs may be adjusted to account for this discretization. Thus, for example, when the availability time of demand $k \in \mathscr{K}$ is before the first node defined in the network, $o(k)$ is set to the time instance of that node and the waiting cost and time of the demand are adjusted accordingly. The resulting SSND still presents the same degree of complexity as the general network design problems. The simplification of the network provides the means, however, to address much larger problem dimensions with commercial mixed-integer software, corresponding, for example, to the cases of several North American railroads.

Intermodal traffic and operations are the topic of the second extension of the SSND we discuss (to simplify the presentation, car classification activities are not included). As already indicated, intermodal demand must be loaded onto cars at the origin terminal and must be unloaded at destination. This major difference with regular traffic, induces an additional layer to the SSND network. The container layer corresponds to the entry and exit of container OD demand into and out of the railroad system. Holding arcs capturing waiting prior to blocking or prior to final delivery at destination are part of the container layer. Inter-layer arcs between the container and car layers support the container-to-car assignment and loading, in one direction, and the container unloading, in the opposite direction (when the service network and schedule is given, as above, the time instance of those arcs correspond to a possible block for the cars, which corresponds to a possible service for the block.)

Representing the many types of containers and container-compatible cars, as well as the large number of rules governing the loading of containers on railroad cars is a major challenge for tactical (and strategic) modeling. Indeed, one cannot explicitly include the huge number of feasible loading patterns into aggregated service network design formulations. This is still a challenging research issue. We present an approximation procedure, which proved appropriate for tactical planning in the North American context, where double-stacking is largely used (the approximation may be easily adapted for the simple case of single staking). Double stack means that two containers may be loaded one on top of the other, following very stringent rules, e.g., a 53-feet long container may be positioned on top of a 40-feet container or two 20-feet containers, but the opposite patterns are generally not allowed.

The approximation is based on the observations that (1) cars built to haul containers (also called Well cars) come in several multi-platform configurations, a platform providing two slots, one on the bottom and one on top, for containers of a given length; (2) 40-feet and 53-feet are the two main container types, the former world wide (with the 20-feet ones, which can be approximated as two 20s = one 40), the latter mainly in North America (43-feet and 45-feet may be modeled through the 53s), and correspond to the main platform-loading capability of cars; (3) cars able to carry 53-feet containers are more expensive (to rent and operate) than the other, more regular, cars and one therefore aims to use not more than necessary; (4) *length* is a major constraining feature for trains and blocks both in terminals and when put on trains. Then, given that the length of a car is determined for the most part by the number of platforms it provides, the approximation makes use of the *platform*, of a given type, as a loading unit, and considers 40- and 53-feet long container and platform types.

Consider the basic loading rules for these container and platform types:

- *40-feet platform*: (1) single 40-feet container in the bottom slot; (2) two 40-feet containers in the two slots; (3) one 40-feet container in the bottom slot and one 53-feet in the top slot; 4) empty;
- *53-feet platform*: (1) all the configurations of a 40-feet platform; (2) single 53-feet container in bottom slot; (3) two 53-feet containers in the two slots.

The procedure aims to "maximize" the number of forty-feet platforms per unit of train length. Consequently, when the number of 53-feet containers (nb_{53}) is greater than or equal to the number of 40-feet containers (nb_{40}), the 40s should be placed in bottom slots and the 53s on top, as much as possible. The numbers of 53-feet (nbp_{53}) and 40-feet (nbp_{40}) platforms are given by (13.46) and (13.47), respectively. These four parameters are then the basis for decision-variable definitions for the numbers of containers and platforms, of each type, assigned to each block. The values of these variables are governed by constraints implementing relations (13.46) and (13.47), and are used to compute costs and enforce capacities.

$$nbp_{53} = \max\{0, \lceil (nb_{53} - nb_{40})/2 \rceil\}, \tag{13.46}$$

$$nbp_{40} = \lceil (nb_{53} + nb_{40})/2) \rceil - nbp_{53}. \tag{13.47}$$

Similar to any other transportation mode (Chap. 12), rail services require resources to operate, people, cars, and locomotives, in particular. While a rich literature addresses resources-management issues (Sect. 6), most research and contributions target operational planning issues, in which the service network and schedule are given. Multicommodity network flow optimization (linear formulations with integer flow variables) is the methodology of choice in those cases. Given the scope and length limits of this book, we do not detail this methodology. We rather focus on the challenge of representing resource-management concerns at the level of tactical planning and within SSND models. The goal is not to integrate the details of scheduling and managing resources, but rather to capture the main impacts of resource management on the tactical plan. This is a broad and largely unexplored research area, which needs significant work, not only for railroads, but for consolidation-based transportation in general.

The initial and current developments focus mainly on the availability and routing of material resources (also sometimes called "assets"). They follow from the operational needs and developments aimed at balancing assets, as well as from the growing requirements of efficiently running a scheduled railroad without a level of resources higher than what is strictly needed. With respect to the first aspect, recall that trade and, thus, demand flows are unbalanced, different products and quantities flowing, say, from West to East than vice-versa. This results in vehicles, cars and locomotives, of certain types, becoming available after providing service at yards where they are not needed for the next cycle of operations but missing at others. Empty cars and locomotives must therefore be moved, *repositioned*, for the next cycle. So-called "full-asset utilization" policies illustrate the second aspect, where resources are ideally expected to circulate continuously in the network (accounting for the maintenance requirements, of course) supporting the scheduled services.

The basic translation of the previous discussion into a mathematical formulation are the *design-balancing* constraints

$$\sum_{a \in \mathscr{A}_\eta^+} \sum_{\sigma \in \Sigma} \delta_\sigma^a y_\sigma - \sum_{a \in \mathscr{A}_\eta^-} \sum_{\sigma \in \Sigma} \delta_\sigma^a y_\sigma = 0, \ \forall \eta \in \mathscr{N}, \tag{13.48}$$

where $\delta_\sigma^a = 1$ if service $\sigma \in \Sigma$ operates on arc $a \in \mathscr{A}$ (i.e., one of its legs defines arc a, which terminates or initiates at node η), and 0, otherwise.

The design-balancing constraints (13.48) state that the number of resources brought into a yard by all incoming services equals the number of resources taken out of the yard by the selected outgoing services. The constraints assume that one unit of resource is required by each occurrence of each service (recall that $y_\sigma \in \mathbb{Z}_+$), where the unit may represent a locomotive, a group of locomotives, a car, a group of cars, or a crew. In this sense, resources are assimilated to services. The formulation is still rich, however. Several resource types may be defined, with the corresponding design-balancing constraints. Moreover, the δ_σ^a parameters may be refined and tailored for particular resources and restrictions of services or yards. On the other hand, it is difficult to represent cost and utilization characteristics of particular resource types, such as the assignment of resources to particular home yards, which is a regular feature of transportation systems, and the maximum duration before returning home for maintenance.

A path-based formulation provides the modeling tool to address these shortcomings. The idea is to explicitly define the sequence of tasks a resource has to undertake as a *cycle* θ of service legs, and holding arcs in time-dependent formulations. Let Θ represent the set of resource cycles, and u^{R} the quantity of resources available in the system. Let $\mathscr{L}(\theta)$ be the set of service legs $l_i(\sigma) \in \mathscr{L}(\sigma), \sigma \in \Sigma$, the resource cycle $\theta \in \Theta$ supports in sequence, with the definitional parameter $\delta^{\theta}_{l_i(\sigma)} = 1$ if service leg $l_i(\sigma) \in \mathscr{L}(\sigma), \sigma \in \Sigma$, is supported by resource cycle $\theta \in \Theta$, and 0, otherwise. Let f_θ represent the fixed cost of selecting and operating resources on cycle $\theta \in \Theta$, and let us define the resource selection decision variable $y_\theta \in \mathbb{Z}_+$ as the number of resources executing cycle $\theta \in \Theta$.

The total resource cost $\sum_{\theta\in\Theta} f_\theta y_\theta$ is then added to the objective function of the SSND model. Constraints (13.49) are added to the formulation to connect the selection of the sufficient number of resources and the requirements of the selected services.

$$\sum_{\theta\in\Theta} \delta^{\theta}_{l_i(\sigma)} y_\theta = y_\sigma, \qquad \forall l_i(\sigma) \in \mathscr{L}(\sigma), \sigma \in \Sigma, \tag{13.49}$$

$$\sum_{\theta\in\Theta} y_\theta \leq u^{\mathrm{R}}, \qquad \forall \eta \in \mathscr{N}. \tag{13.50}$$

Notice that constraints (13.50), limiting the number of resources selected to the availability of resources, is not needed as stated, the resource cost driving the number of selected resources to the minimum required to run the system. The constraints may be refined, however, to represent resource availability at each yard, when cycles (i.e., the resources executing them) are linked to a home yard as its respective domicile, from where it originates and where it returns at the end. Notice also that the attributes of resource cycles may be controlled during generation, e.g., one may forbid generating cycles longer than permitted by the rules governing the resource type and its home yard (see also the discussion of Sect. 3.4). As discussed in Chap. 12, this approach is extremely promising for linking resource management and service network design for tactical planning, but much more research is required on modeling the various cases and objectives and on developing efficient solution methods for large problem instances.

6 Bibliographical Notes

The literature on railroads and railroad planning goes many years back, generally presenting application-based contributions and reflecting often industry practice. Several survey papers synthesize the story and contributions of operations research, including network design methodology, to railroad planning, e.g., Assad (1980b);

Dejax and Crainic (1987); Crainic (1988); Crainic and Laporte (1997); Cordeau et al. (1998); Crainic (2000); Newman et al. (2002); Crainic (2003); Ahuja et al. (2005a); Crainic and Kim (2007); Bektaş and Crainic (2008); Crainic (2009); Yaghini and Akhavan (2012).

Early contributions focus on single problems or combinations of a limited number of issues. These include the pioneering service selection, routing and makeup model of Assad (1980a), the train routing and the scheduling model of Morlok and Peterson (1970). Huntley et al. (1995) developed a computerized routing and scheduling system for CSX Transportation, while Ireland et al. (2004) developed a planning system for Canadian Pacific Railway that brought together several separate procedures without building a comprehensive model.

Blocking has often been addressed as a separate problem to be solved before the selection of services. Bodin et al. (1980) proposed one of the first such models, a non-linear mixed-integer formulation, blocking delays being dependent on the number of cars assigned to each block. Newton (1996); Newton et al. (1998); Barnhart et al. (2000) formulates the blocking problem as a network design model, arcs representing candidate blocks among classification yards. No fixed costs are associated to blocks, the number of blocks which can be build at each yard being limited through budget constraints. A path-formulation and a branch-and-price algorithm (Barnhart et al. 1998) are proposed in the first two contributions, while a dual-based Lagrangian relaxation is used in the latter to decompose the problem into easier-to-address subproblems, namely a continuous multicommodity flow problem and an integer block formulation that selects blocks satisfying yard capacity constraints (addressed by a branch-and-cut algorithm). Ahuja et al. (2007) follows the same approach in an arc-based formulation, proposing a large neighborhood search algorithm aimed at addressing large problem instances. Jha et al. (2008) then proposes arc and path-based time-space formulations for the block-to-train assignment problem. The latter formulation proved the most flexible and amenable to be efficiently solved either with an *a priori* set of paths, or a dynamic-path generation procedure. Metaheuristics for the arc or path-based formulations are proposed by, e.g., Yaghini et al. (2011, 2012); Yue et al. (2011). Uncertainty has been rarely addressed in models targeting freight railroad planning. A few contributions addressing blocking problems have been proposed (e.g., Yang et al. 2011; Hasany and Shafahi 2017), but much more research is required in this area.

Service selection was also often treated separately of the other planning problems (Assad 1980a; Morlok and Peterson 1970; Martinelli and Teng 1996; Yaghini et al. 2014). It has also been addressed in two steps, service routes and frequencies being determined first (e.g., Marín and Salmerón 1996a,b; Goossens et al. 2004), the schedule being constructed in a second step, based on the routing patterns yielded by the first step (e.g., Nozick and Morlok 1997; Brännlund et al. 1998; Caprara et al. 2002, 2006; Cacchiani et al. 2010; Cacchiani and Toth 2012).

Models aiming for integration of tactical planning issues were proposed simultaneously with those targeting individual issues described above. Crainic et al. (1984) presents what is probably the first service network design model addressing simultaneously the selection of services and their frequencies, car classification and blocking, train makeup, and freight routing. It is noteworthy that the model integrates the distribution of empty cars through one or several origin-destination demand matrices (generated through demand-distribution models from the surplus and penury levels at yards, which were derived from the loaded car demand). These matrices become commodities to be handled simultaneously with all other OD commodities in the problem. The model takes the form of a static path-based, nonlinear network design formulation accounting for congestion and accumulation-delay phenomena in yards and on rail tracks, service-quality targets, and trade-offs between operating and time-related costs. Block fixed costs were not included; they were approximated through the accumulation-delay costs and the limits on yard-specific block dimensions. A heuristic solution method was used to address realistically-sized problem instances derived from the case of a large North-American railroad.

Crainic and Rousseau (1986) generalizes the model for the tactical planning of consolidation-based multicommodity multimode freight transportation systems. Bektaş et al. (2010) later studied Lagrangean-based relaxation and decomposition algorithms. The authors show that, first, non-linearities may be handled efficiently through decomposition and, second, that the relaxation of the flow constraints, which yields an arc decomposition, has computationally better convergence properties than the dualization of the capacity constraints. These results are very encouraging for this demanding but important research topic.

A number of contributions followed toward the end of the 80; and during the 90's. Haghani (1989) presents a model which attempts to combine train routing and scheduling, make-up, as well as empty car distribution on a space-time network with fixed travel times and pre-specified traffic rules. A heuristic is used to address a somewhat simplified version of the model and illustrate the interest of integrated planning. The model proposed by Keaton (1989, 1992) aims to determine the pairs of yards to connect by direct services, and whether to offer more than one train a day, as well as the routing of freight and the blocking of rail cars. The service network is made up of one network for each pair of yards in the system with positive demand. Arcs represent trains and connections in yards, as well as *a priori* determined blocking alternatives. Gorman (1998) starts from the previous model aiming to design a scheduled operating plan that followed as much as possible the particular operation rules of a given railroad. An innovative tabu-enhanced genetic search metaheuristic is used to generate candidate train schedules, which are evaluated on their economic, service, and operational performances. On relatively small but realistic problems, the metaheuristic performed well and was used for strategic scenario analysis for a major North-American railroad. All these contributions model blocking through classification costs, rather than explicit blocking decision variables.

Zhu et al. (2014) propose a cyclic multi-layer time-space SSND model, which appears to be the first comprehensive formulation to select the train services and schedules to operate for a given schedule length, the car classification policies, the blocks to build in each terminal with their routes within the service network, the train makeup, and the demand itineraries using these services and blocks. The authors also introduce a matheuristic solution methodology combining slope scaling, a dynamic block-generation mechanism, long-term memory-based perturbation strategies, and an ellipsoidal search, i.e., a new intensification mechanism to thoroughly explore very large neighborhoods of elite solutions in an efficient way using information from the history of the search. Experimental results show that the proposed solution method is efficient and robust, yielding high-quality solutions for realistically-sized problem instances. The model of Sect. 4 is based on this work.

As already mentioned, the management of resources, or assets, has a long history of research and applications, yielding a rich corpus of literature, starting with the pioneering work on empty cars and containers (Bomberault and White 1966; White 1968; White and Bomberault 1969; White 1972) and locomotives (Florian et al. 1976). Dejax and Crainic (1987); Cordeau et al. (1998); Piu and Speranza (2014) present detailed surveys and syntheses of the literature until the end of the 80's. Most of this literature and developments address operational planning issues, e.g., distribution and routing. Network flow optimization is the methodology of choice in this field, evolving from the initial transportation problem models to the contemporary integer-flow time-space multicommodity formulations integrating various practical rules and constraints (e.g., Ahuja et al. 2005b; Vaidyanathan et al. 2008b,a; Balakrishnan et al. 2016; Bouzaïene-Ayari et al. 2016; Piu et al. 2015; Ortiz-Astorquiza et al. 2021; Miranda et al. 2020).

Few contributions aimed until rather recently to integrate resource management concerns into tactical planning service network design models. We mentioned the modeling of empty cars as an additional demand proposed by Crainic et al. (1984). Close to the network design methodology, Joborn et al. (2004) proposes a time-space formulation to select kernel paths to move groups of empty cars between pairs of yards by using the residual capacity of a given set of scheduled services. A kernel path corresponds to a sequence of services, which can move the group of cars between its origin and destination, plus waiting and inventory arcs. A particular characteristic of the formulation is that fixed costs and capacity constraints are not associated to the design arcs of the network (services), but rather to the kernel path, that is, to a set of design arcs, which increases the difficulty to solve it. A tabu search metaheuristic was proposed to efficiently address the problem and to show that the proposed model achieves the looked-for economies of scale.

Resource-management considerations were integrated into service network design models through the contributions of Andersen et al. (2009a,b); Pedersen and Crainic (2007); Pedersen et al. (2009) (see also Andersen and Christiansen 2009, where the modeling framework of Crainic et al. (1984) is used for the strategic analysis of a new intermodal service in Europe). Pedersen and Crainic (2007); Pedersen et al. (2009) focus on the management of one asset type, namely, locomotives. The authors introduce the concept of design-balanced SND and present a tabu search metaheuristic to address it (see also Vu et al. 2013; Chouman and Crainic 2015, for metaheuristics targeting the same problem). Andersen et al. (2009a,b) enlarges the scope of the models to include resource cycles, cyclic schedules and the coordination/synchronization of several railroads and navigation services at particular junction points. The authors also show that cycle-based formulations provided more modeling flexibility and computational efficiency. A branch-and-price algorithm is proposed by Andersen et al. (2011) for the cycle-based SSND formulation. It is also noteworthy that the papers mentioned in this paragraph also offer insights into modeling tightly time-constrained systems, as well as rail and rail-road intermodal terminals. Car classification and blocking issues were not addressed, however, nor the train makeup problem, or the case of multiple resource types with complex resource-to-service assignment rules. Integrating resource management and service network design for tactical and strategic planning is still a very active, important, and challenging research area.

We complete this brief literature survey with the case of planning intermodal railroad transport. As indicated previously, intermodality presents additional challenges. In particular, the assignment and loading of containers to cars must be explicitly integrated into the planning methods, while accounting for the multiple and complex loading rules (see, e.g., Mantovani et al. 2017, for a detailed description of the complex rules governing the loading of containers on rail cars of diverse characteristics, particularly when double stacking is performed). An additional layer to the SSND time-space network illustrates this additional complexity. Yet, one finds few contributions targeting rail intermodal transport. Newman and Yano (2000) proposes a day-of-week uncapacitated (in the number of trains one may make up in a yard and operate on a line) train scheduling model, to determine whether intermodal OD demand should be moved by a direct or indirect, through a main yard, service. A decomposition method yielding simpler problem settings provides encouraging results. Morganti et al. (2020) proposes a blocking SSND model for intermodal services, when the service network and schedule are given, which determines container-to-car assignments and loading, blocking, service makeup, and demand itineraries. The model of Sect. 5 is inspired by this paper. Very good experimental results data from a large North American railroad were obtained using a well-known commercial software. Much research work is still needed in this area. Two directions in particular. First, integrate resource management concerns (see Kienzle et al. 2021, for very encouraging developments) and service selection decisions. Second, similar to all the other facets of research on railroad planning, algorithmic developments are needed to address efficiently large problem instances.

7 Conclusions and Perspectives

Rail transportation is very important in economic and environmental terms. Its many benefits follow, however, from a complex organization with several levels of consolidation, e.g., freight into cars, cars into blocks, and blocks into trains. Network design, through its service network design formulations with or without explicit schedules, offers models and methods to address these challenges and efficiently support the planning of freight railroad operations to achieve economic and service-quality objectives. Rail is actually more complex than most other consolidation-based transportation modes and, thus, challenges both the modeling and algorithmic facets of operations research, in general, and network design, in particular.

This chapter presents these issues, challenges, and contributions. It illustrates the long and successful history of the connections between rail tactical planning and operations research development. This is a vibrant research area, in continuous development based on the mutually beneficial interactions between new realities in the field and new methodological developments.

Many research perspectives have been identified during the presentation, particularly in Sect. 6. We do not repeat them here. We recall the challenges of continuing to study the integration of the main components of railroad planning at the tactical level, from scheduled service selection to resource management. Among the other interesting and challenging research directions, we single out two. First, the study and explicit integration of uncertainty into the planning models. Uncertainty may be found in demand (e.g., volume, realization of temporal characteristics, etc.) as well as supply in terms of travel and yard-activity times. How one predicts these elements, both at the level of day-to-day operation and as more rare but disturbing incidents, and how one integrates them, and the options to alleviate their negative effects, into SSND models constitutes a significant research challenge. Second, revenue management starts to interest freight railroads. There is still little literature on revenue management in freight transportation (air cargo is somewhat of an exception) and even less when rail is concerned. Research is needed in the revenue mechanisms as applied to rail transport, as well as in the interaction between planning and these mechanisms.

We conclude recalling the challenge of efficient solution methods for large problem instances. Algorithms are required for multi-layer time-space networks, in both their linear and non-linear incarnations. Decomposition methods and parallel optimization offer one interesting avenue for development. Dynamic generation of paths – services, blocks, demand itineraries, resource cycles –, or of the time-space network, or a combination of both, offer an equally interesting complimentary avenue.

References

Ahuja, R. K., Cunha, C. B., & Şahin, G. (2005a). Network models in railroad planning and scheduling. In *Tutorials in Operations Research INFORMS 2005*, INFORMS (pp. 54–101), Published online: 14 Oct 2014.

Ahuja, R. K., Jha, K. C., & Liu, J. (2007). Solving real-life railroad blocking problems. *Interfaces, 37*, 404–419.

Ahuja, R. K., Liu, J., Orlin, J. B., Sharma, L. A., & Dand, S. (2005b). Solving real-life locomotive-scheduling problems. *Transportation Science, 39*(4), 503–517.

Andersen, J., & Christiansen, M. (2009) Designing new European rail freight services. *Journal of the Operational Research Society, 60*, 348–360.

Andersen, J., Crainic, T. G., & Christiansen, M. (2009a). Service network design with asset management: Formulations and comparative analyzes. *Transportation Research Part C: Emerging Technologies, 17*(2), 197–207.

Andersen, J., Christiansen, M., Crainic, T. G., & Grønhaug, R. (2011). Branch-and-price for service network design with asset management constraints. *Transportation Science, 46*(1), 33–49.

Andersen, J., Crainic, T. G., & Christiansen, M. (2009b). Service network design with management and coordination of multiple fleets. *European Journal of Operational Research, 193*(2), 377–389.

Assad, A. A. (1980a). Modelling of rail networks: toward a routing/makeup model. *Transportation Research Part B: Methodological, 14*, 101–114.

Assad, A. A. (1980b). Models for rail transportation. *Transportation Research Part A: Policy and Practice, 14*, 205–220.

Balakrishnan, A., Kuo, A., & Si, X. (2016). Real-time decision support for crew assignment in double-ended districts for U.S. freight railways. *Transportation Science, 50*(4), 1139–1393.

Barnhart, C., Jin, H., & Vance, P. H. (2000). Railroad blocking: A network design application. *Operations Research, 48*(4), 603–614.

Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W. F., & Vance, P. H. (1998). Branch-and-price: column generation for solving huge integer programs. *Operations Research, 46*(3), 316–329.

Bektaş, T., Chouman, M., & Crainic, T. G. (2010). Lagrangean-based decomposition algorithms for multicommodity network design with penalized constraintsm. *Networks, 55*(3), 272–280.

Bektaş, T., & Crainic, T. G. (2008). A brief overview of intermodal transportation. In G. D. Taylor (Ed.), *Logistics engineering handbook* (Chap. 28, pp. 1–16). Boca Raton, FL: Taylor and Francis Group.

Bodin, L. D., Golden, B. L., Schuster, A. D., & Romig, W. (1980). A model for the blocking of trains. *Transportation Research Part B: Methodological, 14*(1), 115–120.

Bomberault, A. M., & White, W. W. (1966). Scheduling empty box cars. Technical Report. IBM New York Scientific Center, Hawthorne, N.Y.

Bouzaïene-Ayari, B., Cheng, C., Das, S., Fiorillo, R., & Powell, W. B. (2016). From single commodity to multiattribute models for locomotive optimization: A comparison of optimal integer programming and approximate dynamic programming. *Transportation Science, 50*(2), 366–389.

Brännlund, U., Lindberg, P. O., Nõu, A., & Nielsson, J. E. (1998). Railway timetabling using lagrangian relaxation. *Transportation Science, 32*(4), 358–369.

Cacchiani, V., Caprara, A., & Toth, P. (2010). Scheduling extra freight trains on railway networks. *Transportation Research Part B: Methodological, 44*(2), 215–231.

Caprara, A., Fischetti, M., & Toth, P. (2002). Modeling and solving the train timetabling problem. *Operations Research, 50*(5), 851–861.

Caprara, A., Monaci, M., Toth, P., & Guida, P. L. (2006). A Lagrangian heuristic algorithm for a real-world train timetabling problem. *Discrete Applied Mathematics 154*, 738–753.

Cacchiani, V. & Toth, P. (2012). Nominal and robust train timetabling problems. *European Journal of Operational Research, 2019*, 727–737.

Chouman, M., & Crainic, T. G. (2015). Cutting-plane matheuristic for service network design with design-balanced requirements. *Transportation Science, 49*(1), 99–113.

Cordeau, J. F., Toth, P., & Vigo, D. (1998). A survey of optimization models for train routing and scheduling. *Transportation Science, 32*(4), 380–404.

Crainic, T. G. (1988). Rail tactical planning: issues, models and tools. In L. Bianco & A. La Bella (Eds.) *Freight Transport Planning and Logistics* (pp. 463–509). Berlin: Springer.

Crainic, T. G. (2000). Network design in freight transportation. *European Journal of Operational Research, 122*(2), 272–288.

Crainic, T. G. (2003). Long-Haul freight transportation. In R. W. Hall (Ed.), *Handbook of Transportation Science* (2nd edn., pp. 451–516). Norwell, MA: Kluwer Academic Publishers.

Crainic, T. G. (2009) Service design models for rail intermodal transportation. In L. Bertazzi, M. G. Speranza, & J. A. E. E. van Nunen (Eds.), *Lecture Notes in Economics and Mathematical Systems* (Vol. 619, pp. 53–67). Berlin: Springer.

Crainic, T. G., Ferland, J. A., & Rousseau, J. M. (1984). A tactical planning model for rail freight transportation. *Transportation Science, 18*(2), 165–184.

Crainic, T. G., Kim, K. H. (2007). Intermodal transportation. In C. Barnhart & G. Laporte (Eds.), *Transportation, Handbooks in Operations Research and Management Science* (Vol. 14, Chap. 8, pp 467–537). Amsterdam: North-Holland.

Crainic, T. G., & Laporte, G. (1997). Planning models for freight transportation. *European Journal of Operational Research, 97*(3), 409–438.

Crainic, T. G., & Rousseau, J. M. (1986). Multicommodity, multimode freight transportation: A general modeling and algorithmic framework for the service network design problem. *Transportation Research Part B: Methodological, 20*, 225–242.

Dejax, P. J., & Crainic, T. G. (1987). A review of empty flows and fleet management models in freight transportation. *Transportation Science, 21*(4), 227–247.

Florian, M., Bushell, G., Ferland, J., Guertin, G., & Nastansky, L. (1976). The engine scheduling problem in a railway network. *INFOR, 14*, 121–138.

Goossens, J. W., van Hoesel, S., & Kroon, L. (2004). A branch-and-cut approach for solving railway line-planning problems. *Transportation Science, 38*(3), 379–393.

Gorman, M. F. (1998). An application of genetic and tabu searches to the freight railroad operating plan problem. *Annals of Operations Research 78*, 51–69.

Haghani, A. E. (1989). Formulation and solution of combined train routing and makeup, and empty car distribution model. *Transportation Research Part B: Methodological, 23*(6), 433–452.

Hasany, R. M., & Shafahi, Y. (2017). Two-stage stochastic programming for the railroad blocking problem with uncertain demand and supply resources. *Computers & Industrial Engineering, 106*, 275–286.

Huntley, C. L., Brown, D. E., Sappington, D. E., & Markowicz, B. P. (1995). Freight routing and scheduling at CSX transportation. *Interfaces, 25*(3), 58–71.

Ireland, P., Case, R., Fallis, J., Van Dyke, C., Kuehn, J., & Meketon, M. (2004). The Canadian Pacific Railway transforms operations by using models to develop its operating plans. *Interfaces, 34*(1), 5–14.

Jha, K. C., Ahuja, R. K., & Şahin, G. (2008). New approaches for solving the block-to-train assignment problem. *Networks, 51*(1), 48–62.

Joborn, M., Crainic, T. G., Gendreau, M., Holmberg, K., & Lundgren, J. T. (2004). Economies of scale in empty freight car distribution in scheduled railways. *Transportation Science, 38*(2), 459–464.

Keaton, M. H. (1989). Designing optimal railroad operating plans: lagrangian relaxation and heuristic approaches. *Transportation Research Part B: Methodological, 23*(6), 415–431.

Keaton, M. H. (1992). The impact of train timetables on average car time in rail classification Yards. *Journal of the Transportation Research Forum, 32*(2), 345–354.

Kienzle, J., Crainic, T. G., Frejinger, E., & Bisaillon, S. (2021). The intermodal railroad blocking & railcar fleet management planning problem. Technical Report. CIRRELT-2021, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et les transports, Université de Montréal, Montréal, QC, Canada

Mantovani, S., Morganti, G., Umang, N., Crainic, T. G., Frejinger, E., & Larsen, E. (2017). The load planning problem for double-stack intermodal trains. *European Journal of Operational Research, 267*(1), 107–119.

Marín, A., & Salmerón, J. (1996a). Tactical planning of rail freight networks. Part I: Exact and heuristic methods. *European Journal of Operational Research, 90*, 26–44.

Marín, A., & Salmerón, J. (1996b). Tactical planning of rail freight networks. Part II: local search methods with statistical analysis. *European Journal of Operational Research, 94*, 43–53.

Martinelli, D. R., & Teng, H. (1996). Optimization of railway operations using neural networks. *Transportation Research Part C: Emerging Technologies, 4C*(1), 33–49.

Miranda, P., Cordeau, J. F., & Frejinger, E. (2020). A time-space formulation for the locomotive routing problem at the Canadian National Railways. Technical Report. CIRRELT-2020-19, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et les transports, Université de Montréal, Montréal, QC, Canada

Morganti, G., Crainic, T. G., Frejinger, E., & Ricciardi, N. (2020). Block planning for intermodal rail: methodology and case study. *Transportation Research Procedia, 47*, 19–26.

Morlok, E. K., & Peterson, R. B. (1970). A final report on a development of a geographic transportation network generation and evaluation model. In *Proceedings of the Eleventh Annual Meeting, Transportation Research Forum* (pp. 99–103)

Newman, A. M., Nozick, L. K., & Yano, C. A. (2002). Optimization in the rail industry. In P. M. Pardalos & M. G. C. Resende (Eds.), *Handbook of Applied Optimization* (pp. 704–718), New York, NY: Oxford University Press.

Newman, A. M., & Yano, C. A. (2000). Centralized and decentralized train scheduling for intermodal operations. *IIE Transactions, 32*(1), 743–754.

Newton, H. N. (1996). Network design under budget constraints with application to the railroad blocking problem. Ph.D. Thesis. Industrial and Systems Engineering, Auburn University, Auburn, Alabama, U.S.A.

Newton, H. N., Barnhart, C., & Vance, P. H. (1998). Constructing railroad blocking plans to minimize handling costs. *Transportation Science, 32*(4), 330–345.

Nozick, L. K., & Morlok, E. K. (1997). A model for medium-term operations planning in an intermodal rail-truck service. *Transportation Research Part A: Policy and Practice, 31*(2), 91–108.

Ortiz-Astorquiza, C., Cordeau, J. F., & Frejinger, E. (2021). The locomotive assignment problem with distributed power at the Canadian National Railway Company. *Transportation Science, 55*(2), 510–531.

Pedersen, M. B., & Crainic, T. G. (2007). Optimization of intermodal freight service schedules on train canals. Publication CIRRELT-2007-51, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Montréal, QC, Canada.

Pedersen, M. B., Crainic, T. G., & Madsen, O. B. G. (2009). Models and tabu search metaheuristics for service network design with asset-balance requirements. *Transportation Science, 43*(2), 158–177.

Piu, F., Première Kumar, V., Bierlaire, M., & Speranza, M. G. (2015). Introducing a preliminary consists selection in the locomotive assignment problem. *Transportation Research Part E: Logistics and Transportation Review, 82*, 214–237.

Piu, F., & Speranza, M. G. (2014). The locomotive assignment problem: A survey on optimization models. *International Transactions in Operational Research, 21*(3), 327–352.

Vaidyanathan, B., Ahuja, R. K., Liu, J., & Shughart, L. A. (2008a). Real-life locomotive planning: new formulations and computational results. *Transportation Research Part B: Methodological, 42*(2), 147–168.

Vaidyanathan, B., Ahuja, R. K., & Orlin, J. B. (2008b). The locomotive routing problem. *Transportation Science, 42*(4), 492–507.

Vu, D. M., Crainic, T. G., & Toulouse, M. (2013). A three-stage matheuristic for the capacitated multi-commodity fixed-cost network design with design-balance constraints. *Journal of Heuristics, 19*, 757–795.

White, W. W. (1968). A program for empty freight car allocation. Technical Report. 360D.29.002, IBM Contributed Program Library, IBM Corporation, Program Information Department, Hawthorne, N.Y.

White, W. W. (1972). Dynamic transshipment networks: an algorithm and its application to the distribution of empty containers. *Networks, 2*(3), 211–236.

White, W. W., & Bomberault, A. M. (1969). A network algorithm for empty freight car allocation. *IBM Systems Journal, 8*(2), 147–171.

Yaghini, M., & Akhavan, R. (2012). Multicommodity network design problem in rail freight transportation planning. *Procedia Social and Behavioral Sciences, 43*, 728–739.

Yaghini, M., Momeni, M., & Sarmadi, M. (2014). Solving train formation problem using simulated annealing algorithm in a simplex framework. *Journal of Advanced Transportation, 48*, 402–416.

Yaghini, M., Seyedabadi, M., & Khoshraftar, M. M. (2011). Solving railroad blocking problem using ant colony optimization algorithm. *Applied Mathematical Modelling, 35*(12), 5579–5591.

Yaghini, M., Seyedabadi, M., & Khoshraftar, M. M. (2012). A population-based algorithm for the railroad blocking problem. *Journal of Industrial Engineering International, 8*(8), 1–11.

Yang, L., Gao, Z., & Li, K. (2011). Railway freight transportation planning with mixed uncertainty of randomness and fuzziness. *Applied Soft Computing 11*, 778–792.

Yue, Y., Zhou, L., Yue, Q., & Fan, Z. (2011). Multi-route railroad blocking problem by improved model and ant colony algorithm in real world. *Computers & Industrial Engineering, 60*, 34–42.

Zhu, E., Crainic, T. G., & Gendreau, M. (2014). Scheduled service network design for freight rail transportations. *Operations Research 62*(2), 383–400.

Chapter 14
Motor Carrier Service Network Design

Ilke Bakir, Alan Erera, and Martin Savelsbergh

1 Introduction

The trucking, or motor freight, industry provides ground freight transportation services to shippers using road trucks. Motor carriers provide multiple types of services, differentiated to serve shipments with different characteristics. *Truckload* services are offered to shippers who move dedicated trailers or containers each directly from an origin location to a destination location. Truckload services are provided both by large firms with thousands of tractors and trailers but also by small companies that may sometimes operate fleets with only a few vehicles. In contrast, *consolidation trucking carriers* operate both a network of freight transfer terminals and also a fleet of vehicles to provide a schedule of transportation services for shippers moving smaller quantities. There are two primary consolidation service types. *Less-than-truckload* (LTL), or freight, services provide shippers with the capability to send smaller shipments that do not require an entire trailer; an LTL carrier consolidates shipments into truckload movements between terminals to provide cost-effective service. *Package* services serve shippers seeking to move the smallest shipments, typically letters, small parcels, and boxes.

Truck transportation in most countries is currently the dominant land transportation mode, accounting for the largest fraction of revenue and moving the most tons. For example, in the United States in 2016 trucking accounted for 63% of

I. Bakir
Department of Operations, Faculty of Economics and Business, University of Groningen, Groningen, Netherlands
e-mail: i.bakir@rug.nl

A. Erera (✉) · M. Savelsbergh
H. Milton Stewart School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, USA
e-mail: alan.erera@isye.gatech.edu; martin.savelsbergh@isye.gatech.edu

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_14

the total tonnage moved and 62% of the total value of all shipments (Bureau of Transportation Statistics 2018); in the European Union and in Asia, motor freight is similarly important. Trucking services provide fast *transit times* to shippers with only air freight able to provide shorter times. Transit time is a primary measure of *customer service level* in shipping, and many modern freight services guarantee transit times to shippers.

This chapter focuses on service network design for consolidation trucking carriers. Optimization models and solution approaches for the core network design problems, which include flow and load planning, will be covered in detail. Other related operations design problems will be discussed more briefly. The goal is to provide a thorough introduction to these problems and methods and to focus the discussion on ideas that have had an impact on practice. The chapter will also highlight some of the newest work in this area and help guide researchers beginning work in this field.

The remainder of the chapter is organized as follows. Section 2 provides an overview of trucking operations, focusing specifically on the structure of the networks operated by consolidation trucking carriers. Section 3 introduces models for trucking network flow planning and describes exact and heuristic approaches for building solutions to these models. Section 4 then describes integrated flow and load planning models that rely on time-expanded networks and describes large-scale local search heuristics for their solution. Section 5 briefly describes key developments in the trucking service network design literature. Finally, Sect. 6 provides some perspective on the current state of this research area and discusses a number of ongoing research trends that hold promise for this field.

2 Consolidation Trucking Operations

Consolidation trucking carriers plan and operate service networks to provide freight transportation services directly to shippers seeking to move less-than-truckload or package freight. Carriers establish a geographic region within which they will operate, and more specifically determine origin and destination pairs between which they will provide service and for which categories (or *classes*) of freight. Each service offering for an origin-destination pair also includes a price (or freight *rate*) and a transit time. In some cases, transit times provide only a rough estimate of the number of days required for the execution of the transportation service, while in other cases time-definite offerings specify precisely how long a shipment will require and when it will arrive (for example, 2-day or next-morning).

Given a set of service offerings, a consolidation trucking carrier must build and operate a service network to satisfy customer demand feasibly and cost-effectively. To do so, a medium to large carrier operates a network of transfer *terminals*. Trucking terminals have facilities for truck loading and unloading; these *docks* enable rear-loading trucks to park with the trailer deck at the same level of the terminal floor. In LTL operations, the truck trailers that are used to make pickups and deliveries at customer locations are similar or identical to those used for the long distance terminal-to-terminal movements. In package operations, smaller delivery vans are used when visiting customers, and terminals therefore may have different loading areas for different truck types.

All terminals have the capability to *sort* freight shipments to be loaded into different outbound truck trailers or containers. *Cross-dock sorting*, or *cross-docking*, is a sorting system where larger shipments (often on pallets or within intermediate containers) are moved from unloading trailers to loading trailers by forklift or pallet jack. The name cross-docking refers to the fact that shipments are moved directly "across the dock" and are not stored in any intermediate locations. Cross-docking is the primary sorting operation used by LTL carriers. Since parcel shipments are smaller, terminals operated by package carriers typically include automated and/or manual piece-sorting equipment. Examples of piece-sorting equipment include cross-belt sorters or manual sorting cabinets. Smaller packages, parcels, and letters may be consolidated into bags or other types of intermediate containers before they are loaded into trailers. Package carriers may also use conveyor belt systems and additional belt sorters to enable movement of parcels through the terminal as well as to facilitate cross-docking of larger parcels, bags, and intermediate containers.

Thus, the primary role of terminals in trucking networks is consolidation of smaller shipments into truckloads and the related transfer of shipments between inbound and outbound trailers and containers. Consolidation and transfer allows a trucking carrier to provide cost-effective service between large numbers of origins and destinations. For example, a carrier that operates n terminals with direct service between all pairs would need to move trailers on $n(n-1)$ service lanes, but if one of the central locations were used as a transfer hub this number could be reduced to as few as $2(n-1)$ lanes. Individual truckload dispatches in a well-designed consolidation network will have higher trailer utilization and the total required trailer-miles required to move freight from origins to destinations should decrease. However, each individual shipment may travel farther (thus increasing system ton-miles) and may be sorted one or more times at intermediate transfer terminals.

A typical consolidation terminal network is depicted in Fig. 14.1, in this case for an LTL carrier. Carriers typically operate two types of terminals. An *end-of-line*, or satellite, terminal is a smaller facility that only enables transfer of freight between the *pickup-and-delivery* operation and the *linehaul* operation. A *hub*, or breakbulk, terminal is a larger facility that provides both the functionality of an end-of-line terminal while also providing transfer opportunities between terminals in the linehaul network. Each end-of-line terminal may be connected with dispatches to and from only a small set of hub terminals, while hub terminals may provide dispatches to and from a large number of end-of-line terminals and other hubs. Package networks have similar designs.

Effective freight transfer also requires timed coordination of unloading, sorting, and loading activities at transfer terminals. For this reason many carriers divide each operating day into distinct *sorting periods* or, more simply, *sorts*. Trailers arriving for a sort are unloaded and the freight shipments are sorted into outbound trailers for dispatch by the conclusion of the sort. It is quite common for terminals to operate a

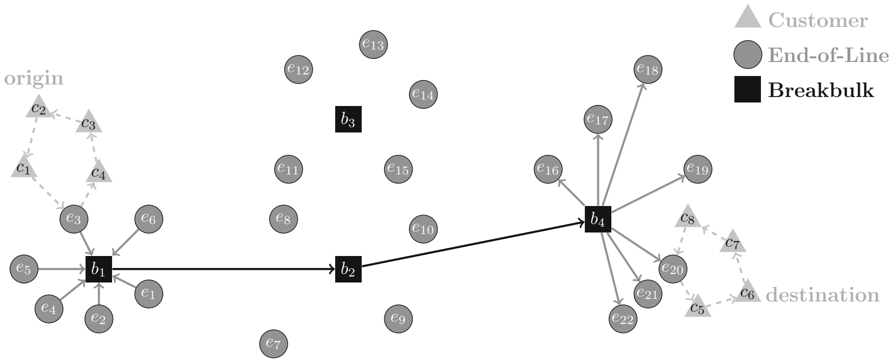


Fig. 14.1 A network of an LTL trucking carrier, serving customer locations using end-of-line and breakbulk cross-dock terminals

morning sort and an evening sort, while larger transfer terminals may also operate additional overnight and midday sorting periods.

Given a network of terminals, a consolidation trucking carrier will operate a pickup-and-delivery operation and linehaul operation. The pickup-and-delivery operation is used to collect freight from customers and transport it to an origin consolidation terminal and to distribute freight for the last-mile from the final terminal to destinations. In package systems, the vehicles used for pickup and delivery are usually smaller delivery vans while LTL carriers often use short or medium length trailers. Pickup and delivery operations require the execution of multi-stop routes with significant time constraints; customer facilities have time windows when they can send or receive shipments, and the carrier has deadlines when freight must leave from or arrive to a terminal in order to meet the service expectation of the customer. It should be noted that high-volume shippers often interface with consolidation carriers by using *drop shipping* or by supplying dedicated customer trailer equipment. In these scenarios, the shipper or the carrier may move truckloads of shipments directly into the linehaul network at a carrier terminal thus skipping the traditional pickup process.

A carrier linehaul network operation provides transportation of truck trailers and containers between its transfer terminals. A *load* in an LTL or package network refers to a trailer or a container that is loaded and dispatched from an origin terminal and headed for unloading at a destination terminal. For LTL carriers, loads are most frequently moved by company truck drivers using either long trailers or trains of two or three short trailers. Short trailers, such as 28-foot pups in the US, provide the carrier with the ability to have single drivers move multiple loads with smaller individual volumes simultaneously. Linehaul movements of loads between terminals, or *dispatches*, tend to be short to allow drivers to return to their home terminals within a single operating day. When loads are created between distant terminals, they frequently are not moved directly by a single driver. Instead, the load may be transferred using two or more *movements*, where each movement is executed either by a company driver or by an outside contractor. The intermediate stops in these sequences of movements are typically called *relays*, and they may occur at terminals or at dedicated relay facilities; operating in this way can both speed the movement of long distance loads while also eliminating the need for some intermediate sorting. For certain long distance loads, freight railroads may be used to move trailers or rail-compatible containers in an outsourcing arrangement; such rail *intermodal* movements are less costly but require longer travel times and may introduce more travel time uncertainty.

2.1 Trucking Service Network Design Problems

In consolidation trucking systems, we refer to the service network as the set of transportation and supporting activities operated by a trucking carrier in order to provide transportation services to shippers. If we think of a network using its general definition as a set of interacting components, then a service network operated by a trucking firm refers to truck transportation movements and associated loading, sorting, and unloading activities. *Service network design* problems in truck transportation focus on building designs and operational plans for these networks; see Crainic (2000) for a comprehensive review of earlier work in all areas of freight transportation. Typically, *physical network* design questions such as determining the type, number, and size of terminal facilities to operate are not considered service network design problems. There is a significant body of literature in facility location (see e.g., Love et al. 1988; Mirchandani and Francis 1990; Drezner and Hamacher 2001; Snyder 2006; Daskin 2011) and a subset that focuses specifically on the location of truck transportation terminals known as hub location problems (see e.g., O'Kelly 1986; Campbell 1994; Alumur and Kara 2008; Farahani et al. 2013), and thus we will ignore these problems in this chapter.

Since trucking service networks can be complex and require many design and planning decisions, a large number of problems could be classified as service network design problems including:

- flow planning problems;
- load planning, routing, and dispatch problems;
- driver and equipment fleet management problems; and
- vehicle routing and scheduling problems.

Flow planning, or freight routing, problems seek to determine how shipments should flow, or be routed, through a terminal network en route from origin to destination. A freight route for an individual shipment specifies the sequence of terminals where the shipment will be transferred via cross-docking or other sorting methods; in most cases, a shipment is unloaded from one truck and reloaded onto another at each of these terminal stops. While it is possible to dynamically determine a freight route for each contracted shipment, it is much more common for carriers to establish a fixed flow plan that specifies a route for each shipment given its origin and destination terminals and its service requirements. This chapter will focus primarily on flow planning problems since they are in some sense the core service network design problem in trucking. Other problems will be considered when they also include some flow planning component, as we describe now.

Load planning, routing, and dispatch problems seek to determine how to build consolidated freight loads from shipments and time their dispatch. For trucking carriers, a load will be a trailerload or a containerload. Load planning problems can be tactical or operational. At the tactical level, a consolidation carrier would like to determine how many loads (of potentially different sizes) need to be dispatched between terminals, and at what times, in order to feasibly serve the demands induced at the flow planning step. When a load is planned between more distant terminals, it is also necessary to determine a movement path for the trailer through the network if it is not to be dispatched directly from its origin to destination terminal. This load routing step determines the sequence of relay points visited by the load and the transportation mode used for each connecting movement leg. When carriers dispatch trains of short trailers, like two pup combinations, it is also necessary to determine which loads to pair up into combinations when routing loads. At the operational level, loads need to be constructed from actual available shipments; often, loads may be cancelled or added on the day of operations, or shipments shifted onto alternate freight routes, to serve demands and utilize transportation capacity most cost-effectively. Modern service network design approaches often integrate flow and load planning rather than treat the problems sequentially; in such models, freight routing decisions and timed load dispatching decisions are made simultaneously. Additionally, operational models for load planning and dispatching may also allow limited flow replanning choices.

It is useful to note here that LTL and package express carriers are not the only firms that need to solve flow and load planning problems. Large shippers and 3PL companies often face flow and load planning problems when designing consolidation operations for distribution networks. Less-than-truckload shipments for such companies can be consolidated and routed through cross-docking facilities or pool points to avoid outsourcing to LTL carriers. Given a network design, such companies use truckload carriers to provide the trucking movements. It is also somewhat common in these cases for shippers to use multi-stop truckload movements referred to as “milk runs”. In this scenario, a shipper loads a truckload trailer at a single origin to be delivered to a sequence of partial load drop-off locations (or alternately might load at multiple pickup locations before moving the trailer to a single final destination for unloading).

Driver and equipment fleet management problems focus on building plans and schedules that enable trucking loads to be executed. Planned loads must be loaded into appropriate equipment, typically trailers or containers that have specific capabilities. Empty equipment repositioning problems are used to ensure that empty trailers and containers of the required equipment types are available over time where needed, and models for flow and load planning are more frequently now including constraints on equipment balance and availability. All trucking loads are moved at some point during their journeys by one or more truck drivers. Truck drivers must be managed to not violate government work regulations and sometimes are also subject to employee union restrictions. Consolidation carriers typically operate driver schedules that also must be planned in advance. Incorporating driver management decisions into flow and load planning models is often a difficult challenge because of the complicated nature of driver constraints.

Finally, vehicle routing and scheduling problems may also be considered service network design problems. For example, the classical capacitated vehicle routing can be described as a one-to-many load planning problem with vehicle resources that must operate on customer-disjoint cycles from a single depot; given this setup, the unique flow decision for each depot-to-customer shipment is to move from the depot along the route serving the customer, visiting intermediate stops as necessary before arriving to the destination. Consolidation carriers operate last-mile pickup and delivery operations that bring shipments from customer origin locations into first-level terminals at the beginning of trips and then distribute them at the end of trips, and thus they face specific vehicle routing and scheduling problems. Since the literature on last-mile truck vehicle routing and scheduling problems is vast, we will not cover it in this chapter. The reader is instead referred to excellent recent survey papers covering the area (see e.g., Golden et al. 2008; Cattaruzza et al. 2017; Braekers et al. 2016; Savelsbergh and van Woensel 2016; Psaraftis et al. 2016).

3 Network Design Models for Flow Planning

We begin with flow planning problems, the core service network design problems faced by LTL and package trucking carriers. The goal of flow planning problems is to determine a plan for consolidation of shipments into flows between transfer terminals to take advantage of certain cost scale economies in transportation. As an introduction, we begin by describing the components of network design mixed-integer programs for flow planning. To do so, we start with a base model of geographic consolidation.

Given a terminal set N, the freight carrier faces the problem of deciding how to transfer freight that originates at some terminal $o \in N$ and is destined for another terminal $d \in N$. We use the term *commodity* to describe such freight, and we let K be the set of all commodities to be moved by the carrier. Suppose that commodity k originates at terminal o_k and is destined for terminal d_k, and let q_k be a measure of the volume (or *flow*) of freight to be transferred. Note that a commodity represents the aggregation of shipments for many customers, and q_k measures this aggregated volume. Furthermore, suppose for now that at most one commodity is defined with the same origin-destination pair (o, d); this is possible, for example, when all shipments moving from o_k to d_k are promised the same transit time. Note here that volume or flow is a rate: a quantity moving (or to be moved) per time.

Typical units of measure for freight flow in trucking are pounds per day or tons per week, but it is important for flow planning models to know how this freight flow converts to the number of truck trailerloads necessary to move the volume.

During operations, detailed information about the size and weight of each shipment is used when determining how to pack and load trailers feasibly and effectively but this information is not known with certainty at the planning stage. For simplicity, it is common instead to convert estimated freight flows into an equivalent number of trailers by using simple factors (for example, with units of trailers per pound). It may be reasonably accurate to use a network-wide conversion factor for this task, however, the mix of freight shipment types (and their associated weight per cubic volume densities) may vary on different origin-destination lanes and thus it may be necessary to use different conversion factors on different lanes.

A *flow plan* is a set of decisions that specifies jointly how all commodities should be transferred from origins to destinations cost-effectively while meeting customer service requirements, the most important of which is the transit time. The simplest flow planning decision for a commodity is to move it in *direct trailers* or containers, loaded at the origin terminal o_k and unloaded at the destination d_k. We refer to this decision as a *direct route* for commodity k. Note that the use of the word "direct" in this context refers to the fact that the freight for this commodity will not be unloaded, sorted, and reloaded at any intermediate hub terminals. However, a direct trailer or container from terminal i to j may certainly be transported by a sequence of movements, by multiple drivers through relay points, or even by using multiple modes of transportation. A trivial and likely expensive flow plan would be to move all commodities along direct routes; note that given enough driver and trailer resources, this direct route flow plan should also be service feasible since there is no faster way to transfer freight between origins and destinations.

Consider then the non-trivial case where some commodities will not be assigned to direct freight routes. Let A be a set of directed arcs where $(i, j) \in A$ models a lane where trailers (or containers) can be loaded at terminal $i \in N$ and moved to terminal $j \in N$ for unloading. In a physical network with hub (N_H) and end-of-line (N_E) terminals, such load arcs (i, j) should exist between all pairs of hub terminals in N_H. On the other hand, when i or j is an end-of-line in N_E, it may be possible to reduce the number of arcs in a network by restricting the generation of direct loads to or from a limited set of terminals. Care should be exercised when doing so, however, since it may be more sensible to allow a model to decide where to build loads. Recall again that a direct load (i, j) does not imply that trailers are moved from i to j with a single driver or by a single movement.

Given A, let p_k be a possible freight route (or *path*) from o_k to d_k. Using the typical definitions from mathematical networks, each p_k is a simple path: a connected sequence of arcs in A beginning at node o_k and ending at node d_k with no cycles. For convenience, p_k may also be used to refer to a sequence of nodes in N where the initial node is o_k, the final node is d_k, and an arc $a \in A$ exists between each pair of adjacent nodes in the sequence. Let P_k be the set of all freight paths in A that connect o_k to d_k. Using these ideas, the primary decisions in every flow planning problem are to assign commodity flow to one or more feasible paths $p_k \in P_k$ for each commodity $k \in K$ to minimize logistics costs while meeting service requirements. Referring again to Fig. 14.1, a path from one of the end-of-line terminals on the left to one on the right, e.g., $c_2 \rightarrow c_1 \rightarrow e_3 \rightarrow b_1 \rightarrow b_2 \rightarrow$

$b_4 \rightarrow e_{20} \rightarrow c_5 \rightarrow c_6$, represents a freight path for that commodity, where a cross-dock transfer occurs at the head node of each arc in the path.

The remainder of this section will develop flow planning optimization formulations using *flat* network models, which we distinguish from *time-space* or *time-expanded* networks which model both geographic locations and explicit decision timing. Flat network models have the advantage that they lead to smaller integer programming instances, but they provide a relatively coarse approximation of trucking operations that is most useful for tactical planning. The input demands q_k represent average flow rates per time (e.g., tons or pallets or equivalent trailers per week) and the output freight and equipment decision variables also will represent flow rates per time. For this reason, it is natural to refer to such models as *rate-based*.

3.1 Arc-Based Flow Planning Model for Consolidation Trucking

To build a flow planning model, we make a few assumptions. Suppose that all shipments using truck movement lane $(i, j) \in A$ are loaded into a trailer or container at i and unloaded and sorted at j, and furthermore that the costs of transportation are separable by lane and the costs of sorting are separable by terminal. Finally, suppose that commodity flow rates q_k are roughly constant over time, and that any timing issues regarding consolidation can be safely ignored. Let x_{ij}^k be a decision variable representing the flow of commodity k moving on lane $(i, j) \in A$, measured in the same units as q_k. A generic mathematical programming formulation for flow planning is then:

$$\text{minimize} \quad \sum_{(i,j)\in A} f_{ij}^T(x_{ij}) + \sum_{i \in N} f_i^H(x_{i*}) \tag{14.1}$$

subject to

$$\sum_{(i,j)\in A} x_{ij}^k - \sum_{(j,i)\in A} x_{ji}^k = \begin{cases} q_k & \text{if } i = o_k \\ -q_k & \text{if } i = d_k \\ 0 & \text{otherwise} \end{cases} \qquad \forall k \in K,\ \forall i \in N \tag{14.2}$$

$$x_{ij} = \sum_{k \in K} x_{ij}^k \qquad \forall (i, j) \in A \tag{14.3}$$

$$x_{i*} = \sum_{k\in K \mid o_k \neq i} \sum_{(i,j)\in A} x_{ij} \qquad \forall i \in N \tag{14.4}$$

$$x_{ij}^k \geq 0 \qquad \forall k \in K,\ \forall (i, j) \in A \tag{14.5}$$

Constraints (14.2) and (14.5) define a simple (uncapacitated) case of the linear multi-commodity flow polytope, and this formulation allows freight flow to be split across many paths for each commodity. The objective function here is specified generically, where transportation costs are separable by lane (i, j) and handling costs are separable by terminal i. We now discuss how typical objective functions lead to network design mixed-integer programs.

First, it is most common in flow planning models for trucking terminal handling costs to be modeled as linear in throughput,

$$f_i^H(x_{i*}) = h_i x_{i*}, \tag{14.6}$$

where h_i is the handling cost rate per flow unit and x_{i*} is the total freight volume handled at terminal i. Handling in flow planning refers to the transfer of freight either via cross-docking of larger shipments or piece sorting in parcel operations. It is reasonable to assume that sorting labor cost or equipment operating cost grows roughly linearly with freight volume in flow planning models. Including handling costs in flow planning models explores a tradeoff with transportation costs; thus, it is quite common to estimate handling cost rates (which can be hard to measure precisely) to strike a reasonable balance with truck transportation costs.

Second, the truck transportation cost function on each arc should exhibit some cost economies scale in flow, *i.e.,* the average cost $\frac{f_{ij}^T(x)}{x}$ should be decreasing for at least some values of x to encourage consolidation. Note that with linear handling costs and linear transportation costs $c_{ij}x_{ij}$, the flow planning problem can be solved simply by finding a minimum cost path for a unit flow from o_k to d_k for each commodity and then moving all flow q_k along this path.

A reasonable approach for estimating truck transportation costs might be to assume a fixed cost d_{ij} for the dispatch of each unit trailerload (or containerload) on lane (i, j). Suppose all trailers have the same *capacity*, *i.e.,* once a trailer contains Q units of flow an additional trailer is needed. Then, f_{ij}^T is the following step function:

$$f_{ij}^T(x) = d_{ij}\left\lceil \frac{x}{Q} \right\rceil, \tag{14.7}$$

where the ceiling function rounds the value of x up to the next unit load. In practice, it is common to simply measure q_k in fractional trailers and to set $Q = 1$. Since the width of each step is always Q units and the height is always d_{ij}, it is straightforward to use an integer variable τ_{ij} to model this step function by simply forcing $\tau_{ij} \geq \frac{x_{ij}}{Q}$. It is important to make note of a few ideas when modeling transportation costs with per trailer lane costs d_{ij}. To estimate d_{ij} accurately requires that we know the movement (relay) path for the load from i to j in advance; it is common to use the most frequently used such path. Furthermore, since LTL and package carriers often dispatch trains of two short trailers together, this approach also is most accurate for carriers where dispatches almost never move short trailers alone; in such scenarios, d_{ij} represents one-half of the cost of moving a two-trailer train from i to j.

It may also be useful to model some transportation cost beyond the fixed cost per trailer that accrues linearly with flow, for example to account for fuel and maintenance costs that may increase with transported load size. In this case, define a linear arc flow cost as

$$f_{ij}^L(x) = c_{ij}x = (h_i + c_{ij}^T)x,$$

and now define the flow planning problem as the following mixed integer linear programming problem:

$$\text{minimize} \quad \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} x_{ij}^k + \sum_{(i,j) \in A} d_{ij} \tau_{ij} - \sum_{i \in N} \sum_{k \in K \mid o_k = i} h_i q_k \tag{14.8}$$

subject to

$$\sum_{(i,j) \in A} x_{ij}^k - \sum_{(j,i) \in A} x_{ji}^k = \begin{cases} q_k & \text{if } i = o_k \\ -q_k & \text{if } i = d_k \\ 0 & \text{otherwise} \end{cases} \qquad \forall k \in K, \ \forall i \in N \tag{14.9}$$

$$x_{ij} = \sum_{k \in K} x_{ij}^k \qquad \forall (i,j) \in A \tag{14.10}$$

$$x_{ij} \leq Q\tau_{ij} \qquad \forall (i,j) \in A \tag{14.11}$$

$$x_{ij}^k \geq 0 \qquad \forall k \in K, \ \forall (i,j) \in A \tag{14.12}$$

$$\tau_{ij} \geq 0 \text{ and integer} \qquad \forall (i,j) \in A \tag{14.13}$$

The non-negative integer variable τ_{ij} measures the minimum required trailers on lane (i, j) when (14.11) is matched with the positive objective function coefficient d_{ij}, thus properly modeling the lane step function dispatch costs given by (14.7). Note also that the objective function subtracts off a constant to avoid paying handling costs at terminals where freight originates; of course, including this constant or any other in the objective function does not affect the flow plan, only its computed cost. In the following subsections, we will no longer include such objective function constants in the flow planning formulations.

This generic arc-based flow planning model is a multi-commodity capacitated fixed-charge network design (MCND) problem. Solving this mixed-integer programming problem exactly can be difficult in practice for consolidation trucking networks of larger size. To understand the likely size of the optimization problems, consider an LTL carrier operating in North America with 100 terminals. If freight demand exists between half of the origin-destination terminal pairs (which is likely an underestimate), the result is a model with roughly 5000 commodities. Suppose further that 30 terminals are hubs and 70 are smaller end-of-lines; then, we should expect at least roughly 900 directed arcs between hub pairs that might be used by any commodity, and an additional few hundred arcs connecting out of or into end-of-lines that can be used only by commodities originating or destined for those terminals. For a network of this size, the number of commodity flow variables x_{ij}^k is large. If each directed arc connecting two hubs serves all 5000 commodities, and if each directed arc connected to an end-of-line serves approximately 50 inbound or outbound commodities, the model may have more than 4 million commodity flow variables and over 150,000 flow balance constraints of type (14.9). Thus, the linear relaxation of (14.8)–(14.13) is a very large linear program. Integer load counting variables τ_{ij} are defined for each arc, so there are at least 1000 of these variables and perhaps more. It can be important in practice to attempt to limit the number of x_{ij}^k variables by restricting which commodities might ever use specific hubs.

To date, exact approaches for solving these problems rely on using cutting planes to strengthen the mixed-integer programming formulation; results have been reported for instances with up to 100 nodes, 400 arcs, and 200 commodities, still too small for application to many real-world consolidation trucking networks. A simple yet effective cutting-plane algorithm for the flow planning MCND problem works as follows. Define *strong inequalities* for all commodities k and lanes (i, j) by:

$$x_{ij}^k \leq q_k \tau_{ij}. \tag{14.14}$$

The strong inequalities are clearly valid, and it has been shown that they are facet-defining for the convex hull of the so-called single-arc design relaxation of the MCND. Intuitively, the benefit of the strong inequalities should be clear when each individual commodity demand is small compared to the capacity of a single trailer Q: we can interpret the inequality as forcing the solution to allocate at least a partial single trailer on any lane where commodity k moves, and at least one trailer if all commodity k flow moves on the lane. Although strong inequalities are helpful, it is likely impractical to introduce them all to the formulation given the number of commodity flow variables. Separation of these inequalities, however, is simple because they can be checked directly for each lane and commodity. Thus, a reasonable exact solution approach for the flow planning MCND problem is to solve the linear programming relaxation at the root, and then to iterate introducing violated strong inequalities and resolving until no violations remain. The resulting linear programming formulation, extended with the identified subset of strong inequalities, is then solved by reintroducing the integrality constraints (14.13) and calling a MIP solver.

3.2 Single-Path and In-Tree Flow Planning Models

In addition to being difficult to solve in its generic form, the flow planning model (14.8)–(14.13) also has a number of drawbacks that limit its usefulness in practice. One deficiency is that flow for each commodity k can be split across potentially many paths connecting o_k to d_k, and the fraction of commodity moved on any such path may be arbitrarily small. When developing a flow *plan* for consolidation, LTL and package trucking carriers seek both realism and simplicity. During operations, the actual freight shipments (and total flow volume) for any commodity may differ from the expected flow used for planning so it is at best not clear when to choose one path for a particular shipment versus another. While we believe that it is appropriate that such models ignore the details of individual shipments and model demand as continuous commodity flows, it is at the same time likely necessary to exercise some control over flow splitting during this planning phase. Of course, in practice carriers may divert shipments during operations onto alternative transfer paths through different cross-dock terminals.

A simple but restrictive way to eliminate commodity flow splitting is to enforce a *single path* constraint when building a flow planning model. Here, all commodity k flow is directed to a single transfer path from o_k to d_k in the plan. Consider the following formulation that embeds this restriction. Suppose we also introduce a new mechanism to model commodity flow where variables y_{ij}^k now measure the *fraction* of commodity k demand volume that is transferred directly from terminal i to j. Using this redefinition, $x_{ij}^k = q_k y_{ij}^k$. Now, if we restrict the y variables to be binary, we can easily enforce a single-path restriction:

$$\text{minimize} \quad \sum_{(i,j)\in A} (d_{ij}\tau_{ij} + c_{ij}x_{ij}) \tag{14.15}$$

subject to

$$\sum_{(i,j)\in A} y_{ij}^k - \sum_{(j,i)\in A} y_{ji}^k = \begin{cases} 1 & \text{if } i = o_k \\ -1 & \text{if } i = d_k \\ 0 & \text{otherwise} \end{cases} \qquad \forall k \in K,\ \forall i \in N \tag{14.16}$$

$$x_{ij} \le Q\tau_{ij} \qquad \forall (i,j) \in A \tag{14.17}$$

$$x_{ij} = \sum_{k\in K} q_k y_{ij}^k \qquad \forall (i,j) \in A \tag{14.18}$$

$$y_{ij}^k \in \{0,\ 1\} \qquad \forall k \in K,\ \forall (i,j) \in A \tag{14.19}$$

$$\tau_{ij} \ge 0 \text{ and integer} \qquad \forall (i,j) \in A \tag{14.20}$$

Constraints (14.18) convert commodity flows into total flow on direct lanes, and thus the objective function and trailer counting constraints can remain as in the initial model. Constraints 14.16 ensure that all commodity k demand is transferred from o^k to d^k. Furthermore, when the y variables are restricted to take binary values, these constraints ensure that the y^k variables identify a single path from o_k to d_k. It is again possible to introduce strong inequalities to this formulation of the form $y_{ij}^k \le \tau_{ij}$ to strengthen the linear programming relaxation.

This single-path flow planning formulation can be referred to as an *unsplittable flow* capacitated network design problem (see e.g., Atamtürk and Rajan, 2002). Note that the original splittable formulation can also be modeled with y variables by relaxing (14.19) to $y_{ij}^k \in [0, 1]$; the original formulation can be recovered by substituting $x_{ij}^k = q_k y_{ij}^k$ in this case.

Practical models for truck flow planning are often even more restricted. Consider the sort operations at a trucking terminal i. Arriving trucks are unloaded, and shipments that must be transferred (since they have not arrived at their final destination) are sorted for loading into outbound trailers. In typical operations, each outbound trailer is destined for a single next terminal j where it will be unloaded entirely. Technology is certainly available today for each shipment to have a customized sorting plan; at terminals with appropriate technology, a shipment can be scanned and then sorted for loading onto an appropriate outbound trailer (by a terminal worker or by automated sorting equipment). However, carriers often operate simpler plans that specify rules that guide how groups of shipments are to be sorted. One option still used in many LTL and package express systems is to determine the next terminal for each unloaded shipment using only its final destination.

Suppose that a consolidation plan is such that all freight shipments unloaded at terminal i with the same final destination d ($i \neq d$) are transferred to a single next terminal j. We will call such a design an *in-tree flow plan* because the directed graph induced by the union of paths for commodities $K_d \subseteq K$ that share a common destination d is a directed in-tree on (a subset of) the terminal nodes N.

In-tree plans are a subset of the feasible single-path plans, and we can modify the formulation to handle this restriction. To do so with the simplest formulation, we introduce and make use of a common redefinition of commodities that is frequently used in network design. It is well known that some multi-commodity network design problems can be formulated where each commodity represents all shipment flow to a common destination (or alternately, all shipment flow from a common origin); these redefined commodities are referred to as *aggregated* commodities. In the destination variant of aggregated commodities, let D be the set of destinations d that have positive inbound freight flow. If we begin with our original origin-destination commodity definition, then we can define aggregated commodities for each destination d with the following net supply of flow:

$$b_i^d = \begin{cases} q_k & \text{if } i = o_k \text{ for some } k \in K_d \\ -\sum_{k \in K_d} q_k & \text{if } i = d \\ 0 & \text{otherwise} \end{cases} \qquad \forall d \in D, \ \forall i \in N. \tag{14.21}$$

It is descriptive to refer to this type of aggregation as a many-to-one commodity into terminal d. Aggregated commodities have the obvious benefit of reducing problem size. If the number of terminals $|N| = n$, then the number of possible flow decisions for each arc is $O(n)$ instead of $O(n^2)$ and the number of flow balance constraints is also reduced by $O(n)$. However, aggregated commodities may not be useful when there is the need to explicitly model specific origin-destination flow path requirements, for example path duration or cardinality constraints. It is also not possible to use aggregated commodities to model a single-path flow planning problem unless the paths inbound to every destination d are constrained also to form a directed in-tree.

Consider now an in-tree flow planning model with many-to-one commodities. Let binary decision variable y_{ij}^d be used to indicate whether freight flow for commodity d (with final destination d) that originates or transfers at terminal i is transferred next to terminal j. Let continuous variable x_{ij}^d measure freight flow for commodity d moving on truck trailers from i to j, as usual. Consider the following formulation:

$$\text{minimize} \quad \sum_{(i,j)\in A} (d_{ij}\tau_{ij} + c_{ij}x_{ij}) \tag{14.22}$$

subject to

$$\sum_{(i,j)\in A} x_{ij}^d - \sum_{(j,i)\in A} x_{ji}^d = b_i^d \qquad \forall d \in D,\ \forall i \in N \tag{14.23}$$

$$\sum_{(i,j)\in A} y_{ij}^d \le 1 \qquad \forall d \in D,\ \forall i \in N \tag{14.24}$$

$$x_{ij}^d \le \left(\sum_{k\in K_d} q_k\right) y_{ij}^d \qquad \forall d \in D,\ \forall (i,j) \in A \tag{14.25}$$

$$x_{ij} \le Q\tau_{ij} \qquad \forall (i,j) \in A \tag{14.26}$$

$$x_{ij} = \sum_{d\in D} x_{ij}^d \qquad \forall (i,j) \in A \tag{14.27}$$

$$y_{ij}^d \in \{0,\ 1\} \qquad \forall d \in D,\ \forall (i,j) \in A \tag{14.28}$$

$$\tau_{ij} \ge 0 \text{ and integer} \qquad \forall (i,j) \in A \tag{14.29}$$

The in-tree model includes constraint (14.24) to ensure that the freight flow paths for destination d form a directed in-tree to d by allowing only one outbound direct transfer arc (i, j) to be selected from terminal i for that freight. Constraint (14.25) ensures that freight destined for d can only be dispatched on lanes included in the selected in-tree, where the capacity coefficient for y_{ij}^d is the smallest big-M value that yields a valid formulation. Again, valid inequalities can be introduced to this formulation. For example, replacing y_{ij}^d with τ_{ij} in (14.25) yields a version of the strong inequalities. Inequalities $y_{ij}^d \le \tau_{ij}$ are also valid for all $d \in D$ and $(i, j) \in A$.

3.3 Path-Based Models for Flow Planning

Each of the flow planning formulations presented thus far has been an *arc-based* network design model. For parsimony, such models use arc flow decision variables of the form x_a^k and x_a to represent respectively the fractional freight flow for commodity k and all commodities moved via truck dispatch arc a. When many possible feasible paths exist for routing commodity k freight from o_k to d_k, this modeling decision has merit since it may reduce the number of required decision variables (and this remains true even for the single-path and in-tree models that select only a single path for each such pair).

In many truck transportation applications, however, it is better to use a *path-based* network design model because of the flexibility these models provide in representing specific restrictions that may arise in practice. Such models replace the variables x_a^k with path-flow variables x_p^k, where p represents some path in P_k for commodity k. We can then modify the generic arc-based model into the following generic path-based model:

$$\text{minimize} \quad \sum_{(i,j)\in A} (d_{ij}\tau_{ij} + c_{ij}x_{ij}) \tag{14.30}$$

subject to

$$\sum_{p\in P_k} x_p^k = q_k \qquad \forall k \in K \tag{14.31}$$

$$x_{ij} \leq Q\tau_{ij} \qquad \forall (i,j) \in A \tag{14.32}$$

$$x_{ij} = \sum_{k\in K} \sum_{p\in P_k \,|\, (i,j)\in p} x_p^k \qquad \forall (i,j) \in A \tag{14.33}$$

$$x_p^k \geq 0 \qquad \forall k \in K,\ \forall p \in P_k \tag{14.34}$$

$$\tau_{ij} \geq 0 \text{ and integer} \qquad \forall (i,j) \in A \tag{14.35}$$

Since all paths in P_k provide connectivity from o_k to d_k, flow balance constraints are no longer required and instead are replaced by (14.31) which partitions commodity demand flow across the available paths in P_k. Constraints (14.33) aggregate all commodity flow on direct movement arc (i, j) by finding all flow on paths for all commodities where the arc (i, j) is included in the path. Note also that although (14.30) does not include a linear cost term for the path flow variables, adding one is possible; in practice, such terms can be used to model freight handling costs at intermediate cross-dock transfer terminals rather than using a cost linear in the total arc flow x_{ij}.

The primary benefit of such a formulation is that the model explicitly defines the sets of allowed transfer paths P_k for each commodity. This feature makes it easy to model a number of real-world restrictions. Most importantly, suppose that a service requirement requires that the duration of the transfer path for commodity k from o_k to d_k (travel time plus terminal cross-dock time) is limited by an upper bound. Then, only paths that meet this duration requirement can be included in P_k. Similarly, it may also be desirable to limit the number of terminal transfers for commodity k. For example, when o_k and d_k are nearby perhaps only one transfer should be considered in any path, but for more distant terminals two or three transfers might be acceptable; again, only acceptable paths need be included in P_k.

Building path sets P_k given o_k and d_k is usually conducted by using a graph search algorithm, like breadth-first or depth-first search, from o_k using the direct arcs $(i, j) \in A$. Constraints on allowable paths can be used to truncate the search tree. For many problems, enumerating the complete feasible path set P_k for each commodity k would create too many decision variables and very large instances. Column generation approaches for solving linear programming relaxations (either at the root node of a branch-and-bound tree or at all nodes in a branch-and-price scheme) can be used in these cases. Heuristics that only enumerate reasonably-sized subsets of the path pools for each commodity may also find good solutions in practice.

To use a path-based model while enforcing an in-tree flow plan structure, we can again add binary arc selection variables y_{ij}^d to the formulation and selection constraints (14.24). Suppose furthermore that the commodity set K is partitioned into subsets $K(d)$, where commodity k is included in $K(d)$ if its freight destination $d_k = d$. To ensure that the set of all paths used for commodities in $K(d)$ forms a directed in-tree into d, the following compatibility constraints can be used:

$$\sum_{p \in P(k) \mid a \in p} x_p^k \leq q_k y_a^{d_k} \quad \forall k \in K,\ a \in A \tag{14.36}$$

Note that these aggregated forcing constraints are stronger than those disaggregated by path, yielding a stronger linear relaxation formulation. The disaggregated constraints have the simpler form $x_p^k \leq q_k y_a^{d_k}$ and are defined for all k, $p \in P(k)$, and $a \in p$; it is easy to see that there are feasible solutions to the disaggregated constraints system when the variables are continuous that are not feasible for the aggregated constraint. Of course, in-tree constraints will force all commodity k flow onto a single path $p \in P(k)$ into d_k for integer values of y. It is thus possible to redefine x_p^k in this case to be a binary selection variable, and to modify constraints (14.31) into assignment constraints with right-hand side values of one. After this modification, the aggregated forcing constraints take the form $\sum_{p \in P(k) \mid a \in p} x_p^k \leq y_a^{d_k}$ for all k and a. Finally, there are some terminals in consolidation trucking systems that cannot be used to transfer inbound freight from other terminals; most end-of-line terminals in LTL systems operate this way. When binary path selection variables are used, it is not necessary to include tree selection variables y_a^d for arcs a departing such terminals since originating flow destined for d will automatically be forced onto a single path and no transfer freight exists.

3.4 Balancing Resources in Flow Planning

The model of transportation costs discussed in Sect. 3.1 assumes that trailer movements τ_{ij} and their associated costs are determined only by one-way loaded flows and thus ignores the important fact that trailer and container resources are reused over time. Empty trailers must be available at load origins before loads can be created and moved, and the problem of *empty repositioning* of equipment is critically important in most freight transportation settings. When empty repositioning is ignored during flow planning, opportunities to use empty trailer capacity to move freight may be overlooked. Flow planning models that explicitly include empty resource flows and their associated costs seek to address this shortcoming.

To show how planned flow costs may be reduced by integrating empty repositioning decisions in flow planning when compared to the sequential deployment of a flow planning model followed by an empty trailer balancing problem, consider a simple example with 3 terminals as illustrated in Figs. 14.2 and 14.3. Suppose $\frac{1}{2}$ trailerloads of demand exists from a to b and from a to c and one trailerload from b to c. Suppose that the distance from a to b or c is $\frac{2}{3}$ and the distance from b to c is one. In the absence of empty balance, the optimal solution is to move full trailerloads on lanes (a, b), (a, c), and (b, c); doing so creates imbalance and two trailers should be returned on leg (c, a). The total trailer distance in this solution is $\frac{11}{3}$, but the loaded trailer distance is only $\frac{7}{3}$. If empties were balanced simultaneously, it is better to load the a to b freight via terminal c. This creates loaded trailers on (a, c), (b, c), and (c, b) and total loaded trailer distance of $\frac{8}{3}$. However, empty balance can be achieved by only sending one empty on (c, a) and thus the total trailer distance is $\frac{10}{3}$.

Up to this point, the set A of arcs $a = (i, j)$ has been used to represent opportunities to move loaded trailers from terminal i to terminal j; note that loading of the trailer occurs at terminal i and unloading at terminal j. It is certainly possible to limit empty trailer movements to the arcs in A. There are some cases, however, when empty trailers might move between terminals where there is *never*

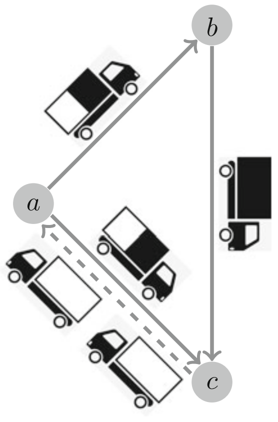


Fig. 14.2 Flow plan without empty balancing (Loaded distance = $\frac{7}{3}$, totaltrailer distance = $\frac{11}{3}$)

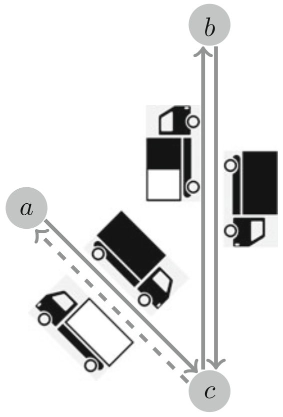


Fig. 14.3 Flow plan with empty balancing (Loaded distance = $\frac{8}{3}$, totaltrailer distance = $\frac{10}{3}$)

a trailer loaded at i to be unloaded at j. Examples that have been encountered in practice include pairs of hubs in large metropolitan areas, nearby pairs of end-of-line terminals, or connections between large customer facilities (another type of end-of-line). To model terminal-to-terminal physical movements of trailers more explicitly, let A^D be a set of *dispatch lanes* for a trucking company that includes A but may also include additional connections where empties may be moved: $A \subseteq A_D$. It is also common in practice for A_D to have an important property: a directed path of dispatch lanes in A^D should exist from j to i, for each $(i, j) \in A$. The existence of the reverse path is a sufficient (but not necessary) condition to ensure that empty resources can return to load origins for reuse. If the reverse path does not exist for each $(i, j) \in A$, which is unlikely in practice, then it is important to guarantee that empty trailers can be balanced using a different mechanism.

Suppose now that empty trailers and containers can be moved on any dispatch lane $(i, j) \in A^D$. Let η_{ij} count the number of empty unit loads moving on dispatch lane (i, j), measured in the same units as τ_{ij}. We can enforce equipment balance then at each terminal i by adding the following constraint to any of the flow planning formulations presented thus far:

$$\sum_{(i,j)\in A} \tau_{ij} + \sum_{(i,j)\in A^D} \eta_{ij} - \sum_{(j,i)\in A} \tau_{ji} - \sum_{(j,i)\in A^D} \eta_{ji} = 0 \qquad \forall i \in N \tag{14.37}$$

Given this balance constraint, flow plans can be determined by including an appropriate cost for moving empty trailers in the objective function. If d_{ij}^E is the cost of moving an empty unit load on dispatch lane (i, j), then we can add the term $\sum_{(i,j)\in A^D} d_{ij}^E \eta_{ij}$ to any of the objective functions to capture empty costs. Doing so is likely to lead to changes in the optimal flow plans and empty trailer balance plans that would result from solving the problems sequentially: some freight will be assigned optimally into natural empty backhaul corridors, and balancing backhaul trailer movements may deviate from the most direct (cheapest) paths to attract freight.

3.5 Slope-Scaling Heuristics for Flow Planning

Flow planning optimization models are important in practice for the design of consolidation transportation networks, but exact optimization can be difficult for instances of realistic size. One heuristic approach to solving these problems is to simply limit the number of decision variables that are defined to a tractable number; this idea is easiest to implement with path-based models as we described earlier, but it is also possible with arc-based approaches. Slope scaling is a different heuristic idea and can be useful for problems that have difficult non-linear objective functions that, when linearized, lead to optimization problems that can be solved efficiently. We will now describe how slope scaling can be used to solve flow planning problems, and how the linearized subproblems decompose into shortest-path problems for both the base model and the single-path model (whose slope-scaling solutions are therefore equivalent). Furthermore, the intree model can also be solved by a shortest-path decomposition (and is therefore equivalent to the base and single-path models) when the objective function cost coefficients are independent of commodity k.

Slope scaling heuristics are useful for problems with difficult non-linear objective functions but with linear constraints and decision variables. The generic model (14.1)–(14.5) has this form, since its constraints are separable by commodity k and, when separated, describe the minimum-cost path polytope since all commodity k flow has a single origin (as well as a single destination) and arc flows are uncapacitated. We now show how to use slope scaling to find solutions first for model (14.8)-(14.13). Note that we can eliminate the integer dispatch variables τ_{ij} by rewriting the objective function recognizing that cost-minimizing values for τ_{ij} follow directly from the flow variables since it is non-decreasing in τ. Thus, we can define:

$$\tau_{ij} = \left\lceil \frac{\sum_{k\in K} x_{ij}^k}{Q} \right\rceil, \tag{14.38}$$

and the optimization model can be rewritten as minimizing the objective function:

$$\sum_{k\in K}\sum_{(i,j)\in A} c_{ij}x_{ij}^k + \sum_{(i,j)\in A} d_{ij}\left\lceil \frac{\sum_{k\in K} x_{ij}^k}{Q} \right\rceil - \sum_{i\in N}\sum_{k\in K\,|\,o_k=i} h_i q_k \tag{14.39}$$

subject to (14.9) and (14.12).

To solve with a slope-scaling approach, we linearize the objective function by replacing the ceiling function:

$$\sum_{k\in K}\sum_{(i,j)\in A} c_{ij}x_{ij}^k + \sum_{(i,j)\in A} \rho_{ij}(t)\sum_{k\in K} x_{ij}^k - \sum_{i\in N}\sum_{k\in K\,|\,o_k=i} h_i q_k, \tag{14.40}$$

where $\rho_{ij}(t)$ is the slope coefficient in iteration t. Rearranging terms yields:

$$\sum_{k\in K}\sum_{(i,j)\in A}\left(c_{ij}+\rho_{ij}(t)\right)x_{ij}^k-\sum_{i\in N}\sum_{k\in K\,|\,o_k=i}h_iq_k, \tag{14.41}$$

A slope scaling heuristic finds each solution to the flow planning problem by selecting a fixed vector $\rho_{ij}(t)$, then determining x_{ij}^k that minimize (14.41) subject to (14.9) and (14.12), and then finally specifying trailer flow variables using (14.38). It should be clear that the minimization problem for $\rho_{ij}(t)$ is a linear program that is separable by commodity k. Moreover, each separable subproblem is to find a minimum-cost path from o_k to d_k for commodity k given arc cost coefficients $c_{ij}+\rho_{ij}(t)$. Since this solution is a single flow path for each commodity k, it also follows that the solution to any slope scaling subproblem will also be a single-path flow plan; thus, solving (14.8)–(14.13) is equivalent to solving (14.15)–(14.20) by slope scaling. Furthermore, note also that the objective coefficients on commodity arc flow in (14.41) are independent of commodity k. If we consider all commodities k that share a common destination $d_k=d$, we can find a joint set of minimum-cost paths given arc costs $c_{ij}+\rho_{ij}(t)$ using an algorithm that produces an in-tree to destination d, like Dijkstra's Algorithm. Thus, a solution found during any slope-scaling iteration for an in-tree flow planning problem is also optimal for the base or single-path problems with the same linearization multipliers $\rho_{ij}(t)$. If instead the commodity arc flow cost coefficients had the more general form $c_{ij}^k+\rho_{ij}(t)$, separate shortest path problems would be necessary for each commodity for the base and single-path slope scaling problems. Furthermore, the slope scaling problem for the in-tree flow planning model would require solving mixed-integer program for each destination d to enforce the tree structure on the joint set of paths.

Consider then the following slope scaling approach for solving the base, single-path, or in-tree flow planning problem. We initialize the slope coefficients using a lower-bounding approximation: $\rho_{ij}(1)=\frac{d_{ij}}{Q}$ for each arc. Then, for each iteration t, we minimize (14.41) subject to (14.9) and (14.12) by first finding a shortest-path in-tree to each destination d using arc costs $c_{ij}+\rho_{ij}(t)$, then assigning commodity flow q_k along the identified shortest path from o_k to d_k for each $k\in K$ yielding $x_{ij}^k(t)$, and finally determining arc trailer flows $\tau_{ij}(t)$ using (14.38). The true objective function cost of this solution is $C(t)$, determined using (14.8). If $C(t)$ is the lowest objective function cost found so far, it is recorded and the best solution is updated. If the solution $x_{ij}^k(t)$ remains unchanged from the prior iteration $t-1$, then we terminate and return the best found solution. Otherwise, we adjust the slope coefficients as follows and move on to iteration $t+1$:

$$\rho_{ij}(t+1)=\begin{cases}\rho_{ij}(t) & \text{if } \sum_{k\in K}x_{ij}^k(t)=0\\ \dfrac{d_{ij}\left\lceil\frac{\sum_{k\in K}x_{ij}^k(t)}{Q}\right\rceil}{\sum_{k\in K}x_{ij}^k(t)} & \text{if } \sum_{k\in K}x_{ij}^k(t)>0\end{cases} \tag{14.42}$$

Note finally that it may also be reasonable to terminate a slope-scaling search after a maximum number of iterations to avoid excessive computation time.

3.6 A Local Search Heuristic for Flow Planning

Local search and metaheuristic extensions of local search are important heuristic approaches for solving flow planning problems. Most heuristics of this type take advantage of the fact that minimum cost path algorithms can frequently be used, with modifications to arc costs or network structure, to decide on new freight flow paths for commodities during search iterations. We now describe the core ideas of a local search approach that for many years was used as a key component in linehaul network design for LTL carriers; see Sect. 5 for more background. The ideas presented here will closely follow those developed initially in Powell (1986) for what was then referred to as the load planning problem for LTL carriers; in the terminology of this chapter, the problem considered is a flow planning problem.

The base problem considered is to create an in-tree flow plan of the type described by constraints (14.23)–(14.28), however we will assume only non-negative trailer flow τ_{ij} as described below. The LTL flow planning problem will be to decide τ_{ij} on direct movement lanes $(i, j) \in A$ and freight flows x_{ij}^d for each aggregated destination commodity $d \in D$. However, this planning problem considers a simpler transportation cost function $f_{ij}^T(x_{ij})$ given total freight flow x_{ij} on lane (i, j) measured in fractional trailerloads. If any flow is assigned to the lane, we incur a fixed cost equivalent to dispatching a minimum flow of trailers M_{ij} at cost $d_{ij}M_{ij}$. Once the capacity of this minimum flow is exceeded, we approximate additional trailer dispatching cost with a linear term $d_{ij}x_{ij}$. If we use (14.22) as the flow planning objective function, then we represent this cost approximation by determining the trailer flows τ_{ij} as follows:

$$\tau_{ij} = \begin{cases} 0 & \text{if } x_{ij} = 0 \\ M_{ij} & \text{if } 0 < x_{ij} \leq M_{ij} \\ x_{ij} & \text{if } M_{ij} < x_{ij} \end{cases} \tag{14.43}$$

Note that the objective function cost term $d_{ij}\tau_{ij}$ exhibits cost scale economies in x_{ij} for the first M_{ij} units of freight flow to encourage consolidation. If M_{ij} were one for all (i, j), this trailer counting function is a lower bound for the trailer step function introduced earlier in model (14.8)–(14.12). If q_k measures weekly rates of demand, then $M_{ij} = 5$ would indicate that at least one trailer should be sent each weekday on dispatch lane (i, j) if any flow is assigned to that lane. Minimum dispatch frequencies can be used to ensure that reasonable service levels are provided to commodities that use this lane in their (o_k, d_k) path.

We now describe a two-tier local search heuristic for finding solutions to this model. The first tier heuristic selects a subset $A_S \subseteq A$ of direct service lanes (i, j) to make available for use; including lane (i, j) in A_S implies that a minimum of M_{ij} trailers will flow on the lane. A feasible first-tier solution is one where at least one freight path exists between o_k and d_k for each (disaggregated) commodity $k \in K$ using only arcs in A_S. Given such a feasible A_S, the second tier heuristic determines a freight flow path for each $k \in K$ to minimize the value of the objective function. The flow q_k for commodity k is assigned to each arc (i, j) in its path, thus determining the total arc flows x_{ij} and trailer flows t_{ij} and the concomitant objective function value. Note that the joint set of freight flow paths into each $d \in D$ must form a directed in-tree.

We begin with the second tier problem, referred to in the literature as the *routing subproblem*. First, note that this minimization problem is on its own a piecewise-linear convex multi-commodity flow problem with a relatively simple cost structure. If we rewrite the objective function as

$$\min \sum_{(i,j)\in A_S} (D_{ij} + c_{ij}x_{ij}), \tag{14.44}$$

we can replace the trailer flow variables with D_{ij} by adding the following constraints:

$$D_{ij} \geq d_{ij}M_{ij} \qquad \forall\,(i, j) \in A_S \tag{14.45}$$

$$D_{ij} \geq d_{ij}x_{ij} \qquad \forall\,(i, j) \in A_S \tag{14.46}$$

and we can solve a mixed-integer programming problem with constraints (14.23)–(14.25) and (14.27) and (14.28). Note that the in-tree structure is what makes this optimization problem difficult since it is otherwise a linear program. A reasonable solution approach might be to only introduce tree selection variables and constraints for nodes i and destinations d when violations occur when they are ignored.

Another idea is to solve the second tier routing subproblem by local search. To do so, note that a feasible set of in-tree paths for each destination d can be found by solving a shortest-path problem with arc costs $d_{ij} + c_{ij}$. By moving all q_k flow for commodity k along this shortest path, it should also be clear that the total cost of the resulting arc flows $\overline{x}_{ij}$,

$$\sum_{(i,j)\in A_S} \left(d_{ij} + c_{ij}\right)\overline{x}_{ij}, \tag{14.47}$$

is a lower bound on the objective function value for the routing subproblem. In fact, if $\overline{x}_{ij} \geq M_{ij}$ for all arcs, this solution is optimal. Thus, the only way to improve a solution is to find arcs where $\overline{x}_{ij} < M_{ij}$ and determine whether any commodities can be re-routed to use them and reduce cost. To do so, let an *override* indicate when a specific value y_{ij}^d is forced equal to one by the heuristic; the goal of setting an override will be to force a shortest-path algorithm to include arc (i, j) surely in the in-tree to destination d. Given a set of overrides, it is easy to modify an algorithm, like Dijkstra's, to find a shortest-path in-tree to d conditional on including all arcs where $y_{ij}^d = 1$. The idea is simple: when extending labels from j to upstream nodes k along arcs (k, j), we only update the cost label (dual) at k if its cost improves and either $y_{kj}^d = 1$ or no override is set outbound from node k.

Thus, a core component of a heuristic for the routing subproblem will be to determine commodity flows $\overline{x}_{ij}^{d}$ and arc flows $\overline{x}_{ij}$ by building shortest-path in-trees to each destination $d \in D$ modified by current overrides y_{ij}^{d}. Given this solution, we can seek to reduce its cost by attracting flow to arcs where $\overline{x}_{ij} < M_{ij}$. Let $\overline{y}^{d}$ indicate the current in-trees selected for this solution. One approach to attracting flow to (i, j), described originally in Powell (1986) as IFOL-0, is to consider all destinations $s \in D$ where $\overline{y}_{ij}^{s} = 0$ and some flow quantity $q^{s}(i)$ has accumulated at i for transfer onward to s. Note that $q^{s}(i)$ may include both originating flow at i and also the sum of flows from upstream origin terminals k such that the path from k to s on the tree arcs a where $\overline{y}_{a}^{s} = 1$ includes terminal i. If we were to change the in-tree for s such that (i, j) were included in the in-tree (by setting override $y_{ij}^{s} = 1$), then the new path for this flow would include arc (i, j) and then follow the tree arcs from j to s. In the original IFOL-0 approach, the destinations s are sorted by decreasing estimated savings from diversion onto (i, j) computed by $q^{s}(i)$ multiplied by the *marginal* cost of the new path (where an arc marginal cost is zero if its flow is less than M_a, and c_a otherwise), and then processed in order of estimated savings where commodity s is diverted onto (i, j) only if it generates actual cost savings.

Given an approach for the second tier routing subproblem, it remains to discuss a heuristic approach for selecting the arc subset A_S. This first tier problem, or network design master problem, can be addressed with a simple local search heuristic. Consider an LTL trucking network that currently dispatches loads on a set of lanes $\overline{A}_S$. Reasonable local search neighborhoods modify $\overline{A}_S$ by dropping a single arc or adding a single arc each iteration, generating a new routing subproblem solution and moving to the new solution generated only if total cost is reduced. When dropping an arc $a = (i, j)$, a set of destinations D_a where $y_{a}^{s} = 1$ for $s \in D_a$ is disrupted and a new in-tree needs to be constructed for each s to create a feasible solution. On the other hand, adding an arc a creates an opportunity to re-route flow to reduce cost and a procedure like IFOL-0 can be used to see which flow should be attracted to a. One approach to structuring such a heuristic would be to consider dropping all arcs from $\overline{A}_S$ first one-by-one, focusing first on those connecting breakbulk terminals to end-of-lines and vice versa and then moving to those connecting two breakbulks. After considering all such drops, a set of arcs could be considered for adding to the network. Several passes over a drop-add sequence should be conducted.

4 Network Design Models for Flow and Load Planning

Load planning in consolidation trucking is a more detailed task than flow planning and is at its core a scheduling activity. Given a flow plan, a trucking company needs to provide adequate transportation capacity between terminal pairs to support the flows over time. Generally a *schedule* is constructed for a time horizon, like a week or a month, to provide this capacity. This schedule typically includes *loads*, *empties*, *movements or dispatches*, and *drivers*. A load is planned to be built at some origin terminal at some time, and then dispatched to a destination terminal to arrive by some time. A load also specifies a planned type of trailer or container equipment to be used (and its size). Empty loads (or empties) are planned both to recirculate equipment back to load origins but also, in some cases, to move drivers back to their home terminals. Movements, or dispatches, refer to terminal-to-terminal movements by drivers or outsourced transportation modes with one or more loads. The movement and driver schedule required to execute planned loads and empties is typically not considered part of the load planning problem.

Rate-based flat network service network design models are still useful for flow planning, and path-based variants in particular can model some important timing considerations for commodities. However, they have a few important drawbacks. Flat network models are not particularly useful for detailed flow and load planning primarily because they do not accurately model the *timing* of consolidation activities at transfer terminals or details about when equipment is available for dispatch. For this reason, flat network models are not usually deployed for *load planning* problems that seek to explicitly create plans for timed dispatches of trucks during an operating day. We now introduce *time-expanded* network models for such service network design problems. In this section, we will use the term *flow and load planning models* to refer to those that both create capacity and plan shipment flows through a consolidation network while simultaneously planning loaded and empty trailer dispatches during a planning horizon.

4.1 A Time-Expanded Model for LTL Flow Planning

Before we explore models for joint flow and load planning, we introduce an important time-expanded network model for LTL flow planning; the model and solution approach in this section were first described in Jarrah et al. (2009). Consider a time-expanded network where $\mathscr{N}$ and $\mathscr{A}$ represent the set of time-space (terminal, time) nodes and time-space arcs (denoting the timed trailer dispatch lanes), respectively. To model the network over time, suppose that each geographic terminal in N is replicated once for each of the five weekdays to yield the nodes in $\mathscr{N}$. Similarly, each geographic load dispatch lane in A is also replicated for each weekday to yield the arcs in $\mathscr{A}$. Note that load lanes that take more than a single travel day to reach their destination are connected forward to the appropriate destination node in $\mathscr{N}$. The arcs in $\mathscr{A}$ also now include holding arcs forward one time period (weekday) for each terminal node in $\mathscr{N}$. Since carriers actually dispatch loads at more than a single time per day, this model is best described as one of tactical flow planning rather than a detailed load planning and dispatch model.

Given this time-expanded network structure, each commodity demand now specifies a timed origin node and timed destination node, both in $\mathscr{N}$. Using this structure, the volume of freight moving between geographic terminals can be modeled to vary by day-of-week. Furthermore, the transit time requirement for each origin-destination pair can be modeled (at the level of days) by choosing an appropriate timed destination node.

If the goal is to produce an in-tree flow plan for each terminal destination $d \in N$, it is possible to formulate the flow planning problem with binary in-tree selection variables. To do so, it is possible to modify a path-based flow planning model like (14.30)–(14.36) to one where the primary binary decision variables are w_ℓ^d, indicating whether or not complete in-tree ℓ is selected for destination $d \in N$. This approach assumes that the same in-trees will be used each operating day of the week in the flow plan. In such a formulation, each in-tree is comprised of time-expanded paths from origin terminals into d, replicated for each operating day and consistent with the in-tree property that only a single outbound terminal $j \in N$ can be selected for flow outbound from $i \in N$ for destination d. In fact, since the model also includes holding arcs, the in-tree property is extended in this case to holding: if any freight at (i, t) destined for d is held to $(i, t + 1)$, then all such freight must be held.

A feature of such an in-tree model is that a unique time-expanded path connects each timed origin to a timed copy of d. Thus, if a specific tree ℓ is selected for d, a precise mapping of commodity freight volumes destined to d to time-expanded arcs is known; in this way, a set of tree selection decisions implies freight volumes on all time-expanded arcs which in turn specifies load counts and fixed transportation costs. Additionally, empty trailer balancing constraints are included in this model and are a straightforward extension of constraints (14.37) to the case with time-expanded nodes and arcs.

Specifying a flow planning model with tree variables is convenient, but the drawback in practice is that there are far too many feasible in-trees for each destination to enumerate. A heuristic approach to solve this integer programming model is to use a slope-scaling heuristic to linearize the fixed costs, and then to use column generation to solve the resulting linear programs without enumerating all feasible in-trees. The slope-scaling approach proposed here is very similar to the generic approach presented in Sect. 3.5.

Given a set of slope-scaling linearization factors, it can be shown that the empty balancing problem is independent of the tree selection variables. Thus, the slope-scaling linear programming problem can be decomposed into a simple linear subproblem for empty balancing (with fractional empties) and another for selecting in-trees for each destination. The empty problem needs only to be solved once. The in-tree selection linear subproblem is also simple, and in fact can be solved by inspection by choosing for each destination the tree with the smallest cost coefficient; thus, the in-tree selection problem results in integer solutions.

All of these observations motivate the following approach for solving the slope-scaling LP for a given set of linearization factors. Given an initial set of possible in-trees ℓ with at least one per destination, the in-tree selection LP is solved (by inspection). The resulting dual variables associated with the tree selection constraints are used by an integer program that is solved for each destination d that seeks to find (if possible) a new in-tree with negative reduced cost. This in-tree selection integer program will not be described in more detail here, but it should be noted that it is based on enumerating sets of possible time-expanded paths into the timed node copies of destination d in $\mathscr{N}$ and using binary variables to select a joint set that satisfies the in-tree property. Once the LP is solved to optimality via this column generation procedure, the slope scaling multipliers are updated using the approach described in this chapter and the LP is solved iteratively until a stopping criterion is met.

This time-expanded flow planning approach was implemented in practice for a major US LTL carrier. The carrier at the time operated nearly 150 terminals, and thus needed to solve large-scale instances with up to 725 time-expanded nodes, 30,000 arcs, and 680 time-space destination commodities. Computational results demonstrate that the algorithm was able to find improvements of 4–5% in flow plan costs when compared to the carrier's base flow plan.

4.2 Time-Expanded Models for LTL Flow and Load Planning

Modern LTL operators often provide services between many origin-destination terminal pairs with rapid transit times, often as short as overnight or 2 days. Even with such tight time constraints, it still may make sense to transfer freight multiple times at intermediate hubs. In such situations, the *timing* of consolidation is critical: will the cross-docking occur in the overnight hours, or during the day, or in the evening with freight picked up that day from the local operations? Models that attempt to determine flows and build a schedule of loads need detailed timing to make these decisions accurately.

Consider then a modeling framework for flow and load planning problems that includes the following features: (1) detailed time-space network modeling, where nodes denote (terminal, time) pairs and arcs denote timed movements, with fine time discretization (with multiple decision epochs in a day for each terminal) representing a single week of activity, (2) integrated consideration of loaded and empty trailer movements, and (3) support for flexible plans that use an in-tree flow plan structure but do not require the same trees or schedule of loads every day of the week.

We now specify a path-based flow and load planning model with these features. Each commodity k now specifies an origin and destination terminal as usual but additionally is associated with a specific day-of-week and a latest delivery day and time at the destination; since originating freight arrives primarily from the pickup-and-delivery operation, it is assumed that it all becomes available simultaneously in the evening of each day (for example, 7 p.m.). Paths for each such commodity are now sequences of load dispatches and terminal holding arcs that denote waiting at terminals. In-tree structure is enforced for terminal nodes regardless of the time-of-day of individual dispatches. It is not difficult to allow a different in-tree structure for destination d at different times during the planning horizon (for example, a new in-tree can be specified for each day-of-week separately). For simplicity of exposition, in this chapter we present the model where a single in-tree per destination terminal

d is specified that persists for the entire planning horizon; the extension where the tree arc choices can vary by day-of-week is not very different.

For the time-expanded network formulation, let $\mathcal{N}$ and $\mathcal{A}$ again represent the set of time-space (terminal, time) nodes and time-space arcs (denoting the timed trailer dispatch lanes), respectively. The set of time-space arcs also includes freight holding arcs between consecutive nodes at the same terminal. Note that since loads and empties are now planned at specific times (denoted with decision variables τ_a), this becomes a load planning model. Commodity demand q_k is measured in fractional trailers, and the set of time-space paths of commodity k is denoted with $\mathcal{P}(k)$. Consistently with the notation used throughout, N and A refer to the geographic terminal locations and the geographic direct lanes connecting terminals, respectively. Each commodity is required to follow a single time-space path from origin to destination. Since in-tree variables are defined on the geographic network, it could be possible to have multiple paths for the same commodity that satisfy the in-tree requirement but this is prevented with the single-path constraints. The translation function $l(a)$ maps a time-space dispatch arc $a \in \mathcal{A}$ to its direct geographic lane $l(a) \in A$. Finally, $\Delta^+(u)$ is the set of all direct lanes $(u, j) \in A$. Then, we have the following time-space formulation for the path-based flow and load planning problem, which we denote PFLP-TS:

$$\text{minimize} \quad \sum_{a \in A} d_a \tau_a + \sum_{k \in K} \sum_{p \in \mathcal{P}(k)} h_p q_k x_p^k \tag{14.48}$$

subject to

$$\sum_{p \in \mathcal{P}(k)} x_p^k = 1 \qquad \forall k \in K \tag{14.49}$$

$$\sum_{l \in \Delta^+(u)} y_l^d \le 1 \qquad \forall u \in N, \ \ \forall d \in N \tag{14.50}$$

$$\sum_{p \in \mathcal{P}(k): a \in p} x_p^k \le y_{l(a)}^{d^k} \qquad \forall k \in K, \forall a \in \mathcal{A} \tag{14.51}$$

$$\sum_{k \in K} \sum_{p \in \mathcal{P}(k): a \in p} q_k x_p^k \le \tau_a \qquad \forall a \in \mathcal{A} \tag{14.52}$$

$$\sum_{a \in \delta^+(i)} \tau_a - \sum_{a \in \delta^-(i)} \tau_a = 0 \qquad \forall i \in \mathcal{N} \tag{14.53}$$

$$x_p^k \in \{0, 1\} \qquad \forall k \in K, \ \ \forall p \in \mathcal{P}(k) \tag{14.54}$$

$$y_l^d \in \{0, 1\} \qquad \forall d \in N, \ \ \forall l \in \Delta^+(u), u \in N \tag{14.55}$$

$$\tau_a \in \mathbb{Z}_+ \qquad \forall a \in \mathcal{A} \tag{14.56}$$

Solving formulation PFLP-TS exactly, or its extension when in-trees are allowed to differ over time, is generally not possible for planning instances typically found in practice. Smaller regional LTL carriers with just a handful of terminals may lead to instances that can be solved by modern integer programming software, especially if care is taken to manage the number of feasible time-space paths for each commodity included in the sets $\mathscr{P}(k)$.

Larger carriers with hundreds of terminals can easily lead to instances with 500,000 time-space arcs, 50,000 commodities, and millions of feasible time-space commodity paths. Rather than trying to solve these integer programs exactly, then, we instead present *integer-programming-based local search* heuristics for finding solutions. In these approaches, all neighbors in the local search neighborhood are identified by feasible solutions to a smaller integer program, and the search for an improving solution is performed by solving that integer program.

In local search, we begin with a feasible incumbent solution (in this case, a set of feasible decision variables x, y, and τ) and search for a neighboring solution (or simply *neighbor*) whose cost is less than the cost of the incumbent. If such a solution is found, it becomes the new incumbent and we continue the search. The search can be terminated when a certain number of iterations has been performed, a time limit has been reached, or no additional improving neighbors can be found.

For this flow and load planning problem, let us first consider neighbors that are defined by reoptimizing the in-tree for each single specific destination terminal (or terminal-delivery day) d. If the in-tree plan selection variables y for all other destinations are fixed at their current values along with the timed dispatch paths for commodities inbound to those locations specified by x, a restricted IP can be solved to search for new values only for variables y^d and x^k for commodities k where $d^k = d$; each such solution is considered a neighbor. Trailer flow variables τ are never fixed in this approach. Note that one idea used to direct to the search toward promising destinations d is to only consider destination terminals for which a large amount of freight is destined; other approaches can be considered that prioritize the reoptimization of larger terminals more frequently.

More specifically, given a current feasible solution $(\bar{x}, \bar{y}, \bar{\tau})$ at some iteration of the search algorithm, $PFLP{-}TS(d)$ is defined by adding fixing constraints (14.57) and (14.58) to the original formulation (14.48)–(14.56):

$$y_l^u = \bar{y}_l^u \quad \forall u \in N : u \neq d \tag{14.57}$$

$$x_p^k = \bar{x}_p^k \quad \forall k \in K : d^k \neq d \tag{14.58}$$

Erera et al. (2013a) uses this time-expanded network model and solution technique to study flexible flow and load plan designs, including (1) a day-differentiated plan, where an in-tree structure is preserved but the trees are not required to be the same each day of the week, (2) a same-path plan, where the tree requirement is dropped but the freight between two terminals has to follow the same sequence of terminals every day, and (3) an unrestricted plan, where freight is routed without the tree restriction or the same path requirement. Computational experiments demonstrate that high-quality solutions to large-scale problem instances, which represent actual freight volumes transported by the super-regional LTL carrier Saia, can be obtained when limiting the restricted neighborhood search IPs to a solution time limit of 90 s. The study reports cost savings (relative to the initial load plan provided by the carrier) of approximately 4% for traditional load plans and approximately 6.5% for day-differentiated load plans by running the local search for 6 h.

Other IP-based local search neighborhoods may be promising for this problem. One idea is to use an integer program to attempt to attract flow to a specific lane $l \in A$ or to drive flow off of l. Such lane-based neighborhoods may benefit from the fact that they can adjust the in-trees for multiple destinations during a single search iteration; this may be especially important when in-trees for multiple nearby terminals all need to be adjusted simultaneously to remove enough flow from certain time-space dispatch arcs to reduce cost. Another potentially useful idea is to not include the trailer balance constraints (14.53) when solving the neighborhood search integer programs; when the time-space networks are large, there are large numbers of these constraints which can slow the search IP significantly. Instead, we might specify lower bounds on the number of trailers dispatched on some time-space arcs when they have been identified in earlier iterations as useful backhaul lanes for returning empties to outbound-heavy terminals.

Consider an alternative IP-based search approach for solving PFLP-TS where at each iteration, the type of neighborhood (*attract* or *reduce* freight) is chosen as well as a specific geographic lane $l \in A$. Given a lane and a neighborhood type, a set of destination terminals D' is identified whose in-trees may be affected by attracting or removing flow from lane l. For each of these destinations $d \in D'$, a new in-tree is determined via a heuristic as an option to replace the current in-tree to d. The *NewOrOldTree* IP is then solved to search the neighborhood, where for each $d \in D'$ a decision is made to leave its tree unchanged or to adopt the new tree while also choosing new commodity time-space paths compatible with the selections. All in-trees and commodity paths for destinations $d \notin D'$ are fixed and thus remain unchanged.

The success of this approach clearly depends also on the methods used to generate new in-trees. One method useful for the *attract* neighborhood is similar to the IFOL-0 procedure of Powell (1986) described earlier in this chapter. Determining new trees for the *reduce* neighborhood is more complicated, since determining an appropriate new in-tree for d that excludes lane l requires selecting from potentially many feasible choices.

We now describe the *NewOrOldTree* IP, given as follows:

$$\text{minimize} \quad F + \sum_{a \in \mathscr{A}'} c_a \tau_a + \sum_{k \in K'} \sum_{p \in \mathscr{P}'(k)} h_p q_k x_p^k \tag{14.59}$$

subject to

$$\textstyle\sum_{p \in \mathscr{P}'(k)} x_p^k = 1 \qquad \forall k \in K' \tag{14.60}$$

$$x_p^k \le 1 - z_{d^k} \quad \forall k \in K', \quad \forall p \in \mathscr{P}'(k, OldTree(d^k)) \tag{14.61}$$

$$x_p^k \le z_{d^k} \quad \forall k \in K', \quad \forall p \in \mathscr{P}'(k, NewTree(d^k)) \tag{14.62}$$

$$\textstyle\sum_{k \in K'} \sum_{p \in \mathscr{P}'(k): a \in p} q_k x_p^k + f_a \le \tau_a \quad \forall a \in \mathscr{A}' \tag{14.63}$$

$$z_{d^k} \in \{0, 1\} \qquad \forall k \in K' \tag{14.64}$$

$$x_p^k \in \{0, 1\} \qquad \forall k \in K', \ \forall p \in \mathscr{P}' \tag{14.65}$$

$$\tau_a \ge MT_a \qquad \forall a \in \mathscr{A}' \tag{14.66}$$

$$\tau_a \in \mathbb{Z}_+ \qquad \forall a \in \mathscr{A}' \tag{14.67}$$

Note that the search formulation (14.59)–(14.67) is no longer simply a restriction of the original integer program. For each destination $d \in D'$, binary decision variable z_d is used to select the new in-tree or old (current) in-tree to d. Time-space paths are (potentially) changed only for some commodities $k \in K'$, where K' denotes all commodities destined for a terminal in D'; note that the current time-space path for some such commodity k remains feasible if the old in-tree is selected, but it may or may not be feasible if the new in-tree is selected for d^k. Let $\mathscr{A}'$ be the subset of time-space dispatch arcs whose trailer flow might change given D' and the specified in-trees. Let F denote all costs associated with trailer movement on arcs $a \in \mathscr{A} \setminus \mathscr{A}'$ and handling for commodities $k \in K \setminus K'$. Constraints (14.61) and (14.62) ensure compatibility of path selection with the new/old tree selections. Constraint (14.63) ensures that enough trailers move along an arc a to carry the freight assigned to the paths passing through a. Here, f_a denotes the sum of the fractional freight for commodities $k \in K \setminus K'$ that will remain moving on dispatch arc a.

Finally, this local search approach uses a different method to model the impact of empty trailer flows on the flow and load plan. Let MT_a be the current minimum number of trailers that must move on arc a in order to guarantee flow balance; this quantity is determined by periodically solving an *empty trailer repositioning minimum cost network flow (MCNF)* formulation given the current x and y solution, and then setting $MT_a = \tau_a$ for any arc a on which empty trailers are planned. The idea here is that the best times and locations to move empty trailers are dictated largely by the underlying freight demand and thus do not need to change frequently, so lower bounds can be used to create useful backhaul opportunities that can be exploited when selecting the flow plan and specific loaded dispatches. Of course, it is also necessary to solve the empty balancing problem at the end of the final iteration to ensure that the final trailer load plan is balanced.

This heuristic is tested in Lindsey et al. (2016) on large-scale problem instances with numerous algorithmic configurations, where a configuration is defined by the rules used in each iteration to (1) decide whether to search an *attract* or a *reduce* neighborhood, and (2) choose the lane l to be used to generate the neighborhood. Computational experiments demonstrate that the approach is effective at generating high-quality solutions in reasonable computation times. Cost reductions from the base flow and load plan of the partner LTL carrier were found in the range of 6–7%.

4.3 Dynamic Discretization Discovery

Solving flow and load planning models that use time-expanded networks to within reasonable provable optimality gaps for the large-scale instances found in practice has been beyond reach for a long time.

However, a novel paradigm, dynamic discretization discovery, has emerged recently as a way to effectively and efficiently find optimal or near-optimal solutions to models using time-expanded networks (Boland et al. 2017). Dynamic discretization discovery allows the solution of such models on a fine discretization without ever fully constructing it. The paradigm has three main components:

- The design of time-indexed IP models based on a *partial discretization of time*, that are efficiently solvable in practice and that yield lower bounds, upper bounds, or exact solutions;
- The design of algorithms that *dynamically discover* partial discretizations, *i.e.,* algorithms that can "refine" a partial discretization of time in order to strengthen the quality of a time-indexed IP model; and
- The design of algorithms that efficiently solve time-indexed IP models.

The latter is stated for completeness sake. In many situations, the use of a standard (commercial or open source) IP solver suffices.

A partially time-expanded network $\mathcal{D}_{\mathcal{T}} = (\mathcal{N}_{\mathcal{T}}, \mathcal{A}_{\mathcal{T}})$ is derived from subsets of the time points that could be modeled at each terminal node. Specifically, we denote the collection of modeled time points as $\mathcal{T} = \{\mathcal{T}_i\}_{i \in N}$, with $\mathcal{T}_i = \{t_1^i, \ldots, t_{n_i}^i\} \subseteq \{1, \ldots, T\}$ representing the time points modeled at terminal node i and T denoting the planning horizon. Given $\mathcal{T}$, the *timed* node set $\mathcal{N}_{\mathcal{T}}$ then has a node (i, t) for each $i \in N$ and $t \in \mathcal{T}_i$.

The timed arc set of a partially time-expanded network consists of arcs of the form $((i, t), (j, \bar{t}))$ where $(i, j) \in A$, $t \in \mathcal{T}_i$, and $\bar{t} \in \mathcal{T}_j$. Note that arc $((i, t), (j, \bar{t}))$ does *not* have to satisfy $\bar{t} = t + t_{ij}$, where t_{ij} is the travel time from terminal node i to terminal node j. In fact, the flexibility to introduce arcs $((i, t), (j, \bar{t}))$ with a travel time that is different from the actual travel time t_{ij} is an essential feature of the partially time-expanded networks, and provides a mechanism to control both the size of the time-expanded network and the approximation properties of the IP model based on it.

Now consider a partially time-expanded version of the generic model presented in Sect. 3.1, *i.e.,*

$$\text{minimize} \quad \sum_{k \in K} \sum_{a \in \mathscr{A}_{\mathscr{T}}} c_a x_a^k + \sum_{a \in \mathscr{A}_{\mathscr{T}}} d_a \tau_a - \sum_{i \in N} \sum_{k \in K \mid o_k = i} h_i q_k \tag{14.68}$$

subject to

$$\sum_{a=((i,t),(j,\bar{t})) \in \mathscr{A}_{\mathscr{T}}} x_a^k - \sum_{a=((j,\bar{t}),(i,t)) \in \mathscr{A}_{\mathscr{T}}} x_a^k = \begin{cases} q_k & \text{if } i = o_k,\ t = e_k \\ -q_k & \text{if } i = d_k,\ t = l_k \\ 0 & \text{otherwise} \end{cases} \forall k \in K, \quad \forall i \in N \tag{14.69}$$

$$x_a = \sum_{k \in K} x_a^k \qquad \forall a \in \mathscr{A}_{\mathscr{T}} \tag{14.70}$$

$$x_a \le Q\tau_a \qquad \forall a \in \mathscr{A}_{\mathscr{T}} \tag{14.71}$$

$$x_a^k \ge 0 \qquad \forall k \in K,\ \forall a \in \mathscr{A}_{\mathscr{T}} \tag{14.72}$$

$$\tau_a \ge 0 \text{ and integer} \qquad \forall a \in \mathscr{A}_{\mathscr{T}} \tag{14.73}$$

Observe that if the time discretization is complete, *i.e.,* $\mathscr{T}_i = \{1, \dots, T\}$ for $i \in N$, and $\mathscr{A}_{\mathscr{T}}$ consists of all arcs $((i,t),(j,\bar{t}))$ with $\bar{t} = t + t_{ij}$ for $i, j \in N$ and $t = 1, \dots, T - t_{ij}$, together with all arcs $((i,t),(i,t+1))$ for $i \in N$ and $t = 1, \dots, T-1$, representing the possibility to wait at terminal node i, then the time-expanded network is acyclic and the above formulation is an exact formulation for the flow planning model.

By choosing $\mathscr{N}_{\mathscr{T}}$ and $\mathscr{A}_{\mathscr{T}}$ carefully, the model may be guaranteed to provide either a lower or an upper bound on the optimal value of flow planning model.

To obtain a lower bound, the concept of a "short" arc is helpful: $((i,t),(j,\bar{t})) \in \mathscr{A}_{\mathscr{T}}$ is *short* if $\bar{t} \le t + t_{ij}$. Three conditions, together, guarantee a lower bound from the IP: (i) $(o_k, e_k) \in \mathscr{N}_{\mathscr{T}}$ and $(d_k, l_k) \in \mathscr{N}_{\mathscr{T}}$ for all $k \in K$, (ii) for all $(i,t) \in \mathscr{N}_{\mathscr{T}}$ and all $j \in N$ with $t + t_{ij} \le l_j$, there exists a $\bar{t}$ with $((i,t),(j,\bar{t})) \in \mathscr{A}_{\mathscr{T}}$, and (iii) every arc in $\mathscr{A}_{\mathscr{T}}$ is short. Note that if the time discretization at terminal node j is quite coarse, it may be that $\bar{t}$ is less than t, suggesting travel backwards in time! Nevertheless, good lower bounds can result. It can be proved that the best such lower bound is obtained by setting $\bar{t} = \max\{t' : t' \le t + t_{ij},\ (j,t') \in \mathscr{N}_{\mathscr{T}}\}$, and permitting no other arc from (i,t) to j to be included in $\mathscr{A}_{\mathscr{T}}$. A partially time-expanded network created in this way has the *longest-arc* property.

A condition that guarantees an upper bound from the IP, provided the IP is feasible, is that all arcs are "long": $((i,t),(j,\bar{t})) \in \mathscr{A}_{\mathscr{T}}$ is *long* if $\bar{t} \ge t + t_{ij}$.

The fundamental idea underlying the dynamic discretization discovery paradigm is to always work with a partial discretization of time so as to ensure that the resulting models can be solved efficiently, but to guarantee that, upon termination of the algorithm, an optimal (continuous-time) solution is (or can be) produced. That is, the idea is to solve a sequence of small IPs, rather than a single large IP.

Thus, whenever the lower-bound IP model does not generate a feasible (and hence optimal) solution, its solution uses some timed arc that is too short. By refining the discretization at the terminals node at the head of that arc, the timed arc that is too short can be made to "disappear", *i.e.,* not be present in the network associated with the refined partial discretization. If the longest-arc property is enforced for the partially time-expanded network constructed at each iteration, the timed arc that is too short, $((i, t), (j, \bar{t}))$ say, can be removed simply by adding $\hat{t}$ to $\mathscr{T}_j$ for any $\hat{t}$ satisfying $\bar{t} < \hat{t} \leq t + \tau_{ij}$. The effect will be to *lengthen* the timed arc, to $((i, t), (j, \hat{t}))$. The natural choice is to take $\hat{t} = t + t_{ij}$.

Excellent computational results for medium-sized instances of the flow and load planning model have been obtained with an interval-based variant of the dynamic discretization discovery algorithm outlined above (Marshall et al. 2020); instances derived from the western operations of a US carrier with about 15 terminals, about 100 load arcs connecting terminals, and about 450 commodities are solved to within 1% of (proven) optimality in about 10 min.

5 Bibliographical Notes

In this section, we provide more detail about specific important papers in consolidation trucking service network design. First, we will discuss some important papers focusing on exact solution approaches useful for flow planning models. Next, we review important papers in the chronology of flow planning for consolidation trucking. We then review papers on flow and load planning and related network design papers that use time-expanded networks. Finally, some discussion of problems downstream from load planning will also be reviewed. These notes are not meant to be a complete and comprehensive chronology, but should provide the reader with a useful initial overview of some of the more important papers in the literature.

The generic arc-based trucking network flow planning model is a multi-commodity capacitated fixed-charge network design (MCND) problem (see e.g., Crainic, 2000). Early work describes Lagrangian approaches for computing lower bounds for cases when the capacity that can be installed on each arc is bounded (see e.g., Crainic et al., 2001). Recently, Chouman et al. (2017) provides an excellent summary of recent exact approaches for this problem class, including those described in Frangioni and Gendron (2009) and Raack et al. (2011). Furthermore, the paper outlines the components of effective cutting plane algorithms for the problem, including one that relies on introducing violated strong inequalities of the form $x_{ij}^k \leq q_k \tau_{ij}$ to improve lower bounds. Evidence is provided that cutting plane algorithms work better with disaggregated commodity representations. Problems solved to optimality or to small gaps are those with at most 100 nodes, 700 arcs, and 400 commodities. Atamtürk and Gunluk (2017) provides a useful review of approaches for the construction of useful classes of valid inequalities for capacitated network design problems, including earlier work in Atamtürk (2002) and Atamtürk and Rajan (2002). Slope-scaling heuristics for these design problems are introduced in Crainic et al. (2004).

An early and important stream of research on flow planning for LTL trucking consolidation networks was initiated in Powell and Sheffi (1983) and Powell (1986); related work using similar flat network models is also covered in Powell and Sheffi (1989) and Powell and Koskosidis (1992). It is important to note here that the work was described as *load planning* (for example, in the title of the 1983 paper), but in the context of these definitions used in this chapter it is best to classify this work as focused on flow planning. Powell (1986) introduces the in-tree flow planning problem for LTL carriers and develops a detailed local search solution heuristic for the problem. Follow-on work in Powell and Koskosidis (1992) constrains plans further by clustering EOL terminals to a primary breakbulk and aligning their flow plans. This paper also presents refined solution approaches to the routing subproblem, including a gradient-based approach for finding primal feasible solutions and subgradient optimization and dual ascent approaches that produce lower bounds and enable estimations of optimality gaps.

The heuristic developed in these papers was implemented originally for flow planning at the U.S. LTL carrier Ryder Truck Lines in an interactive planning system known as APOLLO (Advanced Planner Of LTL Operations). Development continued at Yellow Freight within a system known as SYSNET; Bell et al. (2003) reports that SYSNET was still in use, in an updated version, at Yellow Freight over a decade later. The ideas in these systems were then sold broadly to LTL carriers by the Princeton Transportation Consulting Group (and later Manhattan Associates) within the SuperSPIN system. The software was reportedly used by every major national and regional U.S. LTL carrier in the 1990s and remained in use for nearly 25 years afterwards. Braklow et al. (1992) describes a case study where the software was successfully implemented at Yellow Freight System, and a reduction in number of end-of-line terminals resulted in higher freight density and therefore reduced handling costs and improved service level.

Other authors in this time period developed LTL flow planning models. Roy and Delorme (1989) introduces *NETPLAN*, a nonlinear mixed-integer network optimization model that simultaneously considers flow planning and empty rebalancing using a path-based model. The objective function minimizes the total transportation and consolidation costs, with penalties for overutilization of trailer capacity and failure to meet service standards. An iterative solution methodology (introduced in Crainic and Rousseau 1986) is used to solve the problem. This study, along with Crainic and Roy (1988) and Roy and Crainic (1992), tests the approaches using data from two large Canadian LTL companies and shows that both service offerings could be expanded reliably while also reducing total operating costs.

More recently, Meuffels et al. (2009) addressed a ground transportation consolidation problem for the express package industry using similar network design ideas. This work was conducted for TNT Express, and it is part of a larger group of operations research projects described in Fleuren et al. (2013). In the paper, relatively small tactical express networks are considered for consolidation optimization using flat networks. Additionally, time-feasible schedules for the vehicle fleet are also determined given the consolidation plan.

Time-expanded network models for trucking service network design do not appear in the literature until the 2000s. Prior to this work, some use of models of this type was documented for express package service providers that typically use air and truck movements. Work in this area is described in Barnhart and Schneur (1996), Kim et al. (1999), Barnhart et al. (2002), and Armacost et al. (2002). Decisions in these models include the timed routes of aircraft from (potentially) different fleet types and ground truck transfer decisions to enable service-feasible transfer of packages from origins to destinations. In these papers, integer programming optimization models that use path variables are developed and solved, often relying on column generation for solving large-scale linear programming relaxations.

With increasing demand for faster and time-definite freight transfer due to changing customer service expectations driven in part by the package express industry, LTL carriers now need to plan networks with tighter service guarantees. Jarrah et al. (2009) is the first to consider LTL flow planning with a time-expanded network that allows modeling of explicit service commitments (measured in transit days) to customers that is solved with a slope-scaling approach. The path-based model creates an in-tree flow plan with empty trailer balancing by considering a planning week with a single time-space node on each weekday; in this way, it is not a detailed load dispatch planning model and may still overestimate consolidation opportunities.

Erera et al. (2013a) and Lindsey et al. (2016) model large-scale detailed flow and load planning problems for LTL carriers, following advice from Powell (1986): "Ideally the problem should be formulated as a detailed scheduling problem where the scheduled departure of each tractor would reflect not only a decision that balanced transportation and handling costs but also the actual level of service constraints for each shipment being carried." Both papers introduce integrated flow and load planning integer programming models that use a path-based formulation on a time-space network, and both solve the models using different IP-based local search techniques. Erera et al. (2013a) defines the local search neighborhood by restricting the base integer program to only change flow plan and freight routing variables for a single destination d each iteration, while Lindsey et al. (2016) considers neighborhoods defined by adjusting many in-trees simultaneously (and associated time-space freight paths) to add or remove flow from individual geographic lanes each iteration.

The most modern flow and load planning work in the research literature has focused on developing better approaches for determining the discretization of time-expanded networks. An important new idea is dynamic discretization discovery, where iterative techniques are used to expand the size of a time-expanded network representation of a consolidation trucking network for flow and load planning. A dynamic discretization discovery algorithm for the load plan design problem, which enforces path selection from a set of candidate paths and an in-tree structure, is presented in Hewitt (2019). Follow-on research is developed in Marshall et al. (2020), and unpublished research in this area focuses on adapting principles of dynamic discretization discovery to pragmatic heuristic solution approaches that are often required by large-scale carrier instances.

Other research papers have focused on service network design problems in trucking that lie downstream of flow planning and load planning. One example is the load and dispatch problem considered in Cohn et al. (2007) and Root and Cohn (2008). These papers consider ground package trucking operations for a large carrier given a fixed flow plan. The goal is to build loads and trailer dispatches that meet service requirements, considering both single-trailer and double-trailer combination dispatches using 28-foot pup trailers. Set partitioning models with composite variables that define complete paths for one or more trailers are developed.

Crainic and Roy (1992) focuses on driver scheduling and presents a modeling framework for generating regular driver routes for LTL carriers, given a flow and load plan. The model takes into consideration operational aspects of driver route generation, such as cyclic routes, regular and overtime costs, and maximum permitted duty and working times. The model is solved in three stages: segment generation, route generation, and route selection. Segments are used as the main elements in a set covering model, and a column generation approach is developed. More recently, Erera et al. (2013b) also investigates driver scheduling given a flow and load plan. The paper first introduces the *load plan scheduling problem*, which develops a detailed operational schedule (timed schedule of trailer, tractor, and driver dispatches) required to operate a plan. The approach can be used either with a flow plan only that specifies a geographic transfer sequence of freight for each commodity, or with a flow and load plan where timed trailer dispatches have already been planned. When only a flow plan is given, a detailed load plan is first constructed using a heuristic that sequentially assigns commodities to time-space paths of loads by minimizing path *marginal cost* within a GRASP framework (greedy randomized adaptive search procedure). A key insight in this paper is that some constructed trailer dispatches can be shifted in time in order to improve the driver schedule and its cost, when doing so does not impact the feasibility of the consolidation plan. A novel linear programming formulation is presented to maximize the total width of all trailer dispatch time windows such that the load plan remains feasible. Then, using these adjusted time windows, driver tours are constructed serving each trailer dispatch within its newly-expanded time window using a set covering model and a column generation heuristic.

6 Concluding Remarks and Research Directions

Operations research and service network design models have been used quite effectively for improving motor freight consolidation planning over the past 35 years. Initial successes with frequency-based flat network models for flow planning have now been augmented significantly with flow and load planning models that use detailed time-expanded networks and integrate loaded and empty dispatch planning.

Going forward, there are a number of areas that the field can continue to address to improve service network design for trucking. For example, there is a clear need to better integrate driver and trailer resource planning with flow and load planning. Since driver schedules are highly constrained, it would be best to build plans that recognize that all (or most) dispatches will be covered by driver tours. Crainic et al. (2016) develops initial ideas for effective approaches, and work such as that presented in Hewitt et al. (2019) has developed heuristic approaches that may enable these approaches to be deployed on real-world problems of practical scale.

Another important area for investigation is the set of dynamic planning problems that seek to manage and mitigate uncertainty in both customer demand and supply conditions. Early work in Zhang (2010) focuses on initial ideas for dynamic load planning given updated demand information. Recently, UPS has begun investigation of dynamic load building given primary and alternate freight routing paths for its LTL freight division; fast heuristic approaches are developed in Ridouane et al. (2020) for allocating inbound shipments to scheduled trailer capacity in an effort focused on successfully transferring freight to meet service requirements while also identifying potential scheduled trailerloads for cancellation and cost savings.

Finally, it is also important to extend trucking service network design models to incorporate uncertainty in freight demand. Given that the deterministic planning problems are already very difficult to solve, this is a particular challenge. Early important work in this direction is presented in Lium et al. (2009), and this paper shows using small generic examples that the structure of service designs identified when explicitly modeling uncertainty can be quite different from the designs that result from deterministic models. Baubaid et al. (2018) has more recently considered the stochastic planning problem of setting primary and alternate flow plan paths specifically for LTL freight networks; the paper defines the p-alt planning problem under demand uncertainty, p limits the number of outbound terminals that freight destined to d can flow to next. A 1-alt design represents a standard in-tree flow plan. The approach uses sample average approximation to find p-alt plans that minimize (an approximation of) expected costs (including failure penalties when capacity is not available). Unfortunately, adding multiple scenarios and linking constraints to the already-difficult multi-commodity fixed-charge network design problem severely limits the size of the problems that can currently be addressed by this approach. It may be necessary to develop service network design problems for trucking networks that rely on simpler, approximate problem representations when planning under uncertainty and then to test and refine those designs with more detailed models.

References

Alumur, S., & Kara, B. Y. (2008). Network hub location problems: The state of the art. *European Journal of Operational Research, 190*(1), 1–21.

Armacost, A. P., Barnhart, C., & Ware, K. A. (2002). Composite variable formulations for express shipment service network design. *Transportation Science, 36*(1), 1–20.

Atamtürk, A. (2002). On capacitated network design cut-set polyhedra. *Mathematical Programming, 92*(3), 425–437.

Atamtürk, A., & Gunluk, O. (2017). Multi-commodity multi-facility network design. www.1707.03810.

Atamtürk, A., & Rajan, D. (2002). On splittable and unsplittable flow capacitated network design arc–set polyhedra. *Mathematical Programming, 92*(2), 315–333.

Barnhart, C., & Schneur, R. R. (1996). Air network design for express shipment service. *Operations Research, 44*(6), 852–863.

Barnhart, C., Krishnan, N., Kim, D, & Ware, K. (2002). Network design for express shipment delivery. *Computational Optimization and Applications, 21*(3), 239–262.

Baubaid, A., Boland, N., & Savelsbergh, M. (2018). Dealing with demand uncertainty in service network and load plan design. In W. J. van Hoeve (Ed.), *Integration of constraint programming, artificial intelligence, and operations research. CPAIOR 2018. Lecture notes in computer science* (vol. 10848, pp. 63–71). Berlin: Springer.

Bell, P. C., Anderson, C. K., & Kaiser, S. P. (2003). Strategic operations research and the Edelman prize finalist applications 1989–1998. *Operations Research, 51*(1), 17–31.

Boland, N., Hewitt, M., Marshall, L., & Savelsbergh, M. (2017). The continuous-time service network design problem. *Operations Research, 65*(5), 1303–1321.

Braekers, K., Ramaekers, K., & van Nieuwenhuyse, I. (2016). The vehicle routing problem: State of the art classification and review. *Computers and Industrial Engineering, 99*, 300–313.

Braklow, J. W., Graham, W. W., Hassler, S. M., Peck, K. E., & Powell, W. B. (1992). Interactive optimization improves service at Yellow Freight System. *Interfaces, 22*(1), 147–172.

Bureau of Transportation Statistics. (2018). Transportation statistics annual report 2018. Tech. Rep., United States Department of Transportation. Washington, D.C. https://doi.org/10.21949/1502596

Campbell, J. F. (1994). Integer programming formulations of discrete hub location problems. *European Journal of Operational Research, 72*(2), 387–405.

Cattaruzza, D., Absi, N., Feillet, D, & González-Feliu, J. (2017). Vehicle routing problems for city logistics. *EURO Journal on Transportation and Logistics, 6*(1), 51–79.

Chouman, M., Crainic, T. G., & Gendron, B. (2017). Commodity representations and cut-set-based inequalities for multicommodity capacitated fixed-charge network design. *Transportation Science, 51*(2), 650–667.

Cohn, A., Root, S., Wang, A., & Mohr, D. (2007). Integration of the load-matching and routing problem with equipment balancing for small package carriers. *Transportation Science, 41*(2), 238–252.

Crainic, T. G., Frangioni, A., & Gendron, B. (2001). Bundle-based relaxation methods for multicommodity capacitated fixed charge network design. *Discrete Applied Mathematics, 112*, 73–99.

Crainic, T. G. (2000). Service network design in freight transportation. *European Journal of Operational Research, 122*(2), 272–288.

Crainic, T. G., & Rousseau, J. M. (1986). Multicommodity, multimode freight transportation: A general modeling and algorithmic framework for the service network design problem. *Transportation Research Part B, 20B*(3), 225–242.

Crainic, T. G., & Roy, J. (1988). OR tools for tactical freight transportation planning. *European Journal of Operational Research, 33*(3), 290–297.

Crainic, T. G., & Roy, J. (1992). Design of regular intercity driver routes for the LTL motor carrier industry. *Transportation Science, 26*(4), 280–295.

Crainic, T. G., Gendron, B., & Hernu, G. (2004). A slope scaling/Lagrangean perturbation heuristic with long-term memory for multicommodity capacitated fixed-charge network design. *Journal of Heuristics, 10*(5), 525–545.

Crainic, T. G., Hewitt, M., Toulouse, M., & Vu, D. M. (2016). Service network design with resource constraints. *Transportation Science, 50*(4), 1380–1393.

Daskin, M. S. (2011). *Network and discrete location: Models, algorithms, and applications*. Hoboken: Wiley.

Drezner, Z., & Hamacher, H. W. (2001). Facility location: Applications and theory. Berlin: Springer.

Erera, A., Hewitt, M., Savelsbergh, M., & Zhang, Y. (2013a). Improved load plan design through integer programming based local search. *Transportation Science, 47*(3), 412–427.

Erera, A. L., Hewitt, M., Savelsbergh, M. W., & Zhang, Y. (2013b). Creating schedules and computing operating costs for LTL load plans. *Computers and Operations Research, 40*(3), 691–702.

Farahani, R. Z., Hekmatfar, M., Arabani, A. B., & Nikbakhsh, E. (2013). Hub location problems: A review of models, classification, solution techniques, and applications. *Computers & Industrial Engineering, 64*(4), 1096–1109.

Fleuren, H., Goossens, C., Hendriks, M., Lombard, M., Meuffels, I., & Poppelaars, J. (2013). Supply chain-wide optimization at TNT express. *Interfaces, 43*(1), 5–20.

Frangioni, A., & Gendron, B. (2009). 0–1 reformulations of the multicommodity capacitated network design problem. *Discrete Applied Mathematics, 157*(6), 1229–1241.

Golden, B. L., Raghavan, S., & Wasil, E. A. (2008). *The vehicle routing problem: Latest advances and new challenges* (vol. 43). Berlin: Springer.

Hewitt, M. (2019). Enhanced dynamic discretization discovery for the continuous time load plan design problem. *Transportation Science, 53*(6), 1731–1750.

Hewitt, M., Crainic, T. G., Nowak, M., & Rei, W. (2019). Scheduled service network design with resource acquisition and management under uncertainty. *Transportation Research Part B: Methodological, 128*, 324–343.

Jarrah, A. I., Johnson, E., & Neubert, L. C. (2009). Large-scale, less-than-truckload service network design. *Operations Research, 57*(3), 609–625.

Kim, D., Barnhart, C., Ware, K., & Reinhardt, G. (1999). Multimodal express package delivery: A service network design application. *Transportation Science, 33*(4), 391–407.

Lindsey, K., Erera, A., & Savelsbergh, M. (2016). Improved integer programming-based neighborhood search for less-than-truckload load plan design. *Transportation Science, 50*(4), 1360–1379.

Lium, A. G., Crainic, T. G., & Wallace, S. W. (2009). A study of demand stochasticity in service network design. *Transportation Science, 43*(2), 144–157.

Love, R. F., Morris, J. G., & Wesolowsky, G. O. (1988). *Facilities location*. New York: Elsevier.

Marshall, L., Boland,. N., Savelsbergh, M., & Hewitt, M. (2021). Interval-based dynamic discretization discovery for solving the continuous-time service network design problem. *Transportation Science, 55(1), 29–51.*

Meuffels, I., Fleuren, H., Cruijssen, F., & Dam, E. (2009). Enriching the tactical network design of express service carriers with fleet scheduling characteristics. *Flexible Services and Manufacturing Journal, 22*, 3–35.

Mirchandani, P., & Francis, R. (1990). *Discrete location theory. Wiley-Interscience series in discrete mathematics and optimization*. Hoboken: Wiley.

O'Kelly, M. E. (1986). The location of interacting hub facilities. *Transportation Science, 20*, 92–106.

Powell, W. B. (1986). A local improvement heuristic for the design of less-than-truckload motor carrier networks. *Transportation Science, 20*(4), 246–257.

Powell, W. B., & Koskosidis, I. A. (1992). Shipment routing algorithms with tree constraints. *Transportation Science, 26*(3), 230–245.

Powell, W. B., & Sheffi, Y. (1983). The load planning problem of motor carriers: Problem description and a proposed solution approach. *Transportation Research Part A: General, 17*(6), 471–480.

Powell, W. B., & Sheffi, Y. (1989). Design and implementation of an interactive optimization system for network design in the motor carrier industry. *Operations Research, 37*(1), 12–29.

Psaraftis, H. N., Wen, M., & Kontovas, C. A. (2016). Dynamic vehicle routing problems: Three decades and counting. *Networks, 67*(1), 3–31.

Raack, C., Koster, A. M., Orlowski, S., & Wessäly, R. (2011). On cut-based inequalities for capacitated network design polyhedra. *Networks, 57*(2), 141–156.

Ridouane, Y., Herszterg, I., Boland, N., Erera, A., & Savelsbergh, M. (2020). Near real-time loadplan adjustments for less-than-truckload carriers. Optimization Online. http://www.optimization-online.org/DB_HTML/2020/01/7565.html

Root, S., & Cohn, A. (2008). A novel modeling approach for express package carrier planning. *Naval Research Logistics, 55*, 670–683.

Roy, J., & Crainic, T. G. (1992). Improving intercity freight routing with a tactical planning model. *Interfaces, 22*(3), 31–44.

Roy, J., & Delorme, L. (1989). NETPLAN: A network optimization model for tactical planning in the less-than-truckload motor-carrier industry. *INFOR Journal, 27*(1), 22–35.

Savelsbergh, M., & van Woensel, T. (2016). 50th anniversary invited article–city logistics: Challenges and opportunities. *Transportation Science, 50*(2), 579–590.

Snyder, L. V. (2006). Facility location under uncertainty: A review. *IIE Transactions, 38*(7), 547–564.

Zhang, Y. (2010). Advances in LTL load plan design. Ph.D. Thesis, Georgia Institute of Technology.

Chapter 15
Liner Shipping Network Design

Marielle Christiansen, Erik Hellsten, David Pisinger, David Sacramento, and Charlotte Vilhelmsen

1 Introduction

Maritime transportation enables transportation of large volumes at relatively low costs and is therefore a fundamental part of the world trade. It is estimated that around 90% of world trade today is carried by the international shipping industry. Additionally, efficient port structures make it possible to combine sea transportation with land-based modes of transportation.

Roughly speaking, liner shipping is the service of transporting large volumes of cargo by means of high-capacity vessels that follow regular routes on fixed schedules. Within this type of shipping service, the variety of vessel types can be split into vessels designed for Lift-on/Lift-off (LoLo) operations, and Roll-on/Roll-off (RoRo) operations. In LoLo operations, quay cranes located on the docks are used to load and unload the vessels' cargo, while RoRo vessels are designed to carry cargo that can be rolled on and off the vessels.

In this chapter, we mainly focus on *containerised* liner shipping network design, i.e., networks using LoLo operations, though we also briefly introduce network design for RoRo liner shipping. Although we have tried to cover most of the relevant models and algorithms dealing with containerised liner shipping network design, we have chosen to focus on those that seem to be applicable in practice.

This chapter is organised as follows: In Sect. 2 we give a brief introduction to containerised liner shipping, RoRo liner shipping, and network design. We also

M. Christiansen
Department of Industrial Economics and Technology Management, Norwegian University of Science and Technology, Trondheim, Norway
e-mail: mc@ntnu.no

E. Hellsten · D. Pisinger (✉) · D. Sacramento · C. Vilhelmsen
DTU Management, Technical University of Denmark, Lyngby, Denmark
e-mail: erohe@dtu.dk; dapi@dtu.dk; dsle@dtu.dk; chaan@dtu.dk

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_15

introduce the LINER-LIB test instances for network design in containerised liner shipping. The following sections are focused on containerised liner shipping. In Sect. 3 we discuss the challenges in designing a liner shipping network, and give an introduction to the models and algorithms, which will be presented in the following sections. In Sect. 4 we give an overview of integrated Mixed Integer Programming (MIP) models, while Sect. 5 studies two-stage algorithms where the service generation and the flowing of containers are separated in two steps. Section 6 presents the bibliographical notes of the topics addressed in this chapter. The chapter is concluded in Sect. 7 with a short discussion of future trends and challenges. Finally, an overview of the notation used throughout the chapter is found in Appendix 8. Some parts of this chapter are based on the authors' previous survey work presented in Hellsten et al. (2019) and Christiansen et al. (2020).

2 Overview of Liner Shipping and Liner Shipping Network Design

Maritime transportation is a cost-effective means of transportation, as it offers the transport of large volumes of cargo all around the world. The cargo vessels in the maritime industry are generally divided into the following major categories: general cargo ships, bulk cargo carriers, tankers, container ships and RoRo ships. This section presents the network design for containerised liner shipping, which are based on LoLo operations, as well as a brief description of RoRo liner shipping.

2.1 Containerised Liner Shipping

The liner shipping industry is a vital part of the global economy, constituting one of the cheapest modes of cargo transportation. During the last three decades, the volume of containerised cargo has grown by more than 8% per year, and more than 5200 container vessels were in operation worldwide in 2019. Standard containers come in two different sizes, twenty and forty feet, which have given rise to the standard measures of containerised cargo, *twenty foot equivalent units* (TEU) and *forty foot equivalent units* (FFE). The largest vessels carry more than 20,000 TEU and during 2016, a container volume of around 140,000,000 TEU was estimated to pass through the vast liner shipping network (Unctad 2018, 2019). Clearly, any improvement in the network design in the liner shipping industry will correspond to enormous savings.

The liner shipping industry is built up by so called *services*. A service is a fixed cyclic itinerary, sailed by a number of similar vessels. Services usually have weekly or biweekly departures to add consistency and regularity for the customers. The vessels are operated by shipping companies called carriers, where the largest

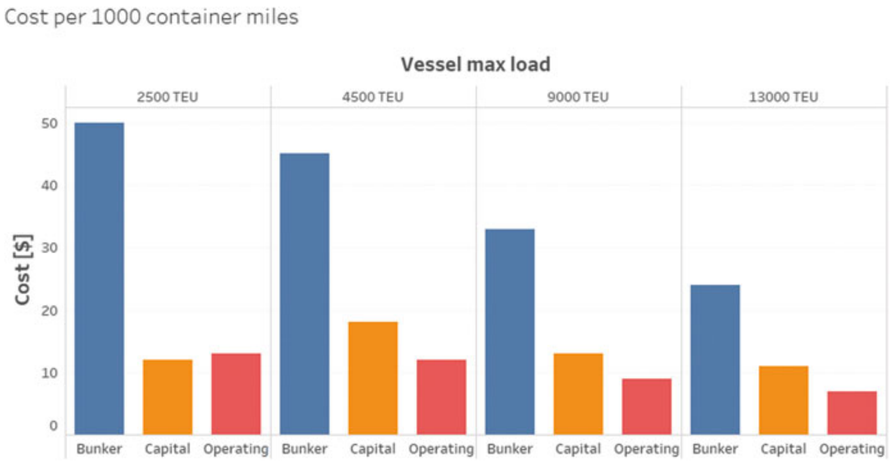


Fig. 15.1 Estimated cost per 1000 container miles for different vessel sizes. The vessels are assumed to sail at 19 knots and the bunker fuel price is estimated as 750 $/tonne. We see that bunker represents the largest cost and that transporting containers on larger vessels requires significantly less fuel. Data has been obtained from Germanische Lloyd (2017)

carriers operate over 600 vessels. The trend is to build ever larger vessels, as they are significantly more energy efficient, see Fig. 15.1. To efficiently utilise those tremendous liner vessels, each region typically has a few larger ports, called *hubs*, where the liner vessels pick up and deliver containers. From the hubs, the containers are then transported to other ports by smaller, more flexible, so called *feeder vessels*. The transfer of a container, from one vessel to another in a port, is called a *transshipment*. Transshipments occur both between larger vessels and smaller vessels, but also between larger vessels when no suitable service connects the origin and destination hub. While transshipments add flexibility, they tend to be costly, as the cargo needs to be unloaded, stored until the arrival of the new vessel and then reloaded again. Finally, to protect the national trade business, many countries forbid foreign carriers to ship cargo between two ports within the country. This is called *cabotage rules*.

The major costs for the carriers are vessel acquirement and bunker fuel. However, other costs, like canal fees, port costs, and transshipment costs, are also highly significant. The fuel consumption is frequently estimated as a cubic function of the speed, as seen in Fig. 15.2. As the speed has such an impact on the fuel consumption, *slow steaming* is often used to reduce the consumption. Especially after the financial crisis in 2008, maritime shipping companies implemented slow steaming policies for cost-cutting purposes. The obvious drawback of slow steaming is that more vessels are required to transport the same amount of cargo and also, that transit times become longer, yielding a lower level of service for the customers. In general, services have two directions, *head-* and *back-haul*, where most of the cargo is transported in the head haul direction. A good example of this is the trade between

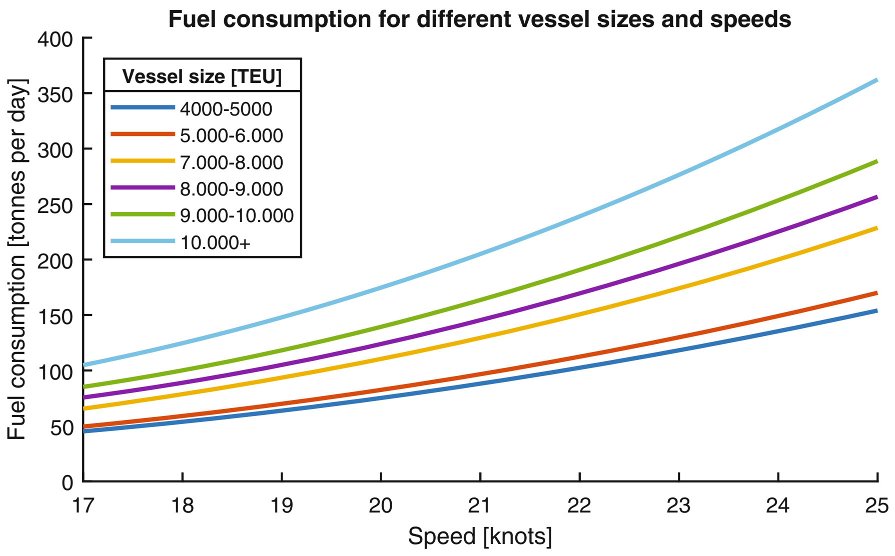


Fig. 15.2 Estimated fuel consumption as a function of steaming speed and vessel size. Data has been obtained from Notteboom and Vernimmen (2009).

Asia and Europe, where most of the goods are delivered from Asia to Europe. In this case, vessels are slow steaming in the back-haul direction where less customers are affected by the increased transit time.

2.2 *Containerised Liner Shipping Network Design*

The Liner Shipping Network Design Problem (LSNDP) can be defined as follows: Given a collection of ports, a fleet of container vessels and a group of origin-destination demands. A set of services is constructed for the container vessels such that the overall operational expenses are minimised, while ensuring that all demands can be routed through the resulting network from their origin to their destination, respecting the capacity of the vessels.

In the following, we present some notation of the LSNDP that will be used throughout the chapter, introducing the necessary notation when required. A table of notation can be found in Table 15.2 in the Appendix.

The set of ports is denoted by N and represents the set of physical ports in the problem. The set of arcs A represents all possible sailings between ports. The set of commodities is denoted K and for each commodity $k \in K$, there is an origin port o_k, a destination port d_k, as well as a quantity q_k measured in TEU. Furthermore, the corresponding unit-cost for transporting a unit of commodity k through arc $(i, j) \in A$ is defined as c_{ij}^k. Finally, the set V denotes the set of *vessel classes*. For each

$v \in V$ there is a corresponding cargo capacity u_v in number of TEUs, an available fleet quantity m_v, and additional speed limitations and fuel consumption parameters. Furthermore, for convenience, the demand of the commodities in each port $i \in N$ is defined as:

$$\xi_i^k = \begin{cases} q_k & \text{if port } i \text{ is the origin port of commodity } k \\ -q_k & \text{if port } i \text{ is the destination port of commodity } k \\ 0 & \text{otherwise.} \end{cases} \tag{15.1}$$

There is a limited fleet of container vessels, but not all vessels need to be used. The deployment of a vessel $v \in V$ has an associated *charter cost* c^v. Additionally, there are other costs related with the resulting network, such as the *sailing cost* c_{ij}^v associated with each vessel and each arc, which is given as a combination of the port call cost and the fuel consumption for the corresponding arc. When containers of commodity $k \in K$ are transferred from one vessel to another in port $i \in N$, there is a *transshipment cost* c_{ik}^{T} for each container. Furthermore, there is an associated sailing time t_{ij}^v for each container vessel v sailing between ports i and j, which is given as a combination of its design speed and the corresponding distance between the ports. Moreover, each port $i \in N$ has an associated berthing time b_i.

One of the main traits of the liner shipping industry is the regular operation of services under a pre-established schedule. Sometimes it is possible to define the set of candidate services in advance. In these cases, let S be the set of feasible services in the model. Notice that S can be exponentially large. Each service $s \in S$ has an associated operational cost c_s. As the set of services is defined beforehand, the operational cost is given as a combination of the sailing cost of the arcs on the service route and the corresponding port-call costs. Moreover, it is required that all services should have *weekly operations*, meaning that if a round trip takes 8 weeks to complete, then eight similar vessels need to be deployed to the service in order to ensure that each port is visited once a week. Therefore, the required number of vessels from vessel class v to maintain the weekly frequency is defined as m_v^s. The structure of the service can be divided into several types according to the number of times a port is visited during the service. A *simple service* or a *circular service* visits each port in the service exactly once. However, a service is often allowed to be non-simple, meaning that a port can be visited several times, as this may improve transit times. Nodes (or ports in the sequence) that are visited several times in the service are denoted *butterfly nodes* and a service containing a single butterfly node defines a *butterfly service*. Another common service type is the *pendulum service*, in which each port is visited twice, once in each direction. Examples of the different type of services for some European ports are illustrated in Figs. 15.3, 15.4 and 15.5. Moreover, in some cases, the structure of the service can naturally be defined by the geographical distribution of the ports. For instance, if the ports are located along the coastline, it can be convenient to define services following an outbound-inbound principle. Such services are similar to pendulum services; however, they

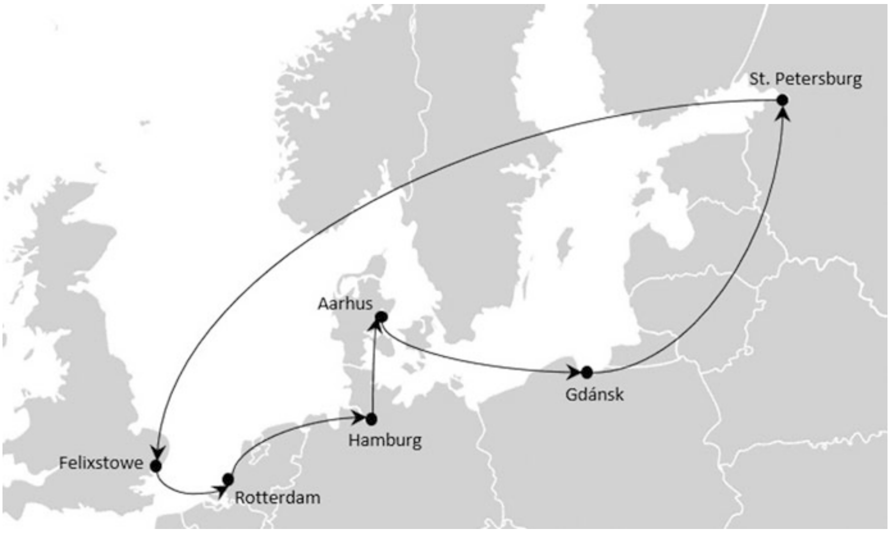


Fig. 15.3 Example of a simple service, where each port is visited exactly once

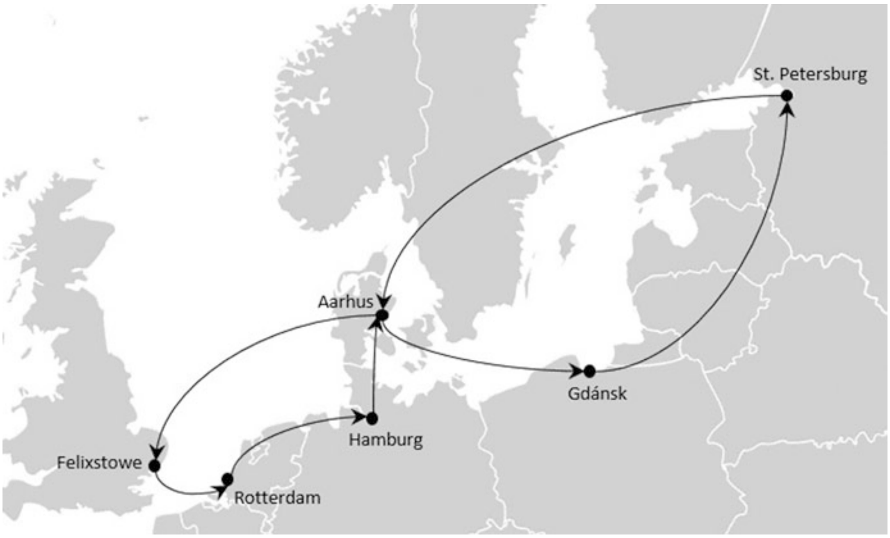


Fig. 15.4 Example of a butterfly service, where Aarhus is the butterfly node

may be asymmetric, as some ports can be omitted in each direction, as illustrated in Fig. 15.6.

When designing a shipping network, different variants of the LSNDP can be considered, varying mainly in the following four respects:

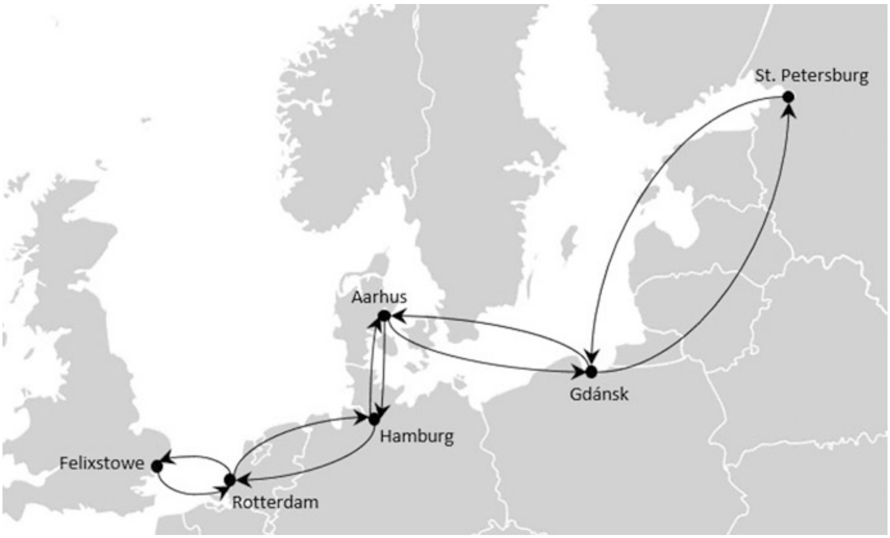


Fig. 15.5 Example of a pendulum service, where each port can be visited both on the head-haul and back-haul trip

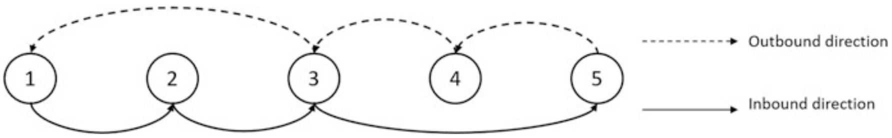


Fig. 15.6 Example of an asymmetric service with five ports with the outbound-inbound principle

- *Transit time constraints.* The demands may be subject to transit times and hence have an associated time limit that must be respected. If the transit time is not respected, perishable goods may become spoiled.
- *Transshipment costs.* The costs of transshipments are a significant part of the operational costs, so it is generally important to represent these costs in the model.
- *Rejected demands.* Although the standard formulation of LSNDP states that all demands must be flowed through the network, many models allow rejection of demands by imposing a penalty.
- *Speed optimisation.* There are three main approaches to model speed optimisation: Models which have constant speed for all services, models which choose a speed for each service, and models which choose a speed on each individual leg in each service. As the fuel consumption depends non-linearly on the speed, it is common to choose between a number of discrete speed alternatives, each with a corresponding cost.

Most models for LSNDP design the network without a specific schedule. Hence the service for each vessel is defined, but not the exact day of arrival/departure. This is typically done in a later step, where port availabilities are negotiated and transshipment times at ports are adjusted.

2.3 RoRo Network Design

RoRo shipping is an important segment within liner shipping. The RoRo ships are vessels designed to carry wheeled cargo, such as cars, trucks, semi-trailer trucks, and railroad cars, that can be driven on and off the ship on their own wheels. In addition, RoRo ships may carry complex cargo that is placed on trolleys and rolled on and off the ships, such as boats, helicopters, and heavy plant equipment. RoRo shipping is often the only viable method of ocean freight transportation for these oversized vehicles, as they may not fit in standard containers. There exist various types of RoRo ships, such as ferries, cruise ferries, cargo ships, and barges. In this subsection, we consider the RoRo ships used for transporting cars, trucks and complex general cargo across oceans known as Pure Car Carriers (PCC), Pure Truck & Car Carriers (PCTC) and general RoRo ships, respectively. A typical PCTC has a carrying capacity in the range of 5500 to 8000 RT43. Here, RT43 is a capacity measure in the RoRo business and corresponds to the size of a 1966 Toyota Corona. In 2016, the world fleet of RoRo ships consists of around 5000 ships with a total capacity of more than 24 million deadweight tons (ISL 2016).

The trades to be serviced in RoRo shipping are usually designed based on a large number of contracts for transportation of cargo between the different port pairs along a trade route. Hence, trade routes are defined as transportation arrangements from one geographical region to another, where the world is divided into a number of geographical regions. Each trade route has a number of loading ports in one region and a number of discharging ports in the other. In Fig. 15.7, two trade routes are illustrated by solid lines and the ports are shown as filled circles. These trade routes are similar to the services in container network design. As seen in Fig. 15.7, the trade routes are not traditionally circular. After a ship has sailed one voyage on a trade route it often needs to reposition to start on the next one due to trade imbalances. This repositioning means ballast sailing, i.e., sailing without cargo, which of course should be reduced as much as possible. The ballast sailing between the two trade routes in Fig. 15.7 is illustrated by a dashed line. Differences in contractual requirements and variety in the types of cargo transported on the various trade routes may restrict which vessels can be assigned to a particular trade route, regarding both capacity and vessel type.

Each trade route is sailed regularly, for example weekly, fortnightly, 3 times per months, depending on demand and contractual obligations. Each sailing on a trade

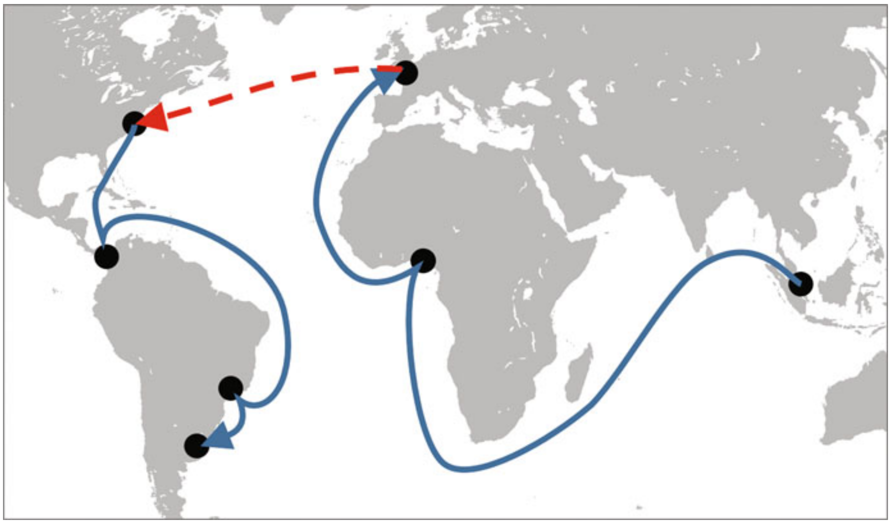


Fig. 15.7 Illustration of two trade routes sailed in sequence, Oceania to Europe and North to South America, with associated ballast sailing from Europe to North America. World map by San Jose under a CC-BY-SA-3.0 license

route is called a voyage, and normally there is a time window to start sailing a voyage. Due to contractual obligations, these voyages are mandatory and must be covered either by a ship in the RoRo shipping company's own fleet or by a chartered ship. A RoRo shipping company owns and operates a heterogeneous fleet of ships having different cargo capacities, sailing speed ranges, and bunker consumption profiles, and serves a given set of trade routes.

The operations within RoRo shipping deviates from container shipping in several ways as well as in the cargo and ships. In container shipping, each ship is normally assigned to a single route, while in RoRo shipping a ship may sail several trade routes during a planning horizon. Ship types, instead of individual ships, are often considered in container shipping, while in RoRo shipping a route for each ship is determined. This also means that in RoRo shipping each trade route may be serviced by different ship types. Furthermore, there is a great variation in when to start each voyage, as well as when and how often to visit each port along the trade. Therefore, existing studies within fleet deployment in RoRo shipping have used time windows for when each voyage along each trade should start. This flexibility is in contrast to container shipping, as each service is usually served on a strict weekly basis, and each voyage along the trade visits all ports in the same order. Finally, transshipment rarely exist in RoRo shipping in contrast to container shipping. This makes the network design easier.

Table 15.1 The seven test instances included in LINER-LIB with indication of the number of ports ($|N|$), the number of origin-destination pairs ($|K|$), the number of vessel classes ($|V|$), the minimum (min v) and maximum number of vessels (max v) in each class

Instance	Category	$\|N\|$	$\|K\|$	$\|V\|$	min v	max v
Baltic	Single-hub	12	22	2	5	7
WestAfrica	Single-hub	19	38	2	33	51
Mediterranean	Multi-hub	39	369	3	15	25
Pacific	Trade-Lane	45	722	4	81	119
AsiaEurope	Trade-Lane	111	4000	6	140	212
WorldSmall	Multi-hub	47	1764	6	209	317
WorldLarge	Multi-hub	197	9630	6	401	601

2.4 The LINER-LIB Test Instances

In order to make it easier to compare algorithms for LoLo liner shipping network design, Brouer et al. (2014a) introduced the LINER-LIB benchmark suite. The test instances in LINER-LIB are based on real-life data from leading shipping companies along with several other industry and public stakeholders. The benchmark suite contains data on ports including port call cost, cargo handling cost and draft restrictions, distances between ports considering draft and canal traversal, vessel related data for capacity, cost, speed interval and bunker consumption, and finally a commodity set with quantities, revenue, and maximal transit time. The commodity data is intended to reflect the differentiated revenue associated with the current imbalance of world trade.

The LINER-LIB benchmark suite consists of seven instances described in Brouer et al. (2014a) and is available at http://www.linerlib.org. The instances range from smaller networks suitable for being solved by exact solution methods to large scale instances spanning the globe. Table 15.1 gives an overview of these instances.

Each of the instances can be used in a low, base, and high capacity case depending on the fleet of the instance. For the low capacity case, the fleet quantity and the weekly vessel costs are adjusted to fewer vessels with a higher vessel cost. For the high capacity case the adjustments are reversed.

Currently, most papers only report results for the base capacity case. Furthermore, most often only the six first instances are considered, with Krogsgaard et al. (2018) being the only paper to report results for the *WorldLarge* instance.

3 Overview of Models and Algorithms

Designing a liner shipping network is a difficult task, embracing several decisions: Not only do we need to construct the individual services, but we should also deploy vessels of the right size to each service and ensure that there is sufficient capacity in the network to transport all containers from their origin to their destination. Designing the individual services is an NP-hard problem. Furthermore, routing the containers through a given network subject to time constraints for each container, can be recognised as a time-constrained multicommodity flow problem, which is also NP-hard.

The problem is further complicated by the fact that ports are often visited several times in the same service. This allows containers to quickly be transshipped to other services, and it frees up capacity. However, formulating the problem with multiple visits to a port as a MIP model becomes more difficult.

Finally, one should notice that transshipment costs represent the majority of the cost of routing the containers through the network according to Psaraftis and Kontovas (2015). It is therefore important to carefully model which containers might be transshipped and at which costs. This adds further complexity to the problem, and makes a graph formulation huge and difficult to solve.

First, we will focus on how the problem can be modeled. Generally speaking, *models* for liner shipping network design can roughly be divided into the following two groups:

- *Service selection models*. The idea behind these models is to use a large set of promising candidate services and then select a subset of them to create a network. The set of promising candidate services can both incorporate services designed by experienced planners as well as services internally generated by an algorithm. Many shipping companies and customers do not want the network to be completely restructured, in which case proposing small variations to each service may be a sensible method.
- *Arc formulation models*. These unified MIP models design services and flow containers through the resulting network. In order to handle this task, two sets of variables are needed: Binary variables to select edges in a service, and integer variables to denote the flow on each edge. If multiple visits to a node are allowed (butterfly nodes) then an additional index is needed to indicate the visit number at each node.

Many of the MIP models can in principle solve the LSNDP to optimality. However, due to the intrinsic complexity, only small instances can be solved to proven optimality within a reasonable time. The service selection based models more easily solve the problem to optimality given that only the proposed candidate services are valid. In practice, however, there may be an exponential number of valid services, and we cannot expect to get all services as input. Hence, the found solution will often be sub-optimal.

Large real world problems, in most cases, cannot be solved directly as MIP models. Therefore, it is required to use specialised algorithms to design the liner shipping network. Within this category, one of the most commonly used algorithms are the *two-stages algorithms*, which benefit from the decomposition of the original problem into two tightly related problems: the vessel service network design and the container flow problem. These algorithms can be roughly divided into the following two groups:

- *Service-first and flow-second algorithms*. As the name suggests, these algorithms model and solve the problem in two steps: Designing the services, and flowing containers through the resulting network. Frequently, these algorithms contain a feed-back mechanism, where output from the second-stage flow model is used as input to improve the services in the first stage.
- *Backbone flow algorithms*. It can be difficult to design the individual services without knowing how the containers will flow through the network. Hence, another approach is to reverse the order of the sub-problems in the two-stage algorithms, and start by finding an initial flow (a so-called backbone network) where cargo is flowed through a complete network with all connections between ports available. The connections are priced such that they are expensive at low loads and cheap at high loads, in order to make the cargo gather at few connections. The initial flow can be seen as an accomplishment of the *physical internet* (Montreuil 2011) where point-to-point transport has been replaced by multi-segment intermodal transport.

In liner shipping network design problems, each stage corresponds to a single layer of decision: service selection decisions for the network design problem and continuous decisions for the container flow problem. It is important to notice that the two-stage algorithms are both heuristics, since they first solve one stage, and then optimise the second stage with the first-stage decisions fixed. However, this decomposition by stages make the problem more tractable for larger instances.

Most of the MIP models solve the network design problem by dealing simultaneously with the two layers of decisions. On the other hand, the two-stage algorithms, and most of the heuristic approaches, separate these layers when designing the shipping network. The way in which the two layers interact is what defines the core of the algorithm. For example, when container flow decisions are postponed to the second-stage, the service selection decisions of the first-stage can be made based on estimates for serving the ports. At each iteration of the algorithm, the estimates can be updated based on the current configuration of the resulting network after flowing the containers. In other cases, such as for models or algorithms based on Column Generation, new services can be iteratively constructed using the information obtained from the dual variables of a restricted problem.

4 Models for the LSNDP

In this section we present graph-based models to define the LSNDP. Different formulations are briefly introduced to model the network design problem in liner shipping and a summary of the main mathematical formulations proposed in the literature is presented under different assumptions.

4.1 Service Selection Formulations

This section introduces service flow formulations for the LSNDP, where the set of all feasible services is defined in advance for the model. This reduces the network design to the selection of feasible services.

Let us begin with introducing a basic service formulation. We use the terminology presented in Sect. 2.2 with the addition of the following definitions: Let $G = (N, A)$ be a directed graph. Define for each service s and for each arc (i, j), the associated capacity u^s_{ij}, which is given by the maximum cargo capacity of the corresponding vessel class assigned to the service. Finally, let x^{ks}_{ij} be a continuous variable denoting the amount of commodity k transported in service s on arc (i, j), and y_s a binary variable for the selection of service s in the network. Now, the service formulation of the LSNDP can be expressed as:

$$\min \sum_{s\in S} c_s y_s + \sum_{k\in K}\sum_{(i,j)\in A} c^k_{ij} \sum_{s\in S} x^{ks}_{ij} \tag{15.2a}$$

$$\text{s.t.} \sum_{s\in S}\sum_{j:(i,j)\in A} x^{ks}_{ij} - \sum_{s\in S}\sum_{j:(j,i)\in A} x^{ks}_{ji} = \xi^k_i \qquad i\in N, k\in K \tag{15.2b}$$

$$\sum_{k\in K} x^{ks}_{ij} \le u^s_{ij} y_s \qquad s\in S, (i,j)\in A \tag{15.2c}$$

$$\sum_{s\in S} m^s_v y_s \le m_v \qquad v\in V \tag{15.2d}$$

$$x^{ks}_{ij} \ge 0 \qquad (i,j)\in A, k\in K, s\in S \tag{15.2e}$$

$$y_s \in \{0,1\} \qquad s\in S. \tag{15.2f}$$

The objective function (15.2a) minimises the total operational cost of the network. The first term accounts for the fixed cost of the selected services, whereas the second term constitutes the sailing cost of shipping the demand. Constraints (15.2b) are the flow conservation constraints, and the flow of commodities on the arcs has to respect the capacity of the vessel deployed in the selected service s as formulated in constraints (15.2c). For each vessel class $v \in V$, constraints (15.2d) ensure that the number of deployed vessels on the selected services using vessel class v does not exceed the maximum availability m_v. Finally, the domain of the variables is defined in constraints (15.2e) and (15.2f).

For a better utilisation of the capacity of the deployed vessels, the model can also allow the rejection of cargo by incurring a penalty. Hence, extra continuous variables can be defined to account for the demand that is rejected by the liner company.

As stated, the model requires all demand to be served, but cargo rejection can be included by adding additional variables. Finally, to account for transshipment costs, additional variables f^{ks}_i could be added, denoting the transshipment of commodity k at port i from service s, with a corresponding term in the objective function. The following constraints (15.3) could be used to enforce the sought behaviour, though only if we consider simple services.

$$f_i^{ks} \geq \sum_{\substack{j \in N \\ j \neq i}} x_{ji}^{ks} - x_{ij}^{ks}, \qquad i \in N \setminus \{o_k, d_k\}, k \in K, s \in S. \tag{15.3}$$

This formulation allows designing networks considering only a subset of promising candidate services. It can easily be seen that, as the size of the problem increases, the number of feasible services grows exponentially, making the model intractable to solve. One possible approach to solve this problem for large instances is to apply a Column Generation algorithm.

4.1.1 A Sub-path Service Formulation with Limited Transshipments

In the previous model, the flow of each commodity, on each sailing arc, is modelled explicitly. This leads to a compact formulation, but it turns out to be difficult to add transshipments together with complex service structures. Another approach, which allows limiting the number of transshipments for each commodity, while still permitting complex structures, is to use *sub-paths*. A sub-path is defined as the section of a service travelled by a group of containers. If this section spans the arcs from port i to port j on service s, the sub-path is denoted $\langle i, j, s \rangle$. The set H_s denotes the full set of sub-paths for service s, i.e., the set contains one sub-path $\langle i, j, s \rangle$ for each combination of ports i and j included in service s, and u_s denotes the capacity of service s.

Now, these sub-paths are used to introduce an augmented multicommodity flow network. The augmented network contains one node for each port and one link for each sub-path of each service. The sub-path structure also extends to more complex services, e.g., butterfly services. Let A_{ij}^s denote the set of sailing arcs of service s included in sub-path $\langle i, j, s \rangle$. The cost of routing one container of commodity k on sub-path $\langle i, j, s \rangle$ is denoted c_{ijs}^k. Finally, r_k denotes the maximum allowed number of sub-paths on which commodity k can travel. By limiting the maximum number of sub-paths allowed for a commodity, we also implicitly limit the number of transshipments.

We now present a multicommodity model based on flows along sub-paths in the augmented network. The binary variable y_s is equal to 1 if service $s \in S$ is selected, and 0 otherwise. The flow of commodity k using sub-path $\langle i, j, s \rangle$ as the h^{th} stage is defined by the variable x_{ijs}^{hk} for $s \in S$, $\langle i, j, s \rangle \in A_s$, and $h = 1, 2, \ldots, r_k$. Finally, z_k is equal to the unmet demand (number of containers) for commodity $k \in K$.

The sub-path service formulation can then be written as:

$$\min \sum_{s \in S} c_s y_s + \sum_{k \in K} \sum_{s \in S} \sum_{\langle i,j,s \rangle \in A_s} \sum_{h=1}^{r_k} c_{ijs}^k x_{ijs}^{hk} + \sum_{k \in K} c_k^R z_k \tag{15.4a}$$

$$\text{s.t.} \sum_{s\in S}\sum_{\langle o_k,j,s\rangle\in H_s} x^{1k}_{o_k js} + z_k = q_k \qquad \forall k\in K, \tag{15.4b}$$

$$\sum_{h=1}^{r_k}\sum_{s\in S}\sum_{\langle j,d_k,s\rangle\in H_s} x^{hk}_{jd_k s} + z_k = q_k \qquad \forall k\in K, \tag{15.4c}$$

$$\sum_{s\in S}\sum_{i:\langle i,j,s\rangle\in H_s} x^{hk}_{ijs} - \sum_{s\in S}\sum_{l:\langle j,l,s\rangle\in H_s} x^{h+1,k}_{jls} = 0 \qquad \forall k\in K, j\in N\setminus\{o_k,d_k\}, \; h=1,\dots,r_k-1, \tag{15.4d}$$

$$\sum_{k\in K}\sum_{h=1}^{r_k}\sum_{\langle i,j,s\rangle\in H_s: a\in A^s_{ij}} x^{hk}_{ijs} \le u_a y_s \qquad \forall s\in S, a\in A_s \tag{15.4e}$$

$$\sum_{s\in S} m^s_v y_s \le m_v \qquad \forall v\in V, \tag{15.4f}$$

$$x^{hk}_{ijs} \ge 0 \qquad \forall k\in K, s\in S, \langle i,j,s\in H_s\rangle \; h=1,\dots,r_k, \tag{15.4g}$$

$$z_k \ge 0 \qquad \forall k\in K, \tag{15.4h}$$

$$y_s \in \{0,1\} \qquad \forall s\in S. \tag{15.4i}$$

The objective function (15.4a) minimises the total cost comprised of fixed costs for the selected services, the cost of transporting commodities along each sub-path, and finally the penalties incurred for rejected demand. By including penalties, the problem is formulated as a cost minimisation problem as opposed to a profit maximisation problem where c^k_R would instead represent the revenue for transporting one unit of commodity k.

Constraints (15.4b) and (15.4c) ensure that the flow of each commodity k is assigned to sub-paths incident to the corresponding origin port o_k and the destination port d_k. They also ensure that the flow out of the origin port in combination with the unmet demand for commodity k adds up to the total demand for commodity k. Constraints (15.4d) are flow-balancing constraints for intermediate ports. Together with constraints (15.4b) and (15.4c) these constraints ensure that for each commodity k, the demand, minus any unmet demand, will arrive at the destination port using at most r_k sub-paths, i.e., fulfilling the constraint on a maximum number of transshipments.

Constraints (15.4e) impose capacity constraints on the sailing arcs and ensure that only sub-paths from the selected services are used. Constraints (15.4f) ensure that no more than the available vessels are used. Finally, constraints (15.4g)–(15.4i) impose non-negativity and binary restrictions on the respective decision variables.

The model formulation is flexible enough to allow incorporation of several practical container routing issues such as cabotage rules, regional policies and embargoes. The incorporation of many of these constraints can be handled during preprocessing simply by removing sub-paths that are no longer permitted. Balakrishnan and Karsten (2017) show that the problem is NP-hard.

4.2 Arc Formulations

The main problem with a service-based formulation is that generating all services S is prohibitive, due to the high number of combinatorial possibilities. Therefore, an alternative compact formulation is introduced in this section, which is based on an arc formulation. The set of predefined services S is no longer considered in the model, but the services are designed as the problem is solved.

We first present a basic mathematical model based on an arc formulation. For this we again use the notation presented in Sect. 2.2 with small extensions. Let $G = (N, A)$ be a directed graph. Moreover, let S^v be an index set for the services for vessel class v, indexed by s. Let x_{ij}^{ks} be a continuous variable denoting the flow of commodity k on arc (i, j) by service s, and y_{ij}^{s} a binary variable for the selection of arc (i, j) in service s operated by vessel class v. The binary variable γ_i^s is equal to 1 if port i is the hub port in service S. Moreover, we define τ_i^s as a continuous variable for the time in service s of vessel class v departing from port i, and w_s as an integer variable indicating the number of vessels from class v needed to maintain the weekly frequency in service s. Then, the arc formulation of the LSNDP can be expressed as follows:

$$\min \sum_{v\in V}\sum_{s\in S^v} c^v w_s + \sum_{v\in V}\sum_{s\in S^v}\sum_{(i,j)\in A} c_{ij}^v y_{ij}^s + \sum_{k\in K}\sum_{(i,j)\in A} c_{ij}^k \sum_{v\in V}\sum_{s\in S^v} x_{ij}^{ks} \tag{15.5a}$$

$$\text{s.t.} \sum_{v\in V}\sum_{s\in S^v}\sum_{j:(i,j)\in A} x_{ij}^{ks} - \sum_{v\in V}\sum_{s\in S^v}\sum_{j:(j,i)\in A} x_{ji}^{ks} = \xi_i^k \qquad i \in N, k \in K \tag{15.5b}$$

$$\sum_{j:(i,j)\in A} y_{ij}^s - \sum_{j:(j,i)\in A} y_{ji}^s = 0 \qquad i \in N, v \in V, s \in S^v \tag{15.5c}$$

$$\sum_{i\in N} \gamma_i^s = 1 \qquad v \in V, s \in S^v \tag{15.5d}$$

$$\sum_{k\in K} x_{ij}^{ks} \le u_v y_{ij}^s \qquad (i,j) \in A, v \in V, s \in S^v \tag{15.5e}$$

$$\tau_j^s \ge (\tau_i^s + t_{ij}^v + b_j)(1 - \gamma_j^s) y_{ij}^s \qquad i, j \in N, v \in V, s \in S^v \tag{15.5f}$$

$$\sum_{(i,j)\in A} (t_{ij}^s + b_j) y_{ij}^s \le 24 \cdot 7 \cdot w_s \qquad v \in V, s \in S^v \tag{15.5g}$$

$$\sum_{s\in S^v} w_s \le m_v \qquad v \in V \tag{15.5h}$$

$$x_{ij}^{ks} \ge 0 \qquad (i,j) \in A, k \in K, \; v \in V, s \in S^v \tag{15.5i}$$

$$y_{ij}^s \in \{0,1\} \qquad (i,j) \in A, v \in V, s \in S^v \tag{15.5j}$$

$$w_s \in \mathbb{Z}^+ \qquad v \in V, s \in S^v \tag{15.5k}$$

$$\tau_i^s \ge 0 \qquad i \in N, v \in V, s \in S^v. \tag{15.5l}$$

The objective function (15.5a) minimises the cost of deploying the vessels and designing the services, and the cost of transporting the demand quantities through the network. The flow conservation constraints for the cargo variables are given by constraints (15.5b), whereas the flow conservation constraints for the routing variables are given by constraints (15.5c). Constraints (15.5d) ensure that there is only a single hub port for each service. The flow of cargo on an edge (i, j) cannot exceed the capacity u_v of a vessel class, as expressed in (15.5e). If the service does not use a given edge in the graph, i.e., $y_{ij}^s = 0$, then the capacity is zero. The time schedule constraints for the routing variables are given by the time variables in constraints (15.5f). Note that it is necessary to linearise these constraints, as they are non-linear. Moreover, these constraints also ensure the elimination of subtours when designing the liner network, and prevents non-simple services. The weekly frequency of the services and the deployment of the fleet are expressed by constraints (15.5g). The availability of the fleet is limited by constraints (15.5h). Finally, the domain of the variables is defined by constraints (15.5i)–(15.5l).

Furthermore, the arc formulation may also include the rejection of cargo by defining continuous variables that account for the overall demand that is not flowed. Moreover, as this formulation only allows simple services, constraints (15.3) can also be included to account for transshipments.

This model is a simple representation of the arc formulation for the LSNDP, but it can be extended to incorporate the various constraints and considerations encountered in liner shipping. One of the main drawbacks with the formulation is that only simple services can be modelled. Next, we present an extended MIP model based on an arc-flow formulation which incorporates the network design and fleet assignment and accounts correctly for the transshipment cost in intermediate ports. Moreover, the model can handle the inclusion of butterfly services, where it is allowed to visit a single port at most twice during the service, and the capacity of the services dependent on the time horizon and the service length.

Let $G = (N, A)$ be a directed graph. Define the set V as the set of vessels, instead of the set of vessel classes. We consider each vessel v to belong to its own vessel class. Let t_{max} be the length of the time horizon. The design of the network is modelled with the binary variables y_{ij}^v for the utilisation of an arc (i, j) in the service for vessel v. Similarly, as proposed by Miller et al. (1960), positive integer variables e_{ij}^{v} are defined for enumerating the arcs used in the vessel service and avoid subtours in services. As the model allows the definition of butterfly services, the binary variables γ_i^v and z_{ij}^v are required to, respectively, identify the unique center-point, i.e., the hub port in the vessel service, and to allow the possibility of identifying the first and last arc visiting the hub port. These variables are used for modelling the transshipment of cargo in hub ports. The routing of containers through the network is modelled with continuous variables x_{ij}^{kv}, and extra continuous variables are defined to count the amount of the transshipped containers in intermediate ports within the same service. Let the continuous variables f_j^{kv} define the amount of commodity k transshipped by vessel v at port j, while the continuous variables f_{jih}^{kv} denote the amount of commodity k, arriving at port i through arc (j, i), in vessel v, and not leaving in arc (i, h). Furthermore, the model also considers the fleet deployment. The problem is defined for a heterogeneous fleet and the service capacities depend on the deployed vessels as well as the frequency at which they sail. The deployment of a vessel is controlled by the binary variable λ^v, whereas the continuous variables τ_v limit the service length of the vessels. Then, the arc-flow model can be defined as:

$$\min \sum_{k\in K}\sum_{(i,j)\in A} c_{ij}^{k} \sum_{v\in V} x_{ij}^{kv} + \sum_{k\in k}\sum_{i\in N} c_{ik}^{\mathrm{T}} \sum_{v\in V} f_i^{kv} + \sum_{v\in V} c^v \lambda^v \tag{15.6a}$$

$$\text{s.t.} \sum_{v\in V}\sum_{j:(i,j)\in A} x_{ij}^{kv} - \sum_{v\in V}\sum_{j:(j,i)\in A} x_{ji}^{kv} = \xi_i^k \qquad i \in N, k \in K \tag{15.6b}$$

$$f_i^{kv} \geq \sum_{j:(i,j)\in A} x_{ij}^{k} - \sum_{j:(j,i)\in A} x_{ji}^{k} \qquad k \in K, i \in N, v \in V \tag{15.6c}$$

$$f_i^{kv} \geq \sum_{j,h\in N}\sum_{v\in V} f_{jih}^{kv} - M_1(1-\gamma_i^v) \qquad k \in K, i \in N, v \in V \tag{15.6d}$$

$$f_{jih} \geq x_{ji}^{kv} - x_{ih}^{kv} - M_2(2 - y_{ji}^v - y_{ih}^v + z_{ji}^v + z_{ih}^v) \quad k \in K, j, i, h \in N, v \in V \tag{15.6e}$$

$$f_{jih} \geq x_{ji}^{kv} - x_{ih}^{kv} - M_3(4 - z_{ji}^v - z_{ih}^v - y_{ji}^v - y_{ih}^v) \quad k \in K, j, i, h \in N, v \in V \tag{15.6f}$$

$$\sum_{i\in N} \gamma_i^v = 1 \qquad v \in V \tag{15.6g}$$

$$\sum_{(i,j)\in A} z_{ij}^v = 2 \qquad v \in V \tag{15.6h}$$

$$\gamma_i^v - \sum_{j:(i,j)\in A} z_{ij}^v \le 0 \qquad i \in N, v \in V \qquad (15.6i)$$

$$\gamma_i^v - \sum_{j:(j,i)\in A} z_{ji}^v \le 0 \qquad i \in N, v \in V \qquad (15.6j)$$

$$\sum_{j:(i,j)\in A} y_{ij}^v - \sum_{j:(j,i)\in A} y_{ji}^v = 0 \qquad i \in N, v \in V \qquad (15.6k)$$

$$\sum_{j:(i,j)\in A} y_{ij}^v - \gamma_i^v \le 1 \qquad i \in N, v \in V \qquad (15.6l)$$

$$e_{ji}^v - e_{ih}^v + M_4(y_{ih}^v + y_{ji}^v - 2 - z_{ji}^v - z_{ih}^v) \le -1 \qquad i, j, h \in N, v \in V \qquad (15.6m)$$

$$y_{ij}^v - \lambda^v \le 0 \qquad (i, j) \in A, v \in V \qquad (15.6n)$$

$$\tau_v \le t_{max} \qquad v \in V \qquad (15.6o)$$

$$\tau_v = \sum_{(i,j)\in A} y_{ij}^v (t_{ij}^v + b_j) \qquad v \in V \qquad (15.6p)$$

$$\frac{t_{max}}{\tau_v} u_v y_{ij}^v \ge \sum_{k\in K} x_{ij}^{kv} \qquad (i, j) \in A, v \in V \qquad (15.6q)$$

$$z_{ij}^v, y_{ij}^v \in \{0, 1\} \qquad (i, j) \in A, v \in V \qquad (15.6r)$$

$$f_{jih}^{kv} \ge 0 \qquad k \in K, j, i, h \in N, v \in V \qquad (15.6s)$$

$$f_j^{kv} \ge 0 \qquad k \in K, j \in N, v \in V \qquad (15.6t)$$

$$e_{ij}^v \in \mathbb{Z}^+ \qquad i, j \in N, v \in V \qquad (15.6u)$$

$$x_{ij}^{kv} \ge 0 \qquad (i, j) \in A, k \in K, v \in V \qquad (15.6v)$$

$$\gamma_i^v \in \{0, 1\} \qquad i \in N, v \in V \qquad (15.6w)$$

$$\lambda^v \in \{0, 1\} \qquad v \in V \qquad (15.6x)$$

$$\tau_v \ge 0 \qquad v \in V. \qquad (15.6y)$$

The objective function (15.6a) minimises the total cost of transporting the cargo through the network, the transshipment costs of the demand in hub ports, and the associated cost for deploying the vessels. Furthermore, the flow conservation constraints (15.6b) ensure that all demand is satisfied. Constraints (15.6c) account for the amount of containers transshipped in intermediate ports; however, if the corresponding service is non-simple, the model requires constraints (15.6d)–(15.6f) for updating the commodities transshipped by the same vessel in butterfly services. Constraints (15.6g)–(15.6j) are used to handle butterfly services. Constraints (15.6g) identify the unique butterfly node for the vessel service and constraints (15.6h)–(15.6j) find the adjacent arcs corresponding to the first or last visit to the butterfly node of the vessel service. Moreover, constraints (15.6k) are the flow conservation constraints for the network design of the vessel service, and constraints (15.6l) control the number of times a vessel visits the hub port in a service. In addition, constraints (15.6m) use the previous information for correctly enumerating the order in which the vessel traverses the arcs in the vessel service. Additionally, the fleet deployment is controlled by constraints (15.6n), and the corresponding service length is computed in constraints (15.6o) and (15.6p). It can also be seen that the service length is included in the capacity constraints (15.6q), as was first introduced in Agarwal and Ergun (2008). However, the model does not require weekly departures for all ports. The inclusion of time for the service in the calculation of the capacity results in a non-linear model. Therefore, it is necessary to linearise the corresponding constraints (15.6q) together with (15.6o)–(15.6p) in order to obtain a MIP formulation. Finally, constraints (15.6r)–(15.6y) define the domain of the decision variables.

The high number of details in the model allows the representation of a fairly realistic problem, making it possible to design efficient services reducing the overall operational costs. Nonetheless, it can easily be observed that the compact model is computationally hard to solve. The model presents several "big-M" constraints, which produce a very weak relaxation, and Branch-and-Bound techniques provide large integrality gaps and poor bounds. One proposed method to solve this problem is Branch-and-Cut, as it has presented good results to the VRP and other transportation network design problems. The idea is to solve the relaxed problem without the transshipment constraints, (15.6d)–(15.6f), and the connectivity constraints (15.6h)–(15.6j) and (15.6m) in butterfly nodes and then, gradually add cuts to the formulation when those constraints are violated. This was used by Reinhardt and Pisinger (2012), to solve instances of up to 10 ports to proven optimality.

4.3 Considering Non-simple Services in the Formulation

The majority of models for LSNDP are defined using an arc formulation. These formulations are suitable when the structure of the services can be assumed to be simple. However, these formulations become problematic when formulating non-simple services, as it requires the inclusion of many extra variables in the model.

In this section, we present different modelling approaches that can be used when studying the design of more complex services.

4.3.1 Port-Call Formulations

The general idea of this formulation is to define services as sequences of port-calls, instead of by the arcs that connect the physical ports in the graph, as presented in Sect. 4.1. The formulation keeps track of the order in which the ports are visited during a service, which makes it possible to distinguish between multiple visits to a port by the same service. This, in turn, allows the inclusion of non-simple services, which better reflects how services are designed in practice.

The port-call formulation defines service flow variables to model how containers are transported within and between services. Therefore, the transshipment of cargo can then be modelled by the service flows. Hence, not only can this approach capture the cost of transshipment between different services, but also within the same service. Furthermore, the model can also account for the demand that is rejected by considering the cargo that is not flowed in the services.

Nevertheless, the decision variables of this formulation must be extended with an extra sequence index to identify the corresponding physical port visited at certain port-call by the service. Since the complexity of the problem increases with the addition of the extra index, it is common to impose a maximum number of port-calls within a service.

4.3.2 Layer-Networks for Complex Services Structures

One way of managing the difficulty of handling complex service structures together with transshipment restrictions, is by using a graph network with multiple nodes for each port.

The graph network is modelled with multiple layers, where each layer is a complete sub-graph, containing exactly one copy of each port. This way, each visit to a port is considered a separate node, and the maximum number of visits to each port is limited by the number of nodes representing that port. The layers are only connected between nodes representing the same port so that it is possible to switch layers at any given point. A graphical representation of the network can be seen in Fig. 15.8. The key is that, by separating the different visits to a port, complex services can be modelled, together with transshipment costs for transshipments within the same service.

However, the drawback with this modelling approach is that it is significantly more complex. In addition to using more nodes and edges, symmetries can easily arise, as a service could swap layers more or less arbitrarily without changing the represented solution. Some of this can be counteracted by symmetry breaking constraints, but to this day, only smaller instances of up to around 15 ports have been solved with modelling approach. Additionally, to keep down the complexity, it

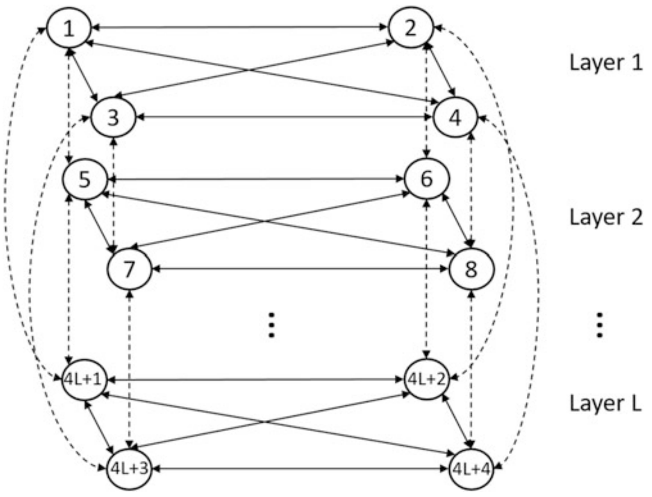


Fig. 15.8 Visual representation of the layer-network with four ports and L layers

can be beneficial to impose a limitation of two layers in the defined graph network. This is a quite realistic limitation, as a service seldom visits a port more than twice. Hence, the problem allows the creation of simple, butterfly and pendulum services.

4.3.3 Time-Space Models

The LSNDP can also be represented by a time-space graph. The key of this modelling approach is to include the temporal aspects into the graph structure. Therefore, this graph structure not only allows the representation of any type of service, but also integrates the vessel scheduling into the modelling.

Let H be the set of time-units after discretising the weekly planning horizon into uniform time steps. This discretisation is used to define the graph structure of the problem. Define $G = (\tilde{N}, A)$ as a directed graph, where $\tilde{N}$ and A are the set of nodes and arcs, respectively.

The set of nodes contains a copy of each port for each of the time steps. Hence, each node $(i, h) \in \tilde{N}$ represents the departure time of a vessel from port $i \in N$ at a specific time $h \in H$. The set of arcs A is composed by two set of arcs; the set of *sailing arcs* A^S and the set of *transshipment arcs* A^T. Each sailing arc $a \in A^S$ connects the departure of a vessel from two different port-nodes, i.e., connects two nodes (i_1, h_1) and (i_2, h_2) such as $i_1 \neq i_2$. Note that, due to the weekly frequency of the services, the times should be computed modulo 7 days, and hence, services are modelled as closed cycles in the graph. Moreover, each transshipment arc $a \in A^T$ represents the cargo transshipment between two services at the same port, i.e., the arc connects two nodes (i_1, h_1) and (i_2, h_2) such as $i_1 = i_2$. As an example, Fig. 15.9 illustrates a time-space graph for four ports with two services.

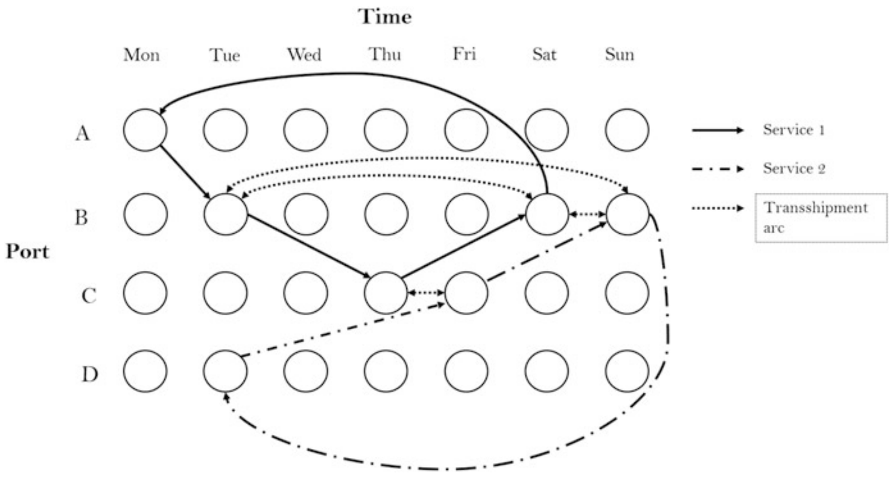


Fig. 15.9 Example of a time-space graph *G* with four ports with a planning horizon discretised by the days of the week. The graph depicts two services, where Service 1 visits port B twice during the same rotation. Transshipment arcs are represented by dashed arcs, connecting the nodes for the same port

The transshipment of cargo between services can be carried out at ports B and C. Moreover, port B is visited twice by Service 1, allowing cargo to be transshipped within the same service.

One of the main advantages with the time-space models, is that in addition to generating the routes for the services, they also generate the schedule. They also make it handy to model more complex operations, such as speed optimisation and transshipment times. Furthermore, if we disallow services to visit the same port twice at the same time slot, we get the same benefits as the layered networks in terms of handling transshipment together with complex services. This, however, again comes at the price of having significantly more nodes and edges in the graph.

An important design choice, when working with time-space graphs is the time discretisation granularity. By defining small time steps, the model can better reflect the aforementioned operations; however, the graph would require a large number of nodes, which increase the overall complexity of the problem. On the other hand, large time steps may be too restrictive, and the model may not capture the additional constraints as accurately.

5 Two-Stage Algorithms

The LSNDP consists of two tightly interrelated problems—the vessel service network design and the container flow problem. One of the most successful approaches so far for finding good solutions to the LSNDP, has been to use heuristics exploiting this two-tier structure.

In general, the two stage algorithms can be divided into two categories: *service first* and *flow first*. In *service first*, the service generation is the leading procedure, and the containers are then flowed, given the generated services. It is then common to use information from the container flow to update the services. This way a feedback loop is created, iteratively improving the services and solving the container flow.

In *flow first*, the containers are instead first flowed without any information regarding the services, and the services are subsequently created to match that flow. The key concept is to try to aggregate the container flows onto a small subset of the arcs, to facilitate the ensuing service generation.

5.1 The Container Flow Problem

Before going into the full two-stage algorithms, let us briefly discuss the container flow problem, which is the lower tier problem in the LSNDP two-tier structure. In general, for a given set of services, the container flow problem reduces to a *multicommodity flow* problem (MCFP) with fractional flows allowed.

In addition to the notation defined in Sect. 2.2, we need an additional parameter; to each arc $(i, j) \in A$ define its corresponding flow capacity, u_{ij}. The arc set A and its corresponding costs c_{ij}^k and capacities u_{ij} are defined by the vessel services, designed in the upper-tier problem. Also, let x_{ij}^k be a continuous variable denoting the flow of commodity k through arc (i, j). The MCFP can then be expressed as:

$$\min \sum_{(i,j)\in A} \sum_{k\in K} c_{ij}^k x_{ij}^k \tag{15.7a}$$

$$\text{s.t.} \sum_{j\in N:(i,j)\in A} x_{ij}^k - \sum_{j\in N:(j,i)\in A} x_{ji}^k = \xi_i^k \qquad i \in N, k \in K \tag{15.7b}$$

$$\sum_{k\in K} x_{ij}^k \leq u_{ij} \qquad (i, j) \in A \tag{15.7c}$$

$$x_{ij}^k \geq 0 \qquad (i, j) \in A, k \in K. \tag{15.7d}$$

Here, the objective, (15.7a), is to minimise the total cost. Constraints (15.7b) are the flow conservation constraints, constraints (15.7c) are the capacity constraints, and constraints (15.7d) define the domain of the variables x_{ij}^k.

When fractional flows are allowed, the MCFP is solvable in polynomial time. For larger instances it is, however, still computationally demanding. As the model generally has to be solved a multitude of times in the presented two-tier solutions to the LSNDP, efficient solution methods to the MCFP are essential.

One of the most common solution approaches is to exploit its block-angular constraints matrix and apply Dantzig-Wolfe Decomposition (Ahuja et al. 1993). The problem should first be reformulated as a path flow formulation, where the goal is to allocate the commodities to a number of flow paths from the commodities' origins to their destinations, while respecting the capacity constraints on the arcs. Let P^k be the set of all paths for commodity $k \in K$, from o_k to d_k, and let P_a^k be the set of paths for commodity k, which uses arc a. Then we define

$$P_a = \bigcup_{k \in K} P_a^k,$$

to be the set of all paths going through arc $a \in A$. For each path, p, for commodity $k \in K$, define its cost $c_p^k = \sum_{a \in A : p \in P_a^k} c_a^k$, and a corresponding decision variable f_p^k, deciding the flow through path p. The path flow formulation can then be expressed as:

$$\min \sum_{k \in K} \sum_{p \in P^k} c_p^k f_p^k \tag{15.8a}$$

$$\text{s.t.} \sum_{p \in P^k} f_p^k = \xi_{o_k}^k \qquad k \in K \tag{15.8b}$$

$$\sum_{p \in P_a} f_p^k \leq u_a \qquad a \in A \tag{15.8c}$$

$$f_p^k \geq 0 \qquad k \in K, p \in P^k. \tag{15.8d}$$

The objective function, (15.8a), minimises the cost. Constraints (15.8b) ensure that all commodities are delivered and constraints (15.8c) assert that the arc capacity cannot be exceeded. Lastly, constraints (15.8d) define the domain for the variables.

The path formulation has a very large number of variables, but generally, only a few of them are needed for the optimal solution. Using column generation, the problem can be restricted to only consider a limited amount of paths for each commodity and new paths can then be generated dynamically. In this way, the path formulation can generally be solved faster than the arc formulation, described earlier. The path formulation makes it relatively easy to implement transit time constraints as they can be handled in the pricing problem.

5.2 *Service First Methods*

While the lower-tier container flow problem is solvable in polynomial time (when no transit time constraints are imposed), the upper-tier service selection problem is NP-hard, and just calculating the objective value of a given solution, requires solving the container flow problem. This makes the service selection problem difficult to solve to optimality and instead several matheuristics have been developed to find good solutions to larger instances. A matheuristic is a method that employs heuristics together with methods from linear and integer programming. In the case of the LSNDP, the most common procedure is to use linear programming tools to solve the MCFP and then various heuristics to update the vessel services.

Typically, we keep a set of services S, each with a designated speed and vessel class, and sufficient vessels to keep the weekly frequency. When solving the container flow for those services, we can see which services are under or over utilised and change the speed and vessel classes of the services accordingly. Looking at the dual variables for the capacity constraints, such as constraints (15.7c), we get information about which arcs would need more capacity to lower the costs. This can then be used to modify or create new services. Then, based on this information, we gradually modify the services, most commonly within a tabu-search framework.

An alternative approach is to instead use linear programming for updating the services. When modifying a service, the evaluation of the resulting change in time and revenue require the full solution of the commodity flow problem. This is generally too expensive, if we would like to compare multiple modifications, and the idea is therefore to instead work with estimates of those changes. We will iteratively update the services one by one, and for each service we find a set of potential ports to either include or exclude from the service. The key is then to, for each of those ports, estimate the change in revenue for the problem and the change in time for the service, if this port would be included or excluded. Where in the services to insert new ports is decided using a greedy heuristic. Estimating the change in service time is straightforward, but for estimating the change in revenue, we need an estimate of how the cargo flow would be affected. For this we solve a shortest path problem for each affected commodity to evaluate the approximate cost of routing it through the resulting network.

With the estimates in place, we can then define an integer program to optimise the service changes. In addition to the notation from the Sect. 2.2, let τ_s be the time length of a service s, let $\Delta^{\mathrm{R}+}_{is}$ ($\Delta^{\mathrm{R}-}_{is}$) be the estimated revenue change, and $\Delta^{\mathrm{T}+}_{is}$ ($\Delta^{\mathrm{T}-}_{is}$) be the estimated duration change from including (excluding) port $i \in N$ in (from) service $s \in S$. Also, let η^+_s (η^+_s) be the maximum number of inclusions (removals) allowed and let $\bar{N}_s$ denote the set of ports which can be included. Let $\hat{m}_v$ denote the number of free vessels of class v, such that

$$\hat{m}_v = m_v - \sum_{s \in S} m^s_v.$$

Lastly, let us define the binary variables x^+_{is} and x^-_{is}, which control the inclusion and removal, respectively, of port i from service s, and the integer variables ζ_s, which denote the number of vessels to add to/subtract from service s. For each service $s \in S$, with corresponding vessel class v, we can then define the following integer program:

$$\max \quad \sum_{i \in N_s} \Delta_{is}^{\mathrm{R}+} x_{is}^{+} + \sum_{i \in \bar{N}_s} \Delta_{is}^{\mathrm{R}-} x_{is}^{-} - c_v \zeta_s \tag{15.9a}$$

$$\text{s.t.} \quad \tau_s + \sum_{i \in N^s} \Delta_{is}^{\mathrm{T}+} x_{is}^{+} + \sum_{i \in \bar{N}_s} \Delta_{is}^{\mathrm{T}-} x_{is}^{-} \leq 24 \cdot 7 \cdot (m_v^s + \zeta_s) \tag{15.9b}$$

$$\zeta_s \leq \hat{m}_v \tag{15.9c}$$

$$\sum_{i \in \bar{N}_s} x_{is}^{+} \leq \eta_s^{+} \tag{15.9d}$$

$$\sum_{i \in N_s} x_{is}^{-} \leq \eta_s^{-} \tag{15.9e}$$

$$\sum_{j \in L_i^{+}} x_{js}^{-} \leq |L_i^{+}|(1 - x_{is}^{+}) \qquad i \in \bar{N}_s \tag{15.9f}$$

$$\sum_{j \in L_i^{-}} x_{js}^{-} \leq |L_i^{-}|(1 - x_{is}^{-}) \qquad i \in N_s \tag{15.9g}$$

$$x_{is}^{+} \in \{0, 1\} \qquad i \in \bar{N}_s \tag{15.9h}$$

$$x_{is}^{-} \in \{0, 1\} \qquad i \in N_s \tag{15.9i}$$

$$\zeta_s \in \mathbb{Z}. \tag{15.9j}$$

Here, the objective (15.9a) is to maximise the increase in revenue. Constraint (15.9b) ensures that there is enough vessels assigned to keep the weekly frequency, and constraint (15.9c) says that no more than the number of free vessels can be added to the service. Constraints (15.9d) and (15.9e) set a limit on the number of insertions and removals, while (15.9f) and (15.9g) prevent certain combinations of insertions and removals. The sets L_i^{+} are defined such that if a port i is to be inserted, then no port in L_i^{+} is allowed to be removed. If instead a port i is removed, then every port in L_i^{-} must remain. If a new port call is inserted in between two ports, then neither of those are allowed to be removed, and if inserting a new port means that a new commodity is transported, then the origin and destination nodes of this commodity are not allowed to be removed. Constraints (15.9d)–(15.9g) are defined to limit the amount of changes which can be applied, as the revenue and time change estimates are made for one or a few changes and deteriorates rapidly when multiple changes are applied. Lastly, (15.9h)–(15.9j) define the domain of the variables.

The algorithm works such that each service, one by one, is updated according to the solution of the above defined mixed integer problem. Then the MCFP is solved to update the total revenue and the effect of new changes is once again estimated with the shortest path procedure. To diversify the solutions, in every tenth iteration the services with lowest utilisation are removed and new services are created using the greedy creation heuristic. A more thorough description of the algorithm can be seen in Brouer et al. (2015). They also use the algorithm to find good solution to all the LINERLIB instances, except for *World Large*.

5.3 Backbone Flow

The main idea in backbone flow algorithms is to reverse the order of two-phase algorithms by first flowing the containers, and then constructing services that cover the flow.

For the first step in the backbone flow algorithms, which flow the containers, a directed graph $G = (N, A)$ is used. There are no capacities associated with the edges, as there are no services designed at this stage. To design a backbone flow, which can later be effectively covered by a set of services, it is important to aggregate the flows onto a few arcs. This can be achieved by using for example a fixed charge cost. Another way would be to use a concave edge cost function $c(x)$ of the flow x, reflecting the economy of scale for flowing more containers. There is a large cost associated with opening an arc (i.e., deploying a vessel), while the cost per container decreases as the flow (and hence vessel size) increases. See Fig. 15.1 for an illustration of the costs.

Let x_{ij}^k denote the flow of commodity k on edge (i, j). Then the backbone flow problem becomes a non-linear MCFP as given by

$$\min \sum_{(i,j)\in A} c(\sum_{k\in K} x_{ij}^k) \tag{15.10a}$$

$$\text{s.t.} \sum_{(i,j)\in A} x_{ij}^k - \sum_{(j,i)\in A} x_{ji}^k = \xi_i^k \qquad i \in N, k \in K \tag{15.10b}$$

$$x_{ij}^k \geq 0 \qquad (i, j) \in A, k \in K. \tag{15.10c}$$

As before, the objective, (15.10a), is to minimise the total cost, and constraints (15.10b) are the flow conservation constraints. Constraints (15.10c) define the domain of the variables.

Since the model is non-linear and non-convex, many of the classic optimisation techniques are not applicable. Instead, it is commonplace to use various heuristics and approximation methods, such as, for example, dual ascent methods or tabu search. One classic method is to use successive linearisation of the cost function, where the concave cost function is estimated with a linear function. Typically, this would be done by iteratively routing some or all containers, by solving a number of shortest path problems, based on the current arc costs, and then update the costs according to the new solution. If the arc costs are updated before all containers are routed, generally, the first containers to be routed are more decisive for which arcs are used in the final solution. To circumvent this, the algorithm is run several times,

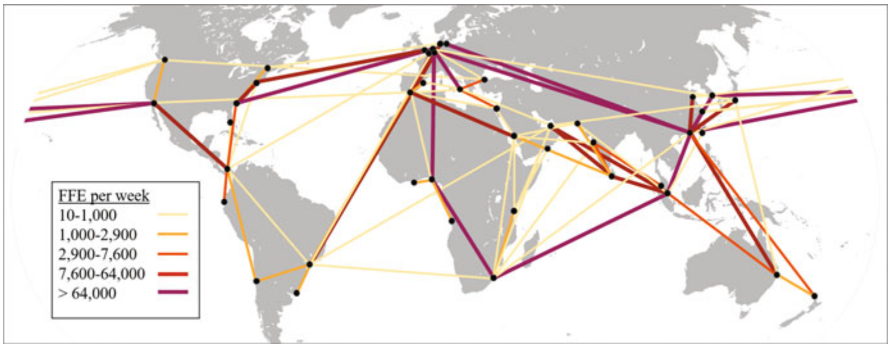


Fig. 15.10 Typical backbone flow for the *WorldSmall* instance. Data from Krogsgaard et al. (2018) and World map by San Jose under a CC-BY-SA-3.0 license

with a random order of the containers, to achieve a reasonable average picture of the backbone flow.

An example of running ten iterations for the demand matrix of the *WorldSmall* instance gives the average arc loads shown in Fig. 15.10. The figure clearly shows that only a fraction of the possible arcs is used in the solution.

5.3.1 From Backbone Flow to Network Design

Having found a backbone flow, the goal is to design services which satisfies it. This becomes a kind of arc-covering problem, but with the addition that the commodities have to be assigned to the different services for the purpose of modelling transshipments and handling capacity constraints. Alternatively, if the aim is to just create a decent initial solution, the service generation can be split in two parts: one where the routes are generated, trying to cover as much as possible of the flow, and one where the containers are assigned to the services, trying to minimise the transshipments while satisfying the capacity constraints. Due to the rather limited amount of arcs, on which containers flow, the pure arc-covering problem can be solved rather effectively using greedy construction heuristics. The idea is to create services by greedily inserting arcs one at a time, until the maximum service time is reached.

In a computational study by Krogsgaard et al. (2018), it is shown that usable solutions can be found in relatively short time. Using the *WorldSmall* instance, the authors generate 20 different sets of services by running the above algorithm where the containers are flown in random order. This can be done in about 80 s, and results in a profitable network, although the resulting network is far from optimal.

While it is unlikely that the backbone flow perfectly matches the structure of the service network, this methodology can be used to quickly generate good initial solutions. The container paths can then be updated to better fit the services, which could be the starting step in a continued two-stage algorithm.

6 Bibliographic Notes

A good introduction to the liner-shipping business along with a description of its main assets and infrastructure can be found in Brouer et al. (2014a). This paper also presents in details a complete model for the LSNDP with all side-constraints and introduces the LINER-LIB test instances. For a detailed review of the research on liner shipping optimisation problems, see the survey papers Ronen (1983, 1993), Christiansen et al. (2004), Christiansen et al. (2013), Meng et al. (2014), Tran and Haasis (2015), Brouer et al. (2016, 2017), and Lee and Song (2017).

Planning problems within RoRo shipping are far less studied in the OR literature compared to container shipping. As far as we know, there is no literature contribution considering RoRo network design. However, other strategic planning issues involving decisions regarding fleet size and mix can be found in Pantuso et al. (2015). On the tactical planning level, the fleet deployment problem consists of assigning ships in the fleet to voyages that must be performed repeatedly on given trade routes. In addition to the ship-voyage assignment, the results from fleet deployment are sailing routes for the ships in the fleet. Deployment models for RoRo shipping are presented by Fagerholt et al. (2009) and Andersson et al. (2015).

In Sect. 4.1, the service formulation for the LSNDP has been used by Álvarez (2009) and Balakrishnan and Karsten (2017). Álvarez (2009) studied the LSNDP at the tactical level, considering the joint routing and deployment of container vessels. The formulation presented in the paper is based on the set of all feasible services, which are given as a combination of vessel class, operating speed and route structure. Moreover, the idea of selecting a subset of sailing services from a pool of candidate services was originally proposed by Balakrishnan and Karsten (2017), who also presented the multicommodity model based on flows along sub-paths in the augmented network.

The arc formulation model, in Sect. 4.2, was originally presented by Reinhardt and Pisinger (2012), in which network design and fleet assignment were combined. The authors were among the first to study exact methods for the network design problem in liner shipping with transshipment operations and butterfly services. They developed a branch-and-cut algorithm and reported results for instances up to 15 ports.

In Sect. 4.3, different approaches for modelling complex-services are presented. The port-call formulation for the LSNDP is based on Plum et al. (2014). The model proposed in this paper has been used to solve the two smallest LINER-LIB instances using CPLEX. The authors reported promising solutions for these instances, but optimal solutions were not achieved due to the large number of constraints and decision variables. The layer-network for complex services structures is based on Thun et al. (2017). The problem is solved to integer optimality using a branch-and-price algorithm, reporting results for small instances of between 5 and 7 ports with up to 14 demands. Finally, the time-space graph is based on Koza et al. (2020). The authors developed a column generation matheuristic and reported very good results for LINER-LIB instances of up to 45 ports.

Section 5 is dedicated to two-stage algorithms, where column generation and Benders' decomposition have been used by Agarwal and Ergun (2008), and various matheuristics have been used by Álvarez (2009), Brouer et al. (2014b) and Karsten et al. (2017). A good introduction to the transit time constrained multicommodity flow problem can be found in Karsten et al. (2015).

The general service-first method, in Sect. 5.2, is inspired by Álvarez (2009) and the model (15.9a)–(15.9h) is based on Brouer et al. (2015). The authors report satisfactory solutions for 6 out of 7 LINER-LIB instances for the case with transit time requirements for the commodities.

In Sect. 5.3 the main idea of backbone flow was presented by Krogsgaard et al. (2018). Sun and Zheng (2016) also use a concave function to optimise the container flow. Moreover, the greedy heuristic for generating services presented in the same Sect. 5.3.1 was also presented in Krogsgaard et al. (2018). Promising computational results are reported for all LINER-LIB instances without the transit time requirements, where the best results are achieved for 4 out of the 7 instances, namely *WestAfrica*, *Pacific*, *WorldSmall* and *WorldLarge*.

7 Concluding Remarks and Future Challenges

Liner shipping is the backbone of international trade, hence it is important to develop decision support tools that can help design more cost-efficient services, and balance several objectives. This includes finding the right trade-off between speed, transportation times, number of transshipments, and operational costs.

Slow steaming together with larger vessels has proven to be an efficient tool for reducing energy consumption. However, slow steaming decreases the capacity of vessels, since they cannot transport as much cargo per time unit as before. Hence, more vessels are needed in order to maintain the same capacity, straining the environment. Bigger vessels tend to be more energy efficient per container, but the increased capacity results in longer port stays, making it necessary to speed up between the port stays. It is therefore necessary to design services such that fewer transshipments are needed, while still ensuring a good utilisation of the mega vessels.

Although liner shipping generally is one of the most energy-efficient modes of transportation per kilometer, the shipping industry emits large quantities of SO_x and NO_x. In the future, we will see container vessels operating with new, greener, propulsion types. Electric vessels may operate shorter services, while liquid natural gas (LNG) may be used for operating longer services. The new propulsion types will make it necessary to completely rethink service network design, since refueling/recharging will be more complicated, and vessels will have a more limited range of operation.

The introduction of autonomous vessels in the container shipping industry may also significantly change the way a network is designed and operated. In particular for feeder-lines we will see more smaller vessels sailing on-demand, depending on the cargo. This will turn the network design process into a dynamic routing problem. It would also be interesting to investigate whether fuel savings are achievable if autonomous vessels are sailing in convoys close to each other. If this is the case, the network design should take these convoys into account.

Nearly every vessel will be delayed in one or more ports during a round trip. Instead of just speeding up (and hence using more energy) advanced disruption management tools need to be developed that can ensure timely arrival to the end customer with the lowest possible energy consumption. Some studies along this path include Brouer et al. (2014a) but much more work needs to be done in this area.

Vessel sharing agreements are an important tool for making it possible to operate larger and more energy-efficient vessels. In a vessel sharing agreement two or more companies share the capacity of a vessel throughout the full rotation or on certain legs. Vessel sharing agreements, however, substantially increase the complexity of designing a network, since some legs and capacities are locked according to the agreement.

8 Notation Used in This Chapter

We have tried to use an uniform mathematical notation throughout the chapter, although each section needs some additional symbols. The notation has been gathered and presented in the Tables 15.2, 15.3, 15.4, 15.5, 15.6, 15.7, 15.8, 15.9, and 15.10. First, the general notation presented in the introduction to be commonly used in the mathematical models of the chapter is presented. Next, the extra notation needed to define the models of each specific section is presented.

Table 15.2 General notation from the introduction

Sets
N := set of ports
A := set of sailing arcs
K := set of commodities
V := set of vessel classes
S := set of services
Parameters
o_k := origin port for commodity $k \in K$
d_k := destination port for commodity $k \in K$
q_k := quantity of commodity $k \in K$
u_v := cargo capacity for vessel class $v \in V$
m_v := available fleet quantity of vessel class $v \in V$
$\xi_i^k := \begin{cases} q_k & \text{if port } i \in N \text{ is the origin port of demand } k \in K \\ -q_k & \text{if port } i \in N \text{ is the destination port of demand } k \in K \\ 0 & \text{otherwise.} \end{cases}$
c_{ij}^k := unit-cost for transporting a unit of commodity $k \in K$ through arc $(i, j) \in A$
c^v := cost for deployment of a vessel from vessel class $v \in V$
c_{ij}^v := sailing cost for vessel class $v \in V$ traversing arc $(i, j) \in A$
c_{ik}^{T} := cost per unit of commodity $k \in K$ transshipped in port $i \in N$
t_{ij}^v := sailing time for vessel class $v \in V$ traversing arc $(i, j) \in A$
b_i := berthing time for the port call $i \in N$
c_s := cost for operation service $s \in S$.
m_v^s := required number of vessels from vessel class $v \in V$ to maintain the weekly frequency in service $s \in S$

Table 15.3 Additional notation for Sect. 4.1

Parameters
u_{ij}^s := capacity for service $s \in S$ along arc $(i, j) \in A$
Decision variables
$x_{ij}^{ks} \in \mathbb{R}^+$:= amount of commodity $k \in K$ transported in service $s \in S$ along arc $(i, j) \in A$
$y_s \in \{0, 1\}$:= equal to 1 if service $s \in S$ is selected in the network, and 0 otherwise

Table 15.4 Additional notation for Sect. 4.1.1 (Balakrishnan and Karsten 2017)

Sets
A_s := set of sailing arcs for service $s \in S$
H_s := full set of sub-paths for service $s \in S$ (a sub-path is a part of a route in which the container travels on a single service and is denoted $\langle i, j, s \rangle$)
A^s_{ij} := set of sailing arcs of service $s \in S$ included in sub-path $\langle i, j, s \rangle$
Parameters
u_a := capacity for arc $a \in A_s$
c^R_k := penalty cost per container for rejected demand of commodity $k \in K$
c^k_{ijs} := cost of routing one container of commodity $k \in K$ on sub-path $\langle i, j, s \rangle$
h_k := maximum allowed number of sub-paths on which commodity $k \in K$ can travel
Decision variables
$y_s \in \{0, 1\}$:= equal to 1 if service $s \in S$ is selected, and 0 otherwise
$x^{hk}_{ijs} \in \mathbb{R}^+$:= flow of commodity $k \in K$ using sub-path $\langle i, j, s \rangle$ as the h^{th} stage for $s \in S$, $\langle i, j, s \rangle \in A_s$, and $h = 1, 2, \ldots, h_k$
$z_k \in \mathbb{R}^+$:= unmet demand (number of containers) for commodity $k \in K$

Table 15.5 Additional notation for Sect. 4.2

Sets
S^v := set of maximum number of services for vessel class $v \in V$
Decision variables
$x^{ks}_{ij} \in \mathbb{R}^+$:= amount of commodity $k \in K$ transported in service $s \in S^v$ along arc $(i, j) \in A$
$\tau^s_i \in \mathbb{R}^+$:= departure time from port $i \in N$ of the vessel operating service $s \in S^v$
$w^s \in \mathbb{Z}^+$:= number of deployed vessels to maintain a weekly frequency in service $s \in S^v$
$\gamma^s_i \in \{0, 1\}$:= equal to 1 if port $i \in N$ is the hub port in the service $s \in S^v$, and 0 otherwise
$y^s_{ij} \in \{0, 1\}$:= equal to 1 if arc $(i, j) \in A$ is selected on the service $s \in S^v$, and 0 otherwise

Table 15.6 Additional notation for Sect. 4.2 (Reinhardt and Pisinger (2012))

Parameters
t_{max} := length of the time horizon
Decision variables
$x^{kv}_{ij} \in \mathbb{R}^+$:= amount of commodity $k \in K$ transported by vessel $v \in V$ along arc $(i, j) \in A$
$f^{kv}_j \in \mathbb{R}^+$:= amount of commodity $k \in K$ transshipped at port $j \in N$ by vessel $v \in V$
$f^{kv}_{jih} \in \mathbb{R}^+$:= amount of commodity $k \in K$ arriving at port $i \in N$ through arc $(j, i) \in A$ and not leaving in arc $(i, h) \in A$ by vessel $v \in V$
$e^v_{ij} \in \mathbb{Z}^+$:= position of arc $(i, j) \in A$ in the service of vessel $v \in V$
$\tau_v \in \mathbb{R}^+$:= route length of the service operated by vessel $v \in V$
$y^v_{ij} \in \{0, 1\}$:= equal to 1 if arc $(i, j) \in A$ is selected in the service for vessel $v \in V$, and 0 otherwise
$z^v_{ij} \in \{0, 1\}$:= equal to 1 if arc $(i, j) \in A$ is either the first or the last arc in the service for vessel $v \in V$, and 0 otherwise
$\gamma^v_i \in \{0, 1\}$:= equal to 1 if port $i \in N$ is the hub port in the service for vessel $v \in V$, and 0 otherwise
$\lambda^v \in \{0, 1\}$:= equal to 1 if vessel $v \in V$ is deployed for operating a service, and 0 otherwise

Table 15.7 Additional notation for Sect. 4.3

Sets
H := set of time-units after discretising the weekly planning horizon
$\tilde{N}$:= set nodes in the time-space graph
A^S := set of sailing arcs in the time-space graph
A^T := set of transshipment arcs in the time-space graph

Table 15.8 Additional notation for Sect. 5.1

Sets
P^k := set of paths for commodity $k \in K$
P_a := set of paths using arc $a \in A$
Parameters
u_{ij}/u_a := flow capacity through arch $a = (i, j) \in A$
c^k_p := unit flow cost of commodity $k \in K$ through path $p \in P_k$
Decision variables
$x^k_{ij} \in \mathbb{R}^+$:= flow of commodity $k \in K$ through arc $(i, j) \in A$
$f^k_p \in \mathbb{R}^+$:= flow of commodity $k \in K$ through path $p \in P_k$

Table 15.9 Additional notation for Sect. 5.2

Sets
N_s := set of ports in service $s \in S$
$\bar{N}_s$:= set of ports available for inclusion into service $s \in S$
L_i := lockset containing the ports which are forbidden to remove, if port $i \in N_s \cup \bar{N}_s$ is included (removed)
Parameters
$\Delta^{\mathrm{R}+}_{is}$:= estimated revenue change from including port $i \in \bar{N}_s$ into service $s \in S$
$\Delta^{\mathrm{R}-}_{is}$:= estimated revenue change from removing port $i \in N_s$ from service $s \in S$
$\Delta^{\mathrm{T}+}_{is}$:= estimated duration change from including port $i \in \bar{N}_s$ into service $s \in S$
$\Delta^{\mathrm{T}-}_{is}$:= estimated duration change from removing port $i \in N_s$ from service $s \in S$
$\hat{m}_v$:= number of free vessels of vessel class $v \in V$
τ_s := round trip time for service $s \in S$
η^+_s := maximum number of inclusions into service $s \in S$
η^-_s := maximum number of removals into service $s \in S$
Decision variables
$x^+_{is} \in \{0, 1\}$:= equal to 1 if port $i \in \bar{N}_s$ is included into service $s \in S$, and 0 otherwise
$x^-_{is} \in \{0, 1\}$:= equal to 1 if port $i \in N_s$ is removed from service $s \in S$, and 0 otherwise
$\zeta_s \in \mathbb{Z}$:= change in number of vessels in service $s \in S$

Table 15.10 Additional notation for Sect. 5.3

Parameters
$c(x)$:= concave cost function of using an edge $(i, j) \in A$ for the flow x
Decision variables
$x_{ij}^k \in \mathbb{R}^+$:= flow of commodity $k \in K$ on edge $(i, j) \in A$

Acknowledgments Marielle Christiansen was partially funded by the Research Council of Norway via the Axiom project. Erik Hellsten, David Pisinger, and David Sacramento were partially funded by the Danish Maritime Fund via the FutureFeed project.

References

Agarwal, R., & Ergun, O. (2008). Ship scheduling and network design for cargo routing in liner shipping. *Transportation Science, 42*(2), 175–196.

Ahuja, R., Magnanti, T., & Orlin, J. (1993). *Network flows: theory, algorithms, and applications*. Upper Saddle River: Prentice Hall.

Álvarez, J. (2009). Joint routing and deployment of a fleet of container vessels. *Maritime Economics & Logistics, 11*(2), 186–208.

Andersson, H., Fagerholt, K., & Hobbesland, K. (2015). Integrated maritime fleet deployment RoRo shipping. *Computers & Operations Research, 55*, 233–240.

Balakrishnan, A., & Karsten, C. (2017). Container shipping service selection and cargo routing with transshipment limits. *European Journal of Operational Research, 263(2)*, 652–663.

Brouer, B., Álvarez, J., Plum, C., Pisinger, D., & Sigurd, M. (2014a). A base integer programming model and benchmark suite for liner-shipping network design. *Transportation Science, 48*(2), 281–312.

Brouer, B., Desaulniers, G., Karsten, C., & Pisinger, D. (2015). A matheuristic for the liner shipping network design problem with transit time restrictions. In F. Corman, S. Voß, & R. Negenborn (Eds.), *Computational logistics* (pp. 195–208). Berlin: Springer.

Brouer, B., Desaulniers, G., & Pisinger, D. (2014b). A matheuristic for the liner shipping network design problem. *Transportation Research Part E: Logistics and Transportation Review, 72*, 42–59.

Brouer, B., Karsten, C., & Pisinger, D. (2016). Big data optimization in maritime logistics. In A. Emrouznejad (Ed.), *Big data optimization: recent developments and challenges, studies in big data* (Vol. 18, pp. 319–344). New York City: Springer International Publishing.

Brouer, B., Karsten, C., & Pisinger, D. (2017). Optimization in liner shipping. *4OR, 15*(1), 1–35.

Christiansen, M., Fagerholt, K., Nygreen, B., & Ronen, D. (2013). Ship routing and scheduling in the new millennium. *European Journal of Operational Research, 228*(3), 467–483.

Christiansen, M., Fagerholt, K., & Ronen, D. (2004). Ship routing and scheduling: Status and perspectives. *Transportation Science, 38*(1), 1–18.

Christiansen, M., Hellsten, E., Pisinger, D., Sacramento, D., & Vilhelmsen, C. (2020). Liner shipping network design. *European Journal of Operational Research, 286*(1), 1–20.

Fagerholt, K., Johnsen, T., & Lindstad, H. (2009). Fleet deployment in liner shipping: A case study. *Maritime Policy & Management, 36*(5), 397–409.

Germanische Lloyd. (2017). http://www.balkanlloyd.com/news/96-germanischer-lloyd-has-conducted-research-showing-that-new-and-efficient-4-500-teu-panamax [Online; accessed February 20, 2017]

Hellsten, E., Pisinger, D., Sacramento, D., & Vilhelmsen, C. (2019). Green liner shipping network design. In *Sustainable shipping* (pp. 307–337). Berlin: Springer.

ISL. (2016). Shipping statistics and market review. Technical Report. Institute of Shipping Economics and Logistics

Karsten, C., Brouer, B., Desaulniers, G., & Pisinger, D. (2017). Time constrained liner shipping network design. *Transportation Research Part E: Logistics and Transportation Review, 105*, 152–162.

Karsten, C., Pisinger, D., Røpke, S., & Brouer, B. (2015). The time constrained multi-commodity network flow problem and its application to liner shipping network design. *Transportation Research Part E: Logistics and Transportation Review, 76*, 122–138.

Koza, D., Desaulniers, G., & Ropke, S. (2020). Integrated liner shipping network design and scheduling. *Transportation Science, 54*(2), 512–533.

Krogsgaard, A., Pisinger, D., & Thorsen, J. (2018). A flow-first route-next heuristic for liner shipping network design. *Networks, 72*(3), 358–381.

Lee, C., & Song, D. (2017). Ocean container transport in global supply chains: Overview and research opportunities. *Transportation Research Part B: Methodological, 95*, 442–474.

Meng, Q., Wang, S., Andersson, H., & Thun, K. (2014). Containership routing and scheduling in liner shipping: overview and future research directions. *Transportation Science, 48*, 265–280.

Miller, C., Tucker, A., & Zemlin, R. (1960). Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM), 7*(4), 326–329.

Montreuil, B. (2011). Towards a physical internet: meeting the global logisitcs sustainability grand challenge. *Logistics Research, 3*, 71–87.

Notteboom, T., & Vernimmen, B. (2009). The effect of high fuel costs on liner service configuration in container shipping. *Journal of Transport Geography 17*(5), 325–337.

Pantuso, G., Fagerholt, K., & Wallace, S. (2015). Uncertainty in fleet renewal: A case from maritime transportation. *Transportation Science, 50*(2), 390–407

Plum, C., Pisinger, D., & Sigurd, M. (2014). A service flow model for the liner shipping network design problem. *European Journal of Operational Research, 235*(2), 378–386.

Psaraftis, H., & Kontovas, C. (2015). Slow steaming in maritime transportation: Fundamentals, trade-offs, and decision models. In C. Lee & Q. Meng (Eds.), *Handbook of ocean container transport logistics: making global supply chains effective* (pp. 315–358). New York: Springer International Publishing.

Reinhardt, L., & Pisinger, D. (2012). A branch and cut algorithm for the container shipping network design problem. *Flexible Services and Manufacturing Journal, 24*(3), 349–374.

Ronen, D. (1983). Cargo ships routing and scheduling: Survey of models and problems. *European Journal of Operational Research, 12*(2), 119–126.

Ronen, D. (1993). Ship scheduling: The last decade. *European Journal of Operational Research, 71*(3), 325–333.

Sun, Z., & Zheng, J. (2016). Finding potential hub locations for liner shipping. *Transportation Research Part B: Methodological, 93*, 750–761.

Thun, K., Andersson, H., & Christiansen, M. (2017). Analyzing complex service structures in liner shipping network design. *Flexible Services and Manufacturing Journal, 29*(3–4), 535–552.

Tran, N., & Haasis, H. (2015). Literature survey of network optimization in container liner shipping. *Flexible Services and Manufacturing Journal, 27*, 139–179.

Unctad. (2018). Review of maritime transport. Technical Report. United Nations Conference on Trade and Development

Unctad. (2019). UnctadSTAT. https://unctadstat.unctad.org/wds/ReportFolders/reportFolders.aspx, [Online; accessed December 9, 2019]

Chapter 16
City Logistics

Teodor Gabriel Crainic, Guido Perboli, and Nicoletta Ricciardi

1 Introduction

The political, economic, and social evolution of society challenges transportation and logistics with new objectives, challenges, and constraints. Efficiency of operations, consistently bringing freight at the designated destination, within the agreed upon time, at the lowest possible cost and with the highest possible quality of service is still, of course, a major goal, as are the profitability of the firms and the economic development of the cities, regions, and countries involved. Increasingly, however, the long-term sustainability of the industry and the society it supports is challenged by citizens and governments alike, bringing into focus the need to reduce the negative externalities of transportation and logistics, in particular, the high levels of congestion in many cities and regions of the world, the environmental damage through emissions and consumption of fossil fuels, and the safety and security of the people. A number of new organization and business models have been proposed to address these issues and aim to conciliate, better still, to jointly "optimize" the economic and social goals of transportation and logistics. City Logistics, the Physical Internet, and their recent combination into Hyperconnected City Logistics, belong to this group. They also share a number of fundamental characteristics,

T. G. Crainic (✉)
CIRRELT and AOTI, Université du Québec à Montréal, Montréal, QC, Canada
e-mail: TeodorGabriel.Crainic@cirrelt.net

G. Perboli
CIRRELT and Department of Management and Production Engineering, Politecnico di Torino, Turin, Italy
e-mail: Guido.Perboli@polito.it

N. Ricciardi
CIRRELT and Department of Statistical Sciences, Sapienza Università di Roma, Rome, Italy
e-mail: nicoletta.ricciardi@uniroma1.it

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_16

including multi and intermodality, cooperation among stakeholders, consolidation, synchronization of operations, resource sharing, and the separation of commercial transactions generating cargo movements from the planning and execution of the corresponding activities, the degree and way of inclusion of each defining variants and applications.

We focus on *City Logistics* in this chapter, as it has a history going back to the 1990s and has generated a good number of studies, implementations, and discussions on success and failure factors. In fact, even though not all proposals and implementations were successful, the City Logistics concepts and proposals are influencing the development and deployment of urban freight transportation and logistics systems. It influences national policy in several countries in Europe and Asia, brings new systems to several cities around the world, and is the source of new forms of private distribution networks (e.g., last-mile service providers, express couriers, and large retail chains).

In its most fundamental meaning, City Logistics aims to reduce the externalities and nuisances, e.g., emissions and, more generally, the environmental footprint, associated to the transportation of freight within urban areas, while sustaining the social and economic development of the organizations and cities involved. City Logistics encompasses several dimensions, and may target demand estimation or the design and organization of the supply activities servicing it. City Logistics studies may focus on a single organization or on several organizations interacting in a city.

The operations research-based research and development of models and methods to support the planning and management of City Logistics systems really got started in the first years of the millennium and it is steadily growing ever since. Network design is one the main methodologies used in this context, the particular settings and characteristics of City Logistics systems bringing modeling challenges and conducting to new formulations, e.g., several layers of facilities and operations, time-dependency of demand and activities, synchronization of fleets at terminals, integration of private and public transportation and logistic means, and combining network design and vehicle routing, to name but a few. It appears that network design models for City Logistics are mainly directed at the strategic and tactical planning of the system. This chapter aims to capture these characteristics and present the network design methodology currently available.

The chapter is organized as follows. Section 2 recalls the two-tier City Logistics setting we use this chapter to describe issues and models, together with the tactical and strategic planning issues that call for network design methodologies. Section 3 proposes a general scheduled service network design modeling framework for strategic and tactical planning of City Logistics systems. Section 4 then focuses on how one may use the modeling framework, as well as a number of important problem and model variants, including the connection to multi-echelon routing, the representation of delivery and pickup operations and costs, the selection of particular corridors or infrastructures, and the explicit consideration of uncertainty. Bibliographical notes are the topic of Sect. 5, recalling the historic developments and the main contributions to the field. We discuss perspectives for other new transportation concepts proposed, identify a number of challenging research avenues, and conclude in Sect. 6

2 City Logistics, Planning, and Design

We describe a "general" *Two-Tier City Logistics*, *2T-CL*, system setting in Sect. 2.1, which includes the most important characteristics found in literature and practice, even though not all characteristics are currently found within the same systems. We use the 2T-CL setting to illustrate City Logistics planning issues and models.

Two-tier systems have been proposed both for single organizations and for multiple organizations, e.g., carriers and other service providers, operating in (parts of) cities under some form of cooperation and resource sharing. We therefore assume in this chapter that the 2T-CL system is planned and managed by a single manager/decision maker, even though resources may be provided and operated by several private and public stakeholders involved in some form of cost/profit/risk-sharing collaboration. We discuss a number of possible modeling consequences of such collaborations in Sect. 5.

2T-CL are consolidation-based systems involving multiple resources in complex interactions, and thus require advanced planning methods, particularly at the tactical and strategic levels where system-wide decisions are made for medium to long-term planning horizons. These planning issues are recalled in Sect. 2.2.

2.1 A Two-Tier Setting

Figure 16.1 illustrates the structure of a 2T-CL system that, similar to any transportation system, 2T-CL has a demand and a supply component. The latter, composed of facilities and the transportation modes and services moving freight among them and customers, is designed and planned to answer the requests of the former according to some performance criteria. In particular, a 2T-CL system aims to satisfy demand and contribute to the sustainable development of the city, that is, to deliver goods from origins to destinations, on time, economically in monetary terms, and efficiently from the societal point of view of the impact on the city. This impact accounts for the street or neighborhood characteristics, e.g., touristic, residential, social (schools, hospitals, leisure, culture, etc.), administrative, etc., and can be defined in terms of emissions, noise, visual degradation, contribution to congestion, and so on. The particular measures considered, as well as their estimation processes, are application specific. While multi-objective optimization may be used to reflect these different measures, economic and impact measures are often combined for planning purposes into a *transportation* or *city-infrastructure-utilization cost* associated to the network representation used in the model. This is the approached used in this chapter.

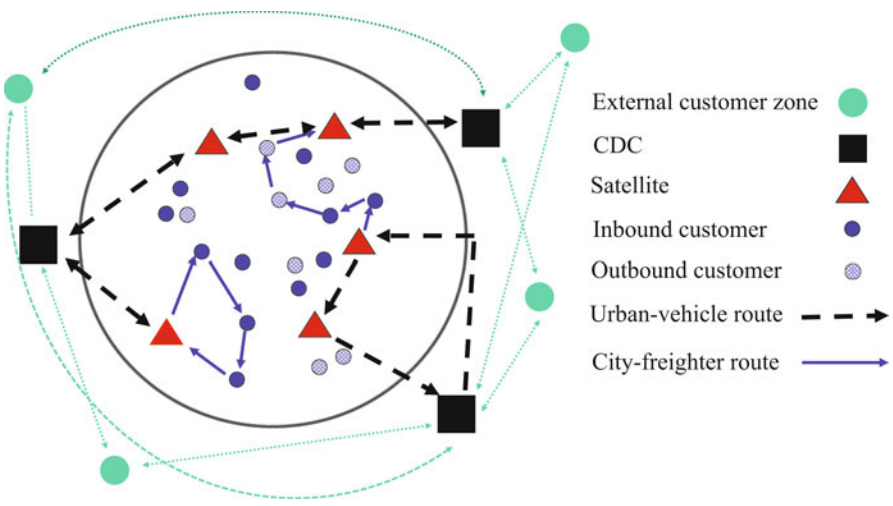


Fig. 16.1 Two-tier city logistics structure

In all generality, the customers of a City Logistics system are all the firms, organizations, institutions, and private citizens that ship or receive freight through the system. For planning purposes, particularly at tactical and strategic levels, the many possible external origins and destinations of such shipments are aggregated into a number of *external zones* (large disks in Fig. 16.1), connected to the city by various transportation modes. Similarly, locations within the city are aggregated into customer-demand zones, referred to as *customer zones* in the following and represented by small disks in Fig. 16.1.

The *demand* side of the system thus consists of a set of freight loads that need to be moved between particular pairs of customer and external zones. Three types of demand may be identified. *Inbound demand* is to be delivered from external-zone origins to customers in the city. Symmetrically, *outbound demand* is to be picked up at customers in the city, and then be shipped to specified external-zone destinations. *Local demand* is to be picked up from a customer within the city and delivered to another customer within the city. Each individual origin-destination (*OD*) demand is characterized by a volume to be moved and time-related features, e.g., its *availability* time representing when the demand would enter the system and could start to be handled, as well as delivery and pick-up *time windows* at customers (locations within the city). More than one physical product is generally handled. To simplify the presentation, but without loss of generality, we assume that the 2T-CL system makes use of smart containers, as those developed within the Physical Internet initiatives, which can be loaded together irrespective of their content; such boxes are referred to as π*-containers* (Montreuil 2011). In modeling terms, this yields problems that are single product, the loading container, but multicommodity in the OD demand treated.

The supply side of a 2T-CL system consists of two layers of terminals and the transportation means linking them and the customer and external zones. *City distribution centers* (*CDC*s) are generally located at the border of the city, close to main interurban transportation infrastructures or intermodal facilities. Inbound and outbound loads are sorted and consolidated at CDCs for distribution into the city or long-haul transportation to external zones destinations. Consolidated inbound demand is shipped, using *urban vehicles*, to intermediary facilities called *satellites* located close to the parts of the city where traffic is controlled and certain vehicle types, e.g., large trucks, are not allowed to penetrate. It is at satellites, illustrated through triangles in Fig. 16.1, that the junction between first and second-tier vehicles takes place, freight being generally transferred between synchronized vehicles according to transdock principles, with little or no temporary storage. *City freighters*, second-tier vehicles, bring inbound freight from satellites to customers, and pick up outbound and local freight at customers delivering it to satellites and destination customers, respectively. The outbound loads brought to a satellite at the "same" time by several city freighters are loaded into urban vehicles to be moved to the appropriate CDC from where the goods are shipped to their final external zones. First and second-tier vehicles carrying inbound or outbound cargo could thus be present simultaneously at a satellite, competing for the capacity it offers for vehicle docking, parking, and cargo transfer.

Several modes and vehicles types make up the *multi/intermodal transportation system* (π-containers make the system intermodal) connecting the facilities and the customers. City freighters, operating at the second tier, may be small vans, eco-friendly electric or hydrogen small vans, traditional or cargo bikes, canal or river barges, individuals using their own cars, etc. The spectrum of possible modes is even larger on the first tier as, increasingly, the utilization of what may be described as massive-flow modes and vehicles, e.g., regular and light rail, is contemplated or tested. We distinguish between *line-based* and *no-line* transportation modes and services. The latter include the various trucks and barges for which one may define services along any path within their admissible network (e.g., a city "trucking network" is often defined to restrict circulation). Line-based modes are often captive of particular infrastructure, such as passenger buses, which are "captive" of their predefined lines, regular and light rail (tramways, subways, etc.) captive of their tracks, and trolleybuses captive of their aerial power lines. This characteristic restricts the definition of line-based services to the network of the corresponding infrastructure, but not necessarily to the service lines and stops operated for passengers. Two main approaches for line-based services are being contemplated within City Logistics projects around the world. On the one hand, regular vehicles may be equipped with special compartments for the transportation of goods while, on the other hand, freight-dedicated vehicles may be operated on the same infrastructure, either independently or as parts of regular convoys.

When different demand types, inbound, outbound, local, may be loaded into urban vehicles and city freighters, loading/unloading rules must be defined. To keep the presentation simple, we assume in the following a *pseudo-backhaul* policy, which means that a vehicle completes the current type of activity before initiating a different one. Thus, for example, a vehicle may leave the depot, perform a sequence of pickup and delivery activity for local demand, and then move to a satellite to synchronize with arriving first-tier urban vehicles and load for a delivery phase of inbound demand. This policy is based on the idea that operations at satellites and customers should be streamlined. Indeed, capacity, time-window, and synchronization restrictions and requirements, as well as the goal of reducing the presence of vehicles in the city, implies efficient vehicle unloading and loading activities, which makes searching for and resorting of loads undesirable.

2.2 Planning and Design

City Logistics belongs to the important class of consolidation-based transportation systems that include rail and less-than-truckload carriers, high-sea navigation lines, intermodal systems, postal services, and so on. Such complex systems require planning at all levels. We briefly recall the issues associated to tactical and strategic planning, as this is where the interactions with network design are the strongest.

Tactical planning for consolidation freight carriers aims to select and schedule services, together with the itineraries used to move freight flows from origins to destinations in the resulting service network. The goal is to satisfy the regular demand in the most cost- and resource-utilization efficient way possible, while satisfying the service-quality levels set by the carrier to answer customer requirements. The tactical plan is thus also generally yielding activity profiles of terminals and the resources required to support the selected services. The service network and plan is determined for a rather short period called *schedule length*, e.g., a day or a week, and it is then repeatedly applied over a certain medium-term planning horizon, the *season*, e.g., 6 months. Note that this decision process assumes that the major elements of the plan, the demand, selected scheduled services, and main resource assignments to services and terminals, will not be modified during regular operations for the length of the planning horizon. Adjustments of the plan to actual demand are then mostly performed through modifications to the routing of demand flows at operation time.

In City Logistics terms, tactical planning targets the regular demand and is about selecting the services and resources that will be operated regularly and repeatedly. In this sense, first-tier service operations should be more stable, particularly for line-based services, as they move larger loads between CDCs and satellites. With this regularity comes the regularity in using the terminal facilities and assigning customers to satellites, while the actual routing may vary from day to day according to the particular demand. The goal is to determine the most cost-effective plan to satisfy forecast demands with the available resources, where the generalized transportation costs account for operations-related costs and city-impact considerations. With such a plan, material and human resources can be allocated for the duration of the planning horizon, which makes management easier and lowers costs.

Strategic planning generally addresses longer-term decisions than tactical ones, with impacts normally valid for years. Setting up a City Logistics system within a given city or part thereof is certainly strategic in nature, as well as the associated decisions on zones covered, partnerships, network and service type (e.g., single or two-tier), legal and financial framework, etc. Most of these issues are generally not of an Operations Research type. Yet, they may be, and hopefully are supported by Operations Research models providing quantitative evaluations and analyses of contemplated systems and policies. Network design is at the core of such methodologies. On the one hand, tactical-planning models provide performance-evaluation tools for contemplated or existing City Logistics system and policy designs. On the other hand, network design models may be built to answer strategic-level decisions such as the number, locations, and characteristics of facilities, CDCs and satellites to built or select, the construction of dedicated infrastructure or the connection of the City Logistics facilities to public transport infrastructure and services, the types of vehicles to use and the dimensions of the fleets, etc.

In the next section, we present a general scheduled service network design model, incorporating the main tactical-planning issues previously discussed. We then discuss specializations and extensions able to address particular settings of the tactical and strategic planning problems.

3 A General SSND Modeling Framework

Scheduled service network design (*SSND*) formulations defined over time-space networks are generally used to model tactical planning problems for consolidation-based freight carriers (Chap. 12). We present in this section a general SSND model for the tactical planning of the 2T-CL system described in Sect. 2.1, addressing the activities at both tiers and their synchronization at satellites, for a time-dependent demand. This path-based SSND formulation thus addresses the main issues of (1) selecting a subset of scheduled services out of the set of possible line- and no-line-based multimodal services; (2) building the multi-tour routes of the second-tier city freighters, (3) determining the itineraries of each demand through the selected City Logistics service network, including the assignment to a CDC and a satellite for inbound and outbound demands; and (4) managing the terminals and the multimodal fleets of urban vehicles and city freighters, which connect and synchronize at satellites. The optimization is performed for a given schedule length, City Logistics infrastructure, potential first and second-tier services, and a deterministic estimation of demand, travel times, and activity times at facilities and customers. The formulation thus combines characteristics of classical SSND models (Chap. 12) and multi-attribute vehicle routing formulations. The goal is to minimize a total generalized cost, reflecting both the economics of operating the system and the potential impact on the city, while satisfying demand. The resulting service plan is supposed to be used repeatedly over a certain planning horizon (Sect. 4).

In a City Logistics context, the schedule length is relatively short, from a few hours to half a day and, thus, each of its $t = 1, \ldots, T$ periods is also relatively short. To simplify the presentation, we assume that the length of any activity, vehicle movement or terminal operation, is an integer number of periods. The external and customer zones defining demand, as well as the CDC and satellite locations are modeled as nodes connected through modal arcs within a physical network representing the 2T-CL system. The scheduled service network design model is built on a time-space network, where the physical nodes are duplicated at all relevant time periods (customer-related nodes are not represented outside the corresponding temporal attributes). This yields the set of nodes $\mathscr{N}$ encompassing the sets $\mathscr{E}$ (external zones), $\mathscr{C}$ (customer zones), $\mathscr{F}$ (CDCs), and $\mathscr{Z}$ (satellites). A set of links $\mathscr{A} = \{a = (i, j)\}$ representing the modal transportation and terminal-holding activities connecting these nodes completes the network representation $\mathscr{G} = (\mathscr{A}, \mathscr{N})$.

Movements are performed by vehicles of different types and modes. Let $\mathscr{M}$ be the set of transportation modes and $\mathscr{T}_m$ and $\mathscr{V}_m$ the sets of urban-vehicle and city-freighter types, respectively, for mode $m \in \mathscr{M}$; the respective vehicle capacities are $u_\tau,\ \tau \in \mathscr{T}_m$ and $u_\nu,\ \nu \in \mathscr{V}_m$. Let $n_{f\tau}$ be the fleet size of urban-vehicle type τ at CDC $f \in \mathscr{F}$, and n_ν the fleet size of city-freighter type ν at the second tier garage.

In most cases, city distribution centers are large facilities, where capacity issues are not critical and sufficient space is available for vehicles to wait for loading and unloading activities. This is, however, not the case for satellites, where the space available for transferring goods limits the number of urban vehicles and city freighters which can be present simultaneously. Furthermore, there is generally no space available for storing goods at satellites, nor for vehicles to wait. The satellite capacity may also be time dependent, due to either opening hours or operations on a shared infrastructure. For example, a passenger tramway cannot wait longer than "normal" at a station because of unloading or loading activities of another tramway with freight somewhere down the line. Several capacity measures account for these limitations for each satellite z and must be enforced at each period: 1) u_z^{T} for the total number of urban vehicles it may accommodate, with $u_{z\tau}^{\mathrm{T}}$ for the number of urban vehicles of type τ of mode m it may accommodate; for no-line modes, this is the actual number of vehicles, while for line-based modes (e.g., tramways) it is the number of available tracks (but could also be the number of cars in a convoy); 2) u_z^{V} for the total number of city freighters it may accommodate; 3) u_z^{K} for the total volume of goods the satellite may handle.

Inbound, outbound, and local requests for transportation make up the demand of the system. When the same customer location is both the origin and the destination of demand, separate nodes are created, the respective demands being then treated individually within the model. Each customer demand $k \in \mathscr{K}$ is defined between a pair $(O(k), D(k))$ of origin—destination nodes in $\mathscr{N}$, where $(O(k) \in \mathscr{E}$ and $D(k) \in \mathscr{C})$ for inbound demand, $(O(k) \in \mathscr{C}$ and $D(k) \in \mathscr{E})$ for outbound, and $(O(k),\ D(k) \in \mathscr{C})$ for local. A volume $d(k)$ is to be moved between these nodes. Time attributes specify when the volume is available at origin, $[a(O(k)),\ b(O(k))]$ and must be delivered at destination $[a(D(k)),\ b(D(k))]$ (some time windows might not be bidding). External zones, the out-of-city origins and destinations, are linked by various transportation modes to the city and the CDCs. The CDC to be used by each demand is to be selected by the model. Let $\mathscr{F}(k) \subseteq \mathscr{F}$ be the set of potential CDCs that may be used for demand k, and $c_{ft}(k)$ the cost of assigning demand k to CDC $f \in \mathscr{F}(d)$ at time t (it could account, e.g., for using another CDC rather than the closest one, using inter-CDC transportation or short-term storage).

Let $\Sigma = \{\sigma\}$ be the set of potential urban-vehicle *services*. Service σ operates a vehicle of type $\tau(\sigma)$, originates at CDC $O(\sigma)$, travels to one or several satellites $\mathscr{Z}(\sigma)$, and returns to CDC $D(\sigma)$, possibly different from $O(\sigma)$ (this may be the case even for line-based modes, when the line connects two CDCs). The urban-vehicle route is thus composed of a series of *legs*, from the CDC to the first satellite, from the latter to the second one, until the last leg from the last satellite to the destination CDC. Let $\mathscr{L}(\sigma)$ be the set of these legs. The schedule of service σ is given by $t(\sigma)$, the departure time from $O(\sigma)$, as well as by the arrival and departure times at all the satellites in $\mathscr{Z}(\sigma)$, which account for the travel time along the arcs of the specific mode as well as the loading and unloading times at satellites. The cost associated to operating service $\sigma \in \Sigma$ is denoted $c(\sigma)$. The cost captures not only the monetary expenses of operating the service, but also the city-infrastructure-utilization cost reflecting the "nuisance" factors related to the presence of the urban vehicle in the city at the particular time of the service.

The second-tier pickup and delivery activities between satellites and customer zones are performed by city freighters operating multi-tour synchronized routes called *work assignments*. A city-freighter work assignment $h \in \mathscr{H}$ operates a city freighter of type $\nu(h)$ over a sequence of work *segments* $w \in \mathscr{W}(h)$, separated by returns to the garage, each segment being made up of visits to satellites to load and unload freight and one or several pickup (outbound demand), delivery (inbound demand), and pickup-and-delivery (local demand) activity phases. The schedule of a work segment starts at period $t(w)$ at the first satellite or customer on its route, and continues with the arrival and departure times at the visited satellites and customers. The cost of operating city-freighter work assignment h, $c(h)$, includes the costs of its segments, those associated to the garage (back and forth movements, idles time, etc.), a vehicle fixed cost, and the city-infrastructure-utilization cost reflecting the "nuisance" factors related to the presence of the city freighter in the city at the particular time of the service.

Freight is moved from the origin to the destination of demand via *itineraries* that include the facilities and the first and-second tier services used. Inbound-demand itineraries are thus made up of the movement from the external zone to a CDC, an urban-vehicle movement, a transsphipment operation at a satellite, and the final distribution by a city-freighter work segment. Outbound-demand itineraries involve the same operations in reverse order. Local-demand itineraries are simpler as they involve the work segment performing the pickup and delivery only. Let $\mathscr{I}(k)$ stand for the set of itineraries that may be used to satisfy customer demand k, with itinerary $i \in \mathscr{I}(k)$ being defined by its selected CDC $f(i)$, urban-vehicle service $\sigma(i)$, satellite $z(i)$, and work segment $w_h(i)$ of work assignment $h(i)$.

Three sets of decision variables are defined to select urban-vehicle services, city-freighter work assignments, and demand itineraries, respectively:

- $y(\sigma) = 1$, if the urban-vehicle service $\sigma \in \Sigma$ is selected, 0, otherwise;
- $\varphi(h) = 1$, if the work assignment $h \in \mathscr{H}$ is selected, 0, otherwise;
- $\xi(i) = 1$, if itinerary $i \in \mathscr{I}(k)$ of demand $k \in \mathscr{K}$ is used, 0, otherwise.

The goal of the SSND formulation is to minimize the number, cost and impact of vehicles in the city, while satisfying demand requirements and capacity limitations. The formulation when no splitting of demand is allowed then becomes:

$$\text{Minimize} \quad \sum_{\sigma \in \Sigma} c(\sigma) y(\sigma) + \sum_{h \in \mathscr{H}} c(h) \varphi(h) \tag{16.1}$$

$$\text{Subject to} \quad \sum_{i \in \mathscr{I}(k)} \xi(i) = 1, \quad \forall k \in \mathscr{K}, \tag{16.2}$$

$$\sum_{k \in \mathscr{K}(\sigma l)} \sum_{i \in \mathscr{I}(k) | \sigma(i) = \sigma} d(k) \xi(i) \le u_{\tau(\sigma)} y(\sigma), \quad \forall l \in \mathscr{L}(\sigma), \forall \sigma \in \Sigma, \tag{16.3}$$

$$\sum_{k \in \mathscr{K}(w(h)t)} \sum_{i \in \mathscr{I}(k) | h(i) = h} d(k) \xi(i) \le u_{\nu(h)} \varphi(h)$$

$$\forall w \in \mathscr{W}(h), \ \forall h \in \mathscr{H}, \ t = 1, \ldots, T, \tag{16.4}$$

$$\sum_{\sigma \in \Sigma(z,t)} y(\sigma) \le u_z^{\mathrm{T}}, \quad \forall z \in \mathscr{Z}, \ t = 1, \ldots, T, \tag{16.5}$$

$$\sum_{\sigma \in \Sigma(z,t,\tau)} y(\sigma) \le u_{z\tau}^{\mathrm{T}}, \forall z \in \mathscr{Z}, \forall \tau \in \mathscr{T}_m, \forall m \in \mathscr{M}, t = 1, \ldots, T, \tag{16.6}$$

$$\sum_{h \in \mathscr{H}(z,t)} \varphi(h) \le u_z^{\mathrm{V}}, \quad \forall z \in \mathscr{Z}, t = 1, \ldots, T, \tag{16.7}$$

$$\sum_{i \in \mathscr{I}(z,t)} d(k) \xi(i) \le u_z^{\mathrm{K}}, \quad \forall z \in \mathscr{Z}, t = 1, \ldots, T, \tag{16.8}$$

$$\sum_{\sigma \in \Sigma(f,t,\tau)} y(\sigma) \le n_{f\tau}, \forall f \in \mathscr{F}, \ \forall \tau \in \mathscr{T}_m, \forall m \in \mathscr{M}, \ t = 1, \ldots, T, \tag{16.9}$$

$$\sum_{h \in \mathscr{H}(\nu)} \varphi(h) \le n_\nu, \quad \forall \nu \in \mathscr{V}_m, \forall \, m \in \mathscr{M}, \tag{16.10}$$

$$y(\sigma) \in \{0, 1\}, \quad \forall \sigma \in \Sigma, \tag{16.11}$$

$$\varphi(h) \in \{0, 1\}, \quad \forall h \in \mathscr{H}, \tag{16.12}$$

$$\xi(i) \in \{0, 1\}, \quad \forall i \in \mathscr{I}(k), \quad \forall k \in \mathscr{K}. \tag{16.13}$$

The objective function (16.1) computes the total generalized cost of the system (operations and negative impact on the city) as the sum of the costs of the selected urban-vehicle services and city-freighter work assignments. Constraints (16.2) indicate that each demand must be satisfied by a single itinerary. The formulation may be easily modified to account for the case when demand may be split by (1) using continuous itinerary flow variables instead of the selection ones, and (2) imposing in constraints (16.2) that the sum of these flows equals the demand volume.

Let $\mathscr{K}(\sigma l)$ be the set of all demands k that may use leg l of urban-vehicle service σ (i.e., there is at least an itinerary of k that includes leg l). Similarly, let $\mathscr{K}(w(h)t)$ be the set of all demands k that may use segment w of city-freighter work assignment h at time t. Then, constraints (16.3) enforce the urban-vehicle capacity restrictions for each leg of the vehicle route. Similarly, constraints (16.4) enforce city-freighter capacity restrictions at all time for each segment of a work assignment. These last two groups of relations are the linking constraints of network design formulations.

Define, (1) $\Sigma(z, t)$ ($\Sigma(z, t, \tau)$) and $\mathscr{H}(z, t)$, the sets of urban-vehicle services (of type τ) and city-freighter work assignments, respectively, stopping at satellite z at period t; (2) $\Sigma(f, t, \tau)$, the set of services initiated at or before period t that are still active at period t; and (3) $\mathscr{I}(z, t)$, the set of demand itineraries using satellite z at period t to load or unload freight. Then, constraints (16.5)–(16.8) enforce the satellite capacity restrictions in terms of total numbers of urban vehicles (services) (16.5), mode-specific urban vehicles (16.6), city freighters (16.7), and freight handled (16.8). Note that the coherence of the respective numbers of urban vehicles and city freighters present simultaneously at satellites is provided by the flow of freight imposed by the demand itineraries. Constraints (16.9) limit the number of services of each type operated out of each CDC at period t to the available fleet at the respective CDC. Constraints (16.10) perform the same role for the city freighters (work assignments) of each type. Constraints (16.11)–(16.13) define the range of the decision variables.

4 Using the Modeling Framework

The previous SSND formulation constitutes a modeling framework that can be adapted and extended to address a rather broad range of planning issues. We discuss a number of those in this section.

Recall that problem definitions and the corresponding models must account not only for the particular system setup and the planning issues addressed, and also the operation and management policies involved. Two aspects of the latter have a particular impact on the design of SSND formulations, the estimation of major exogenous factors, e.g., demand, travel and service times, and costs of goods and services, and the degree of freedom in managing resources. Indeed, tactical planning assumes a certain level of look-ahead capability and the inclusion of an evaluation of future events and their consequences into today's decision processes, through forecasts for the planning horizon considered. Tactical planning also implicitly assumes managerial capabilities to assign and schedule resources in a way that matches the requirements of the tactical plan. The choice of an appropriate modeling/methodological approach is then related to the magnitude of the variability of those factors, the confidence one has in the forecasts, and the amplitude of the managerial capabilities. A rather broad spectrum of problem settings and formulations is thus possible.

Consider, to illustrate, the case of high variability of demand, combined to a low or no confidence in the possibility to adequately forecast it and a high capability to manage resources, generally implies that no advanced planning of operations is possible. The system then reacts to new or varying demand by assigning and dispatching resources to service it. Such a dynamic mode of operations is not, however, within the scope of the problem settings and methodology considered in this chapter. At the other end of the spectrum, deterministic models, e.g., Sects. 3 and 4.1, are appropriate in cases of high confidence in forecasts or estimated low-variability for the duration of the planning horizon. Between these cases, one finds problem settings where one represents the future through some probability distribution.

The literature is very sparse regarding the explicit integration of uncertainty in tactical-planning models and methods for City Logistics. To the best of our knowledge, duration and cost uncertainty, in particular, has not been addressed to any significant extent, a few contributions targeting demand uncertainty. The impact of demand uncertainty on the tactical plan is then generally accounted for through two-stage stochastic programs (Chap. 10), the design decisions selecting the service network appearing in the first stage, while routing is decided in the second. We illustrate such a formulation in Sect. 4.2.

Relative to the management environment and constraints, planning closely to operation-time is beneficial when one has little or no restrictions on mustering facilities and people on very short notice. The so-called *day-before planning* problem class and formulation of Sect. 3 corresponds to this situation. In most cases, however, management is significantly more constrained. Labor contracts often restrict the possible modifications to schedules. The inclusion of massive transportation means, particularly when related to passenger transportation, also involves strict scheduling of operations and advanced planning. The tactical planning formulation of Sect. 4.1 address these cases.

We conclude the section with a few comments on the longer-term planning of two-tier City Logistics systems in Sect. 4.3.

4.1 Tactical Planning for Medium-Term Horizons

Tactical planning is often concerned with structuring service, and the required resource assignment, to be repeatedly operated over a planning horizon several months long. Then, as discussed previously, including the precise routing of second-tier vehicles in the tactical plan appears less appropriate than for the day-before situation described in Sect. 3. One cannot neglect second-tier activities and costs, however, as they impact first-tier decisions, e.g., the freight itineraries and the synchronization with first-tier vehicles at satellites, and the global performance of the City Logistics system. The model described in this section then represents city-freighter routing through an *approximated cost* of servicing a customer zone out of each satellite to which it may be connected.

The inbound and outbound flows are thus captured together with the selection of the satellites which will service each customer zone, and the estimation of the dimensions of the city-freighter fleets required to satisfy demand, and the corresponding utilization of satellite capacity and city freighters. To simplify the presentation, the SSND tactical planning model bellow assumes, without loss of generality, a single city-freighter fleet and the same routing costs for inbound and outbound demand. As for the first-tier decisions, they are the same as before, namely, select the scheduled services to operate out of the set of possible line- and no-line-based multimodal services; determine the itineraries of each inbound and outbound demand, including the assignment to a CDC, a satellite, and a service with, possibly, a compartment; manage the multimodal fleets and terminals. For sake of simplicity, we do not repeat the notation in common with the model of Sect. 3 (see Table 16.1), and present only the new notation and modifications to the existing one.

The multicommodity demand $\mathscr{K}$ includes the inbound and outbound components, noted $\mathscr{K}^I$ and $\mathscr{K}^O$, respectively. Notice that the local demand is not considered in this model as it does not impact the design of the first-tier service network. All the characteristics defined in Sect. 3 are still valid. Similarly, the set of potential first-tier urban-vehicle services defined previously is also considered here, together with their characteristics, modes, types, and costs.

As indicated previously, the second tier is represented through an approximated cost of servicing customers out of satellites (delivery out of satellites to receiving customer zones and pick up at shipping customer zones to deliver at satellites and then CDCs) . To streamline the network representation and the model, however, no explicit satellite-customer arcs are added. Rather, the corresponding cost is added to the cost of the service carrying the flow of the particular demand into or out of the satellite. Let $\mathscr{Z}(k)$ be the set of satellites that may service demand k, and $c(k, z, \sigma)$ the approximated satellite-customer transportation cost of demand k moved in or out of satellite $z \in \mathscr{Z}(k)$ by a service σ (which could service the demand in time). Similar to all other costs defined in this chapter, the assignment costs $c(k, z, \sigma)$ represent not only the transport, unloading, and loading costs, but also city-disturbance factors related to these activities.

With respect to vehicles and modes, the previous definitions hold. We take the opportunity of this tactical planning SSND model, however, to introduce the notion of *compartment*. Several vehicle types have more than one cargo-holding space, as illustrated by the multiple cargo bays of river barges and several proposed cargo tramways, as well as the (vertical or horizontal) separators that may be used within motor vehicles. This definition may be broaden to a partition of a cargo-holding

Table 16.1 Main notation of the SSND model

$\mathscr{G} = (\mathscr{N}, \mathscr{A})$	Time-space network for a schedule length of T periods
$\mathscr{E} = \{e\}$; $\mathscr{C} = \{c\}$	Sets of external and customer zones
$\mathscr{F} = \{f\}$; $\mathscr{Z} = \{z\}$	Sets of facilities, CDCs and satellites
$\mathscr{M} = \{m\}$	Set of transportation modes
$\mathscr{T}_m = \{\tau\}$; $\mathscr{V}_m = \{\nu\}$	Sets of urban vehicles and city-freighter types of mode $m \in \mathscr{M}$
u_τ	Capacity of urban vehicle type $\tau \in \mathscr{T}_m, m \in \mathscr{M}$
$n_{f\tau}$	Fleet size of urban-vehicle type $\tau \in \mathscr{T}_m,\ m \in \mathscr{M}$, at CDC $f \in \mathscr{F}$
u_ν	Capacity of city-freighter type $\nu \in \mathscr{V}_m,\ m \in \mathscr{M}$
n_ν	Fleet size of city-freighter type $\nu \in \mathscr{V}_m, m \in \mathscr{M}$ at the second tier garage
u_z^{T}	Satellite z total capacity in number of urban vehicles of all types
$u_{z\tau}^{\mathrm{T}}$	Satellite z capacity in number of urban vehicles/tracks of type τ
u_z^{V}	Satellite z capacity in number of city freighters
u_z^{K}	Satellite z capacity in volume of goods it may handle
$\mathscr{K} = \{k\}$	Set of origin-destination demands
$O(k)$; $D(k)$; $d(k)$	Origin, destination, and quantity of demand $k \in \mathscr{K}$
$[a(O(k)), b(O(k))]$	Availability time interval of demand $k \in \mathscr{K}$ at its origin
$[a(D(k)), b(D(k))]$	Due date interval of demand $k \in \mathscr{K}$ at its destination
$\mathscr{F}(k) \subseteq \mathscr{F}$	Set of potential CDCs for demand $k \in \mathscr{K}$
$c_{ft}(k)$	Cost of assigning demand $k \in \mathscr{K}$ to CDC $f \in \mathscr{F}(d)$ at time t
$\Sigma = \{\sigma\}$	Set of potential urban-vehicle services
$O(\sigma)$; $D(\sigma)$	Origin and destination CDCs of service $\sigma \in \Sigma$
$\mathscr{Z}(\sigma)$; $\mathscr{L}(\sigma)$	Sets of satellites and legs of service $\sigma \in \Sigma$
$t(\sigma)$	Departure time of service $\sigma \in \Sigma$ from its origin
$c(\sigma)$	Cost (fixed) of service $\sigma \in \Sigma$
$\mathscr{H} = \{h\}$	Set of city-freighter work assignments
$\mathscr{W}(h)$	Set of work segments of city-freighter work assignment $h \in \mathscr{H}$
$c(h)$	Cost of city-freighter work assignment $h \in \mathscr{H}$
$\mathscr{I}(k) = \{i\}$	Set of itineraries, customer demand $k \in \mathscr{K}$

space (for, e.g., pallets or assemblies of smart π-containers), for as long as each compartment so defined may be accessed independently of the others. Moreover, the pseudo-backhaul policy assumed for loading and unloading vehicles implies that one can start loading outbound demand in a compartment only once all inbound freight present in the compartment has been unloaded. Such a policy facilitates streamlining operations at satellites, which is beneficial when capacities are tight as in City Logistics. We introduce compartments into the service definition through the set of *compartment services* $\mathscr{B}(\sigma)$ of service σ. To simplify the presentation, we assume all compartments of a given urban-vehicle type τ have the same capacity u_τ^{B}. Obviously, all compartment services are selected when the corresponding service is selected, and $|\mathscr{B}(\sigma)| = 1$ for single-compartment services.

Freight itineraries in the present context of approximated second-tier routing may be defined straightforwardly based on the service used. To illustrate, consider the itinerary of an inbound demand. It starts at the external-zone of origin from where the goods are received at the selected CDC, where the goods are loaded into an urban-vehicle (and compartment) of the selected service. The goods are then transported to the selected satellite by the selected service (with possibly intermediary stops but no work on the goods considered here), from where they are to be delivered to the final customer zone. It is noteworthy that the selection of a service (and compartment) provides all the necessary decision information, i.e., the CDC, the satellite, and the relevant time stamps, i.e., departure from CDC and arrival to satellite for delivery to the customer zone that, implicitly, takes care of the synchronization issue. Let then $c(k, b, \sigma, z)$ be the unit cost of moving freight of demand $k \in \mathscr{K}$ in compartment $b \in \mathscr{B}(\sigma)$ of service $\sigma \in \Sigma$ among its external and customer zones through satellite $z \in \mathscr{Z}(k)$.

One may therefore write an arc-based SSND formulation based on service and compartment selections and assignments to demands. The decisions variables are:

- $y(\sigma) = 1$, if the urban-vehicle service $\sigma \in \Sigma$ is selected, 0, otherwise;
- $x(b, \sigma, z, k) = 1$, if demand $k \in \mathscr{K}$ is assigned to compartment service $b \in \mathscr{B}(\sigma)$ and satellite $z \in \mathscr{Z}(k)$ (visited by service $\sigma \in \Sigma$), 0 otherwise.

The arc-based formulation of the SSND problem then becomes

$$\text{Minimize} \quad \sum_{\sigma \in \Sigma} c(\sigma) y(\sigma) \tag{16.14}$$

$$+ \sum_{k \in \mathscr{K}} \sum_{\sigma \in \Sigma} \sum_{b \in \mathscr{B}(\sigma)} \sum_{z \in \mathscr{Z}(k)} \left(c(k, b, \sigma, z) + c(k, z, \sigma)\right) d(k) x(b, \sigma, z, k)$$

$$\text{Subject to} \quad \sum_{\sigma \in \Sigma} \sum_{b \in \mathscr{B}(\sigma)} \sum_{z \in \mathscr{Z}(k)} x(b, \sigma, z, k) = 1, \; k \in \mathscr{K}, \tag{16.15}$$

$$x(b, \sigma, z_1, k_1) + x(b, \sigma, z_2, k_2) \leq 1, \; b \in \mathscr{B}(\sigma), \; \sigma \in \Sigma, \; k_1 \in \mathscr{K}^I,$$
$$k_2 \in \mathscr{K}^O, z_1, z_2 \in \mathscr{Z}(k), \; z_1 \geq z_2, \tag{16.16}$$

$$\sum_{k \in \mathscr{K}^I} \sum_{z \in \mathscr{Z}(k)} d(k) x(b, \sigma, z, k) \leq u_\tau^{\mathrm{B}}(\sigma) y(\sigma), \; \sigma \in \Sigma, \tag{16.17}$$

$$\sum_{k \in \mathscr{K}^O} \sum_{z \in \mathscr{Z}(k)} d(k) x(b, \sigma, z, k) \leq u_\tau^{\mathrm{B}}(\sigma) y(\sigma), \; \sigma \in \Sigma, \tag{16.18}$$

$$\sum_{t=1,\ldots,T} \sum_{\sigma \in \Sigma(f,t,\tau)} y(\sigma) \leq n_{f\tau}, \; f \in \mathscr{F}, \; \tau \in \mathscr{T}_m, \; m \in \mathscr{M}, \tag{16.19}$$

$$\sum_{\sigma \in \Sigma(z,t)} y(\sigma) \leq u_z^{\mathrm{T}},\ z \in \mathscr{Z},\ t = 1, \ldots, T, \tag{16.20}$$

$$\sum_{\sigma \in \Sigma(z,t,\tau)} y(\sigma) \leq u_{z\tau}^{\mathrm{T}},\ z \in \mathscr{Z},\ \tau \in \mathscr{T}_m,\ m \in \mathscr{M},\ t = 1, \ldots, T, \tag{16.21}$$

$$\sum_{\sigma \in \Sigma(z,t)} \sum_{k \in \mathscr{K}} d(k)x(b, \sigma, z, k) \leq u_z^{\mathrm{K}},\ z \in \mathscr{Z}, t = 1, \ldots, T, \tag{16.22}$$

$$y(\sigma) \in \{0, 1\},\ \sigma \in \Sigma \tag{16.23}$$

$$x(b, \sigma, z, k) \in \{0, 1\},\ k \in \mathscr{K},\ \sigma \in \Sigma,\ z \in \mathscr{Z}(k) \tag{16.24}$$

The objective function (16.14) minimizes the total generalized cost of selecting and operating services that move inbound and outbound demand flows, distributing demands from satellites and bringing outbound demands to satellites, as well as selecting a CDC for each demand.

Constraints (16.15) ensure that each item is assigned exactly to one compartment, while constraints (16.16) ensure that outbound demand is only assigned to a compartment after the inbound demand is unloaded and the compartment is empty. The compartment capacities for inbound and outbound traffic are enforced by the linking constraints (16.17) and (16.18). These constraints combined with constraints (16.16) enforce the capacity restriction for the entire service. Then, constraints (16.19) ensure that the maximum number of vehicles of each type assigned to a city distribution center is never exceeded. Constraints (16.20) and (16.21) limit the number of urban vehicles present at a satellite at each period in total and per transportation mode, respectively. Finally, constraints (16.22) limit the amount of demand that can be unloaded or loaded at a satellite at each period.

4.2 Demand Uncertainty in Tactical Planning for City Logistics

We now present a stochastic scheduled service network design formulation for the tactical planning of two-tier City Logistics systems when the uncertainty on demand is explicitly taken into account. As discussed above, such formulations are required when major resources must be allocated and their utilization must be planned for the length of the planning horizon, well before the actual operations take place, while simultaneously acknowledging the strategies that are used during operations to adjust the plan to the observed demand. The tactical plan, which is built prior to the beginning of the season, then aims to determine the main structure of the service network and major resource allocation that will be executed regularly at each period of the planning horizon, but without fixing all the operational details that will be address at execution time. The goal is to optimize the overall cost (operations and environmental impact) of the system in terms of service selection and resource allocation plus an estimation of the costs involved in adjusting the plan and operating accordingly over the contemplated planning horizon. One thus expects that the explicit flexibility introduced into the tactical plan translates into the capability of the selected service network to accommodate a certain range of demand variability with no or little modification to the activities of the major resources involved. Such a priori optimization approaches are generally addressed through two-stage stochastic programming formulations with recourse, the latter corresponding to the strategies used to adjust the plan to given realizations of demand (Chap. 10).

In the two-stage stochastic SSND formulation with recourse described in this section, the plan is the object of the first stage and it concerns the selection of the first-tier services and schedules together with the associated demand itineraries between external zones and satellites. The later implies the allocation of customers to particular (satellite, period) combinations, the so-called *rendez-vous points*, providing strong indications on the satellite workloads and the dimensions of the city-freighter fleets required. City-freighter routing decisions are to be taken at each period the system operates, once the actual demand has been observed. They are thus the object of the second stage, and only an approximation of the corresponding costs and operations is integrated in the first-stage formulation. In this sense, the first-stage *urban-vehicle service network design* model is quite similar to the deterministic SSND model of Sect. 4.1. The output of this model includes the selection of services, customer-demand itineraries down to the (satellite, period) rendez-vous points (including the type of city freighter), and the customer-satellite allocations. It is this information that guides and constrains the second-stage recourse strategies, which address the city-freighter routing, determine the extra service capacity required, if any, and, eventually, slightly modify the service network.

We focus in this chapter on strategies that allow slight adjustments to the service network selected in the first stage and which, consequently, involve network-design formulations. Such a strategy aims to increase service flexibility, by fixing the "backbone" of the urban-vehicle service design identified by the first stage, while permitting to modify the corresponding urban-vehicle departure times. This may be viewed as a combined dispatching-routing decision to let the urban vehicles leave somewhat earlier or later than the planed schedule while determining the routing of the "regular" and "extra" city freighters. The presentation of the stochastic SSND model follows the deterministic path-based framework introduced in Sect. 3.

Let $\Omega = \{\omega\}$ be the sample space of the random event. Let $\mathscr{K}(\omega) = \{k(\omega)\}$ be the set of customer-demand realizations (0 demand corresponds to the no-demand case) for $\omega \in \Omega$, $\mathsf{K} = \{\mathscr{K}(\omega)|\omega \in \Omega\}$, and $d(k, \omega)$ the volume associated with demand k given $\omega \in \Omega$. Let $\widehat{d(k)}$ be the point forecast of the volume of customer demand $k \in \mathscr{K}$ used in the first stage of the formulation. Usually, $\widehat{d(k)}$ corresponds to the "best" estimate used in deterministic SSND, representing the "regular" demand one expects to see on a "normal" day (e.g., 80% of the maximum expected demand; no particular value is assumed for the formulation). One then desires the planned resource allocation and scheduling to be able to address this demand or, in other words, to provide at least this level of service in all cases.

The definition of the service network does not change but, as extra city freighters may be required and city-freighter routing is part of the second stage, reduced first-stage demand itineraries, $\mathscr{I}^{\mathrm{R}}(k)$, have to be defined. These include the itineraries corresponding to the possibility of CDC$\leftrightarrow$customer services through extra city-freighters ("artificial" arcs). Then, as the first-stage problem considers an approximation of routing activities and costs only, through customer-to-satellites allocations (for each city freighter type, mode, and period), regular first-stage demand itineraries include satellite$\leftrightarrow$customer links only, rather than actual city-freighter work segments. Let $\tilde{c}(k, i)$ be the cost of itinerary $i \in \mathscr{I}^{\mathrm{R}}(k)$ corresponding to these decisions, i.e., the extra city freighter usage or satellite$\leftrightarrow$customer allocation, respectively (the cost of services is captured in the corresponding term of the objective function). Similarly to the definition of all other costs, $\tilde{c}(k, i)$ reflects the handling and moving freight, as well as a measure of the "nuisance" factors related to the presence of city freighters in the city at the particular time of the delivery.

Two sets of decision variables are defined to select urban-vehicle services and first-stage demand itineraries:

$y(\sigma) = 1$, if urban-vehicle service $\sigma \in \Sigma$ is selected, 0, otherwise;
$\xi(i) = 1$, if itinerary $i \in \mathscr{I}^{\mathrm{R}}(k)$ of demand k is used, 0, otherwise.

An a priori plan then specifies:

- A set of urban-vehicle services $\Sigma(\boldsymbol{y}^1)$ to be operated, and a set of first-stage itineraries $\mathscr{I}^{\mathrm{R}}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1) = \{i(\boldsymbol{y}^1, \boldsymbol{\zeta}^1, k),\ k \in \mathscr{K}\}$ bringing the load of each customer demand k in time to its appointed satellite;
- A set of active *rendez-vous* points (satellite, period), $(z, t) \in \mathscr{R}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1)$, where urban vehicles and city freighters meet and freight is transferred; Customer-to-satellite assignments are associated to each rendez-vous point.

The notation $(\boldsymbol{y}^1, \boldsymbol{\zeta}^1)$ used in this section emphasizes that the second stage optimization problem is constrained by the decisions of the first stage. The *Dispatch and Route* recourse strategy assumes $\boldsymbol{y}^1, \boldsymbol{\zeta}^1$ fixed, focusing on possibly changing the departure times of the selected services, as well as on optimizing the second-tier routing to service inbound and outbound customer demands.

Let $[a(zt),\ b(zt)]$ be the time window of satellite z at rendez-vous point (z, t), such that customers serviced by vehicles starting/finishing their routes within the interval will be serviced (delivery or pick up) on time. Let $[a(\sigma), b(\sigma)]$ be the *opportunity time window* around the departure time of service $\sigma \in \Sigma(\boldsymbol{y}^1)$, derived from the satellite time windows and the travel times between satellites and CDCs. Let $\Sigma(\boldsymbol{y}^1, \sigma)$ be the set of possible departures in the opportunity window of service $\sigma \in \Sigma(\boldsymbol{y}^1)$ among which one selects to instantiate the plan to the observed demand.

The movements of freight by city freighters at the second tier of the system is restricted to the service network and satellite utilization specified in the a priori plan. Given the possibility of extra city freighters visiting CDCs to move excess demand, the definition of a work segment is enlarged to encompass CDCs as the first or last stop on its route. The notation of the city-freighter work assignments and work segments then becomes $\mathscr{H}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1)$ and $\mathscr{W}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1, h)$, respectively. Similarly, the demand itineraries restricted to the first-stage service design, and the possibility of CDC-related work segments, are indicated by $i \in \mathscr{I}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1, k(\omega)) \subset \mathscr{I}(k)$.

The decision variables defined for the second stage then are:

$y^2(k(\omega), \sigma^2) = 1$, if urban-vehicle service $\sigma \in \Sigma(\boldsymbol{y}^1)$ is selected, 0, otherwise;
$\varphi^2(k(\omega), h) = 1$, if work assignment $h \in \mathscr{H}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1)$ is selected, 0, otherwise;
$\zeta^2(k(\omega), i) = 1$, if itinerary $i \in \mathscr{I}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1, k(\omega))$ of demand $k(\omega) \in \mathscr{K}(\omega)$ is selected, 0, otherwise.

The two-stage SSND formulation minimizing the expected generalized cost of the system over the planning horizon may then be written as

$$\text{Minimize} \quad \sum_{\sigma \in \Sigma} c(\sigma) y(\sigma) \tag{16.25}$$

$$+ \sum_{k \in \mathscr{K}} \sum_{i \in \mathscr{I}^{\mathrm{R}}(k)} \tilde{c}(k,i) \widehat{d(k)} \xi(i) + \mathrm{E}_{\mathsf{K}} \left[Q^{RP}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1, \mathscr{K}(\omega)) \right]$$

$$\text{Subject to} \quad \sum_{i \in \mathscr{I}^{\mathrm{R}}(k)} \xi(i) = 1, \; k \in \mathscr{K}, \tag{16.26}$$

$$\sum_{k \in \mathscr{K}(\sigma l)} \sum_{i \in \mathscr{I}^{\mathrm{R}}(k) | \sigma(i) = \sigma} \widehat{d(k)} \xi(i) \le u_{\tau(\sigma)} y(\sigma), \; l \in \mathscr{L}(\sigma), \sigma \in \Sigma, \tag{16.27}$$

$$\sum_{\sigma \in \Sigma(z,t)} y(\sigma) \le u_z^{\mathrm{T}}, \; z \in \mathscr{Z}, \; t = 1, \ldots, T, \tag{16.28}$$

$$\sum_{\sigma \in \Sigma(z,t)} y(\sigma) \le u_{z\tau}^{\mathrm{T}}, \; z \in \mathscr{Z}, \; \tau \in \mathscr{T}_m, \; m \in \mathscr{M}, \; t = 1, \ldots, T, \tag{16.29}$$

$$\sum_{v \in \mathscr{V}} \left[\sum_{k \in \mathscr{K}} \sum_{i \in \mathscr{I}^{\mathrm{R}}(k,v,t)} \widehat{d(k)} \xi(i) \right] / u_v \le u_z^{\mathrm{V}} \quad z \in \mathscr{Z}, \; t = 1, \ldots, T, \tag{16.30}$$

$$\sum_{v \in \mathscr{V}} \left[\sum_{k \in \mathscr{K}} \sum_{i \in \mathscr{I}^{\mathrm{R}}(k,v,t)} \widehat{d(k)} \xi(i) \right] \le u_z^{\mathrm{K}}, \; z \in \mathscr{Z}, \; t = 1, \ldots, T, \tag{16.31}$$

$$\sum_{\sigma \in \Sigma(f,t,\tau)} y(\sigma) \le n_{f\tau}, \; f \in \mathscr{F}, \; \tau \in \mathscr{T}_m, \; m \in \mathscr{M}, \; t = 1, \ldots, T, \tag{16.32}$$

$$y(\sigma) \in \{0, 1\}, \; \sigma \in \Sigma, \tag{16.33}$$

$$\xi(i) \in \{0, 1\} \quad i \in \mathscr{I}^{\mathrm{R}}(k), \; k \in \mathscr{K}, \tag{16.34}$$

with

$$Q^{RP}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1, \mathscr{K}(\omega)) = \tag{16.35}$$

$$\text{Minimize} \sum_{\sigma \in \Sigma(\boldsymbol{y}^1)} c(\sigma) \sum_{\sigma^2 \in \Sigma(\boldsymbol{y}^1,\sigma)} y^2(k(\omega), \sigma^2) + \sum_{h \in \mathscr{H}(\boldsymbol{y}^1,\boldsymbol{\zeta}^1)} c(h)\varphi^2(k(\omega), h)$$

$$\text{Subject to} \sum_{i \in \mathscr{I}(\boldsymbol{y}^1,\boldsymbol{\zeta}^1,k(\omega))} \zeta^2(k(\omega), i) = 1,\ k(\omega) \in \mathscr{K}(\omega), \tag{16.36}$$

$$\sum_{\sigma^2 \in \Sigma(\boldsymbol{y}^1,\sigma)} y^2(k(\omega), \sigma^2) = 1,\ \sigma \in \Sigma(\boldsymbol{y}^1), \tag{16.37}$$

$$\sum_{k(\omega) \in \mathscr{K}(\omega)} \sum_{i \in \mathscr{I}(\boldsymbol{y}^1,\boldsymbol{\zeta}^1,k(\omega),\sigma)} d(k, \omega)\zeta^2(k(\omega), i) \leq u_\tau y^2(k(\omega), \sigma^2), \tag{16.38}$$

$$\sigma^2 \in \Sigma(\boldsymbol{y}^1, \sigma), \sigma \in \Sigma(\boldsymbol{y}^1),$$

$$\sum_{k(\omega) \in \mathscr{K}(w(h)t)} \sum_{i \in \mathscr{I}(\boldsymbol{y}^1,\boldsymbol{\zeta}^1,k(\omega))|h(i)=h} d(k, \omega)\zeta^2(k(\omega), i) \leq u_v \varphi^2(k(\omega), h) \tag{16.39}$$

$$w \in \mathscr{W}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1, h),\ h \in \mathscr{H}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1),\ t = 1, \ldots, T,$$

$$\sum_{\sigma \in \Sigma(\boldsymbol{y}^1)} \sum_{\sigma^2 \in \Sigma(\boldsymbol{y}^1,\sigma,z,t)} y^2(k(\omega), \sigma^2) \leq u_z^{\mathrm{T}}, \tag{16.40}$$

$$(z, t) \in \mathscr{R}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1),\ z \in \mathscr{Z},\ t = 1, \ldots, T,$$

$$\sum_{\sigma \in \Sigma(\boldsymbol{y}^1)|\tau(\sigma)=\tau} \sum_{\sigma^2 \in \Sigma(\boldsymbol{y}^1,\sigma,z,t)} y^2(k(\omega), \sigma^2) \leq u_{z\tau}^{\mathrm{T}}, \tag{16.41}$$

$$(z, t) \in \mathscr{R}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1),\ z \in \mathscr{Z},\ \tau \in \mathscr{T}_m,\ m \in \mathscr{M},\ t = 1, \ldots, T,$$

$$\sum_{h \in \mathscr{H}(\boldsymbol{y}^1,\boldsymbol{\zeta}^1)} \varphi^2(k(\omega), h) \leq u_z^{\mathrm{V}},\ (z, t) \in \mathscr{R}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1),\ z \in \mathscr{Z},\ t = 1, \ldots, T, \tag{16.42}$$

$$\sum_{i \in \mathscr{I}(\boldsymbol{y}^1,\boldsymbol{\zeta}^1,k(\omega))} d(k)\xi(i) \leq u_z^{\mathrm{K}},\ z \in \mathscr{Z},\ t = 1, \ldots, T, \tag{16.43}$$

$$\sum_{h \in \mathscr{H}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1)} \varphi^2(k(\omega), h) \le n_\nu^+, \ \nu \in \mathscr{V}_m, \ m \in \mathscr{M}, \tag{16.44}$$

$$y^2(k(\omega), \sigma^2) \in \{0, 1\}, \ \sigma \in \Sigma(\boldsymbol{y}^1), \tag{16.45}$$

$$\varphi^2(k(\omega), h) \in \{0, 1\}, \ h \in \mathscr{H}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1), \tag{16.46}$$

$$\zeta^2(k(\omega), i) \in \{0, 1\}, \ i \in \mathscr{I}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1, k(\omega)), \ k(\omega) \in \mathscr{K}(\omega). \tag{16.47}$$

The first-stage objective function (16.25) minimizes the total generalized cost computed as the cost of selecting and operating urban-vehicle services to move the point-forecast demand between external zones and satellites, plus the approximated cost of assigning customers to satellites and moving demand between them, plus the expected cost of the recourse over the planning period, i.e., the cost of operating the system according to the a priori plan $x(\boldsymbol{y}^1, \boldsymbol{\zeta}^1)$ adjusted, *instantiated*, for the realized demand $\mathscr{K}(\omega)$ by applying the recourse policy *RP* with cost $Q^{RP}(x(\boldsymbol{y}^1, \boldsymbol{\zeta}^1), \mathscr{K}(\omega))$.

The first-stage constraints (16.26)–(16.34) correspond to constraints (16.2), (16.3), (16.5)–(16.9), (16.11), and (16.13), where the numbers of second-tier city freighters in constraints (16.30) are derived from the total volume of demand leaving the satellite at period t on city freighters of all types, where $\mathscr{I}^{\mathrm{R}}(k, \nu, t)$ stands for the corresponding itineraries of demand k. (Constraints (16.4), (16.10), (16.12) are not needed in the first-stage formulation.)

The objective function of the second-stage formulation (16.35) computes the total cost of operating the selected urban-vehicle services, with possibly new departure times, and the normal and extra city-freighter work assignments. Equations (16.36) indicate that each demand must be satisfied by a single itinerary, while constraints (16.37) state that exactly one departure must be selected for each opportunity window. Constraints (16.38) enforce the urban-vehicle capacity restrictions, where $\mathscr{I}(\boldsymbol{y}^1, \boldsymbol{\zeta}^1, k(\omega), \sigma)$ stands for the set of itineraries of customer demand k being moved on service σ. Constraints (16.39) enforce the capacity of city freighters on all work segments of both regular and extra city-freighter work assignments. Constraints (16.40) and (16.41) enforce the satellite capacity restrictions in terms of urban vehicles, total and by mode, respectively. The satellite capacity in terms of city freighters and freight volume are enforced by constraints (16.42) and (16.43), respectively. Constraints (16.44) limit the number city freighters (work assignments) of each type used to the available fleet augmented with the cardinality of the extra fleet. Finally, constraints (16.45)–(16.47) define the feasible region of the second-stage formulation.

4.3 Designing the City Logistics Network: Strategic Planning

The previous SSND formulations yield “best” (optimal, when solved exactly) tactical plans and two-tier scheduled service networks in terms of the generalized cost of the system given the forecast demand, the 2T-CL system layout, composition, and attributes, as well as current regulations and operation policies. The models may be used not only for drawing tactical plans, however, but also as an analysis and evaluation tool for a broad range of longer-term issues (Sect. 2.2). Scenarios related to the structure, environment or policies of a (two-tier) City Logistics system may be evaluated using the model framework described in this chapter appropriate for the case. To illustrate, the complete formulations (Sects. 3 and 4.2) are appropriate to evaluate the impact on system performance of vehicle fleet deployment including number and dimension of fleets at both tiers, assignment to CDCs or satellites, vehicle characteristics, etc. On the other hand, models approximating the second-tier routing (Sects. 4.1 and 4.2 with approximated routing cost) appear more appropriate to evaluate longer-term scenarios implying a certain level of imprecision in the demand data, such as different system layouts in terms of number, location, and capacity of facilities.

The SSND models may also be used to study different cooperation rules when several service providers (public and private carriers, terminals, etc.) share infrastructure and jointly perform the City Logistics activities under joint planning and operations-management mechanism (e.g., an arm-length information-sharing and decision platform). Such rules may specify, e.g., how costs or vehicle utilization, or both have to be distributed among the participants to reflect their contractual commitment (in terms of type and size of the fleet, territory covered, financial and risk share, etc.). Not much research has been dedicated yet on how to extend SSND models to account for these issues. To illustrate a first step in this direction, consider a group $\mathscr{C}$ of first-tier collaborating carriers, each with a weight ()“share”) $\alpha_c, c \in \mathscr{C}$, representing the target for its level of activity or cost it incurs. Let $[\alpha_c^-, \alpha_c^+]$ represent the interval of variation acceptable for the next planning cycle, and $c(\sigma, c)$ the cost of service $\sigma \in \Sigma$ performed by carrier $c \in \mathscr{C}$. Defining $y(\sigma, c) = 1$, if the urban-vehicle service $\sigma \in \Sigma$ is selected to be operated by carrier $c \in \mathscr{C}$, and 0, otherwise, the total service-selection cost in the SSND objective function becomes

$$\sum_{\sigma \in \Sigma} \sum_{c \in \mathscr{C}} c(\sigma, c) y(\sigma, c), \tag{16.48}$$

the share of that cost borne by each carrier in the coalition being controlled by

$$\alpha_c^- \sum_{\sigma \in \Sigma} \sum_{c \in \mathscr{C}} c(\sigma, c) y(\sigma, c) \leq \sum_{\sigma \in \Sigma} c(\sigma, c) y(\sigma, c) \leq \alpha_c^+ \sum_{\sigma \in \Sigma} \sum_{c \in \mathscr{C}} c(\sigma, c) y(\sigma, c) \tag{16.49}$$

Similar constraints may be added on the operations performed to limit the resource utilization (measured, e.g., in time, km, or weight-km) by the respective carriers. Penalties may also be added to the objective function to add flexibility to the solution methods. This constitutes an interesting research avenue.

The SSND models may also be generalized for a direct utilization as decision-support tools for strategic decisions such as the location/selection of facilities, CDCs and satellites, the design of freight-dedicated corridors throughout the city (particularly between external zones and CDCs and between the latter and the satellites) or the introduction of new/upgraded infrastructure and services. Continuing the previous discussion, he long-term nature of strategic decisions suggest that second-tier routing should be approximated in such "strategic" SSND models, which would include design decision variables on the selection of the contemplated facilities or infrastructure structures, as well as the corresponding commodity-flow variables and the capacity and linking constraints. Budget constraints come naturally to mind as well.

To illustrate, consider the 2T-CL case where one aims to select satellites given selection costs and budget. These costs could represent the renting or utilization of facilities for the next planning period, which corresponds to a tactical planning decision, or actual longer-term strategic acquisition or securing costs. Let $c(z), z \in \mathscr{Z}$, the selection cost of satellite $z \in \mathscr{Z}$ and B the total budget available for satellite selection and utilization. The medium-term SSND formulation of Sect. 4.1 may then be extended with the additional decision variables $\zeta(z) = 1$, is satellite $z \in \mathscr{Z}$ is selected, and 0, otherwise, and the corresponding total selection cost $\sum_{z \in \mathscr{Z}} c(z)\zeta(z)$ added to the objective function (16.14). The right hand side of constraints (16.20)–(16.22) is multiplied by the satellite-selection variable (e.g., $u^{\mathrm{T}}_{z\tau}\zeta(z)$ for the latter constraint) to transform them from capacity to linking constraints. The flow-related linking constraints (16.50) and budget constraints (16.51) must also be added.

$$x(b, \sigma, z, k) \leq \zeta(z)\zeta(z), \; k \in \mathscr{K}, \; \sigma \in \Sigma, \; z \in \mathscr{Z}(k) \tag{16.50}$$

$$\sum_{z \in \mathscr{Z}} c(z)\zeta(z) \leq B \tag{16.51}$$

5 Bibliographical Notes

We aim with this section to give a brief historical survey of and appropriate credit for the developments related to network design and the planning of City Logistics, two-tier systems in particular. General descriptive papers and surveys regarding City Logistics and related operations research developments may be found in, e.g., Taniguchi et al. (2001); Benjelloun and Crainic (2009); Gonzalez-Feliu et al. (2014); Taniguchi (2014); Cattaruzza et al. (2017); Crainic (2008); Bektaş et al. (2017);

Savelsbergh and Van Woensel (2016), while Holguín-Veras et al. (2020a,b) review CL initiatives from the point of view of urban freight management.

Most contributions in the literature addressing the design of City Logistics systems focus on the location of facilities only and proposed location-routing models (mentioned bellow). A few different approaches were proposed. Taniguchi et al. (1999) combines a facility-selection-dimensioning model and a nonlinear traffic-equilibrium formulation to reflect the user (truck drivers) choices; Crainic et al. (2004) introduced a general two-tier city logistics system concept (see also Gragnani et al. 2004), together with a location-allocation methodology for the strategic decision issue of determining the satellite structure of the system. The authors consider inbound demand and single motor-carrier modes and fleets for each of the two tiers of the system; Baldi et al. (2012) address the same problem setting with stochastic costs for the paths from CDC to satellite to customer zone; Gianessi et al. (2016, see also Gianessi (2014)) propose a location-routing formulation integrating decisions related to the design of a ring structure connecting the CDCs, inbound and outbound demands, and routing of inter-CDC flows onto the ring structure to avoid excessive travel through the city and on the highways around; Guerrero-Lorente et al. (2020) propose a location-routing model for a Cl postal network with approximated routing costs; Hu et al. (2020) present a bi-objective location-allocation model to study an underground 2T-CL system for in Beijing, China.

Crainic et al. (2009) proposed the first modeling framework for the short to medium-term planning of inbound-demand, single-mode 2T-CL systems with the possibility of different vehicle types and specific product-to-vehicle assignment rules. The authors also introduced to the City Logistics literature the time-dependency of demand and the corresponding issue of scheduling and synchronizing first-tier urban-vehicle services and second-tier city-freighter multi-tour work assignments. They proposed a path-based formulation for the day-before planning problem, when planning is performed shortly ("the day before") before operations when the demand is known. The SSND modeling framework of Sect. 3 starts from this work.

Crainic et al. (2009) also introduced formulations for the first-tier scheduled service network design and the second-tier synchronized, scheduled, non-substituable origin-destination (OD) demand, multi-depot, multi-tour, heterogeneous vehicle routing problem with time windows (*SSOD-MDMTH-VRPTW*) problems (see, e.g., Nguyen et al. 2013; Crainic et al. 2016b; Nguyen et al. 2017; Bettinelli et al. 2019, for developments on the latter problem). The paper discusses solution-method avenues for all formulations, introducing a meta-heuristic structure for the full model based on decomposing it along tiers, without actually solving the problem.

Two additional observations were made in Crainic et al. (2009). First, that one could view the service selection on the first tier as a particular case of a vehicle routing problem, yielding a two-tier, or, in VRP terminology, a two-echelon SSOD-MDMTH-VRPTW. Several publications followed focusing on two-echelon vehicle routing (e.g., Perboli et al. 2011; Hemmelmayr et al. 2012; Contardo et al. 2012; Mancini et al. 2014; Masson et al. 2017; Grangier et al. 2015; Breunig et al.

2016, 2019; Dellaert et al. 2019) and location-routing problems with facilities being selected on a single tier (e.g., Guyon et al. 2012; Gianessi et al. 2016; Winkenbach et al. 2016; Boccia et al. 2017) or, more rarely, on both tiers (e.g., Boccia et al. 2010, 2011). It is noteworthy that most of these contributions do not include the full range of attributes defined above; thus, OD demand, multiple demand types, time-dependency, multiple tours, and synchronization are quite often missing..

The second observation of Crainic et al. (2009) is that for longer-term planning, e.g., mid-term tactical planning and the evaluation of long-term strategic alternatives, the second-tier routing problem could be approximated and added to the first-tier formulation through appropriately-defined service costs on links connecting satellites and customer-zone. Crainic and Sgalambro (2014) adopted this idea and focused on the modeling of the first-tier service network design within the day-before planning problem, studying the impact of a number of system parameters on the final design. This line of work was significantly extended by Fontaine et al. (2021) while addressing a richer problem setting than in previous literature.

Fontaine et al. (2021) address a 2T-CL setting which integrates inbound and outbound demands (see Crainic et al. 2012, for an initial discussion on the impact of integrating several types of demand into City Logistics planning), decisions on the assignment of customers to consolidation distribution centers and satellites, multiple satellite capacity measures in terms of freight volume and numbers of first and second-tier vehicles, several heterogeneous limited-size fleets of particular transportation modes. The intermodal transportation aspect brings together traditional, road-based, carriers and massive-flow carriers and vehicles captive of their routes or infrastructure, e.g., buses, trolleybuses, tramways, subways, and regular rail bringing flows from CDCs to satellites located within downtown train stations (Trentini and Mahléné 2010; Freemark 2011; Riemann 2019; Lindholm and Behrends 2012; Masson et al. 2017). As several vehicle types have more than one cargo-holding space, as illustrated by the multiple cargo bays of several proposed cargo tramways and the (vertical or horizontal) separators that may be used within trucks, the authors also introduced multi-compartment vehicles. Fontaine et al. (2021) propose an arc-based SSND formulation for the tactical planning of such extended systems, where the customer service cost through pickup or delivery routing is approximated on corresponding customer-satellite arcs. They also proposed an efficient Benders decomposition algorithm (Crainic et al. 2021), which includes specialized valid inequalities and an innovative partial decomposition strategy based on the use of aggregation techniques for deterministic problems. The numerical results show the efficiency of the proposed algorithm compared to a well-known commercial solver, as well as the benefits of considering several transportation modes and demand types. The model of Sect. 3 includes a number of elements of this work.

One finds very few contributions addressing the uncertainty inherent to the City Logistics system. To the best of our knowledge, travel and service time uncertainty has not been addressed in the context of optimization for city logistics planning, although the issue is mentioned by several authors. A limited number of contributions addressed time-dependent travel times (e.g., Liu et al. 2020) but focused on routing issues only and no uncertainty was considered. Contributions were made with respect to developing appropriate data for these problems (e.g. Ehmke et al. 2012; Maggioni et al. 2014), but were not included into tactical planning models.

With respect to demand uncertainty, Crainic et al. (2016a) propose a two-stage stochastic-programming formulation for a 2T-CL tactical planning model aimed at the system setting defined in Crainic et al. (2009). The first stage corresponds to the selection of the first-tier services and the determination of the partial demand itineraries up to the selected satellites, as well as the satellite utilization in terms of customer assignments. The second stage models the selection of ad-hoc additional city freighters and the routing-based strategy to adjust the plan once demand is realized. The authors used the meta-heuristic of Crainic et al. (2009) as the basis of a Monte-Carlo evaluation procedure of several recourse strategies with increasing degrees of flexibility in routing and customer assignments. Not surprisingly, increased flexibility in resource allocation and system management displayed the best performances. This is an encouraging first step in what constitutes a major research issue.

Another rich research challenge corresponds to address the case multiple organizations, public and private carriers and other service providers operating in (parts of) cities under some form of cooperation/coalition and resource sharing agreements (e.g., Morana et al. 2014). A few policy and simulation-based studies may be found in the literature, but we know of only one paper at this time representing some of these rules within a tactical planning model (Crainic et al. 2020, used in the discussion of Section 4.3). The authors introduced penalty costs and constraints limiting the utilization of certain resources according to the shares of participating carriers into a simplified version of the Fontaine et al. (2021) SSND formulation. Experimental results show the impact of such conditions, not only on the computational burden, but also on the distribution of resources and the performance of the system.

6 Conclusions and Perspectives

The transportation and logistics industry is continuously evolving with the evolution of society, politics, and technology. City Logistics constitutes one of the paths of this evolution, echoing and interacting with similar developments in new business and organizational models such as Physical Internet and synchromodality. Operation Research and analytics accompany and sometimes precede this evolution by providing methods and decision-support instruments for the analysis, planning and management of transportation and logistics systems, from potentiality to deployment and operations or abandon. The impact is a two-way street, however. as changes in social and industrial behavior and in technology challenge the field and spur modeling and algorithmic development.

This chapter presented the main outcomes of this interaction from the point of view of network design. The formulations target tactical and strategic planning-level issues and reflect the novel characteristics proper to City Logistics, in particular, several layers (tiers) of facilities and operations, multimodality, multiple heterogeneous fleets, collaboration of private and public service providers, time-dependency of demand and operations, interaction of somewhat regular transport services and local pickup and delivery activities, and synchronization of fleet operations on the two tiers at the intermediary facilities, to name but a few.

Many challenges are still facing the field and new ones are emerging, yielding a rich set of exciting research avenues for Operations Research and Transportation Science. We conclude the chapter briefly discussing a number of those, focusing on modeling and algorithmic challenges from a network-design perspective.

Modeling issues are many and of the utmost interest. We already mentioned in previous sections a number of important and challenging research directions not only for City Logistics but also for (service) network design in general: (1) the integration into SSND formulations inbound, outbound, and local demand into a seamless and comprehensive operation, which requires integrating and synchronizing SSND and time- and OD-dependent multi-tour heterogeneous pickup and delivery routing; (2) the development of SSND methodology for various cases of collaborating and capacity-sharing stakeholder consortia, operating a unified system of private and public resources under various profit-cost-risk-resource utilization sharing rules; (3) the explicit representation and integration of the uncertainty regarding the demand as well as travel and service times at facilities and customer locations.

It is noteworthy that methodological and application challenges emerge, on the one hand, from the particular political, social, and entrepreneurial characteristics of each city and country, which impact directly how technology is accepted and used, how information and traffic flows are ruled, and how people and institutions are allowed to behave. They also follow, on the other hand, from new technology and changing social behavior (see, e.g., Oliveira et al. 2020; Snoeck et al. 2020; Taniguchi et al. 2020). The former includes a very broad range of issues, from the Internet of Things, the Smart City concepts, and the movement towards a numeric and automated transportation and logistics 4.0 industry, to the drones, delivery robots, lockers, and crowd-based logistics which are increasingly part of local delivery. The latter group of issues is equally broad, from the continuously stronger trend of on-line shopping combined to customer requirements for very fast and very cheap delivery, to the increasing variety of time and cost-defined customer-service classes, to revenue management strategies.* How each of these issues is represented individually and when several are jointly present in the problem setting makes for particularly challenging issues. The challenge is increased considering that one needs coherent representations at operational and tactical-strategic levels, and that one aims for objective function and constraint formulations that are amenable to efficient solving.

Worthy of notice is the modeling and algorithmic challenge of combining and synchronizing network design and vehicle routing in an unique formulation. On the one hand, this raises the issue of the adequate modeling of routing into tactic and strategic formulations, and the related efficient and representative approximations of costs and times. On the other hand, one observes that there is not as yet any solution method, neither exact nor meta-heuristic, able to provide consistently high-quality solutions to integrated SSND formulations with synchronization. Moreover, the algorithmic challenge increases significantly with the dimensions of the system in numbers of facilities, customers, fleets, time periods. It is further compounded when uncertainty is explicitly considered.

Research is thus required on high-performance solution methods for deterministic and stochastic network design with explicit or approximated routing models for City Logistics. We mention some very promising avenues, which could, probably should, be combined into efficient solution frameworks: (1) decomposition methods of path and arc-based formulations with novel projections of the synchronization relations on tier-specific subproblems; (2) dynamic generation of first-tier services and second-tier pickup and delivery routes; it is noteworthy that one may address a service route as a vehicle route with particular characteristics and restrictions, opening the way for a unified solution methodology; (3) the dynamic generation of the time-space network delivered very good performances on particular SSND problem settings and is being extended for routing; extending it further to rich SSND with pickup and delivery routing problem settings is particularly fascinating and promising; (4) parallel exact methods and collaborative-search matheuristics for efficiently addressing deterministic and stochastic formulations of realistic dimensions.

References

Baldi, M. M., Ghirardi, M., Perboli, G., & Tadei, R. (2012). The capacitated transshipment location problem under uncertainty: A computational study. *Procedia - Social and Behavioral Sciences, 39*, 424–436.

Bektaş, T., Crainic, T. G., & Van Woensel, T. (2017). From managing urban freight to smart city logistics networks. In K. Gakis, & P. Pardalos (Eds.), *Networks design and optimization for smart cities, series on computers and operations research* (Vol. 8, pp. 143–188). Singapore: World Scientific Publishing.

Benjelloun, A., & Crainic, T. G. (2009). Trends, challenges, and perspectives in city logistics. *Buletinul AGIR, 4*, 45–51.

Bettinelli, A., Cacchiani, V., Crainic, T. G., & Vigo, D. (2019). A branch-and-cut-and-price algorithm for the multi-trip separate pickup and delivery problem with time windows at customers and facilities. *European Journal of Operational Research, 279*, 824–839.

Boccia, M., Crainic, T. G., Sforza, A., & Sterle, C. (2010). A metaheuristic for a two-echelon location-routing problem. In P. Festa (Ed.), *Experimental algorithms, lecture notes in computer science/programming and software* (vol. 6049, pp. 288–301). Berlin: Springer.

Boccia, M., Crainic, T. G., Sforza, A., & Sterle, C. (2011). Location-routing models for designing a two-echelon freight distribution system. Publication CIRRELT-2011-40, Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport, Université de Montréal, Montréal, QC, Canada.

Boccia, M., Crainic, T. G., Sforza, A., & Sterle, C. (2017) Multi-commodity location-routing: flow intercepting formulation and branch-and-cut algorithm. *Computers & Operations Research, 89*, 94–112.

Breunig, U., Schmid, V., Hartl, R. F., & Vidal, T. (2016). A large neighbourhood based heuristic for two-echelon routing problems. *Computers & Operations Research, 76*, 208–225.

Breunig, U., Baldacci, R., Hartl, R. F., & Vidal, T. (2019). The electric two-echelon routing problem. *Computers & Operations Research, 103*, 198–210.

Cattaruzza, D., Absi, N., Feillet, D., & González-Feliu, J. (2017). Vehicle routing problems for city logistics. *EURO Journal on Transportation and Logistics, 6*, 51–79.

Contardo, C., Hemmelmayr, V., & Crainic, T. G. (2012). Lower and upper bounds for the two-echelon capacitated vehicle routing problem. *Computers & Operations Research, 39*(12), 3185–3199.

Crainic, T. G. (2008). City logistics: Advanced urban freight transportation systems. In Z. L. Chen, S. Raghavan (Eds.), *Tutorials in Operations Research 2008, State-of-the-art decision making tools in the information-intensive age*. *Series on computers and operations research* (pp. 181–212). INFORMS.

Crainic, T. G., & Sgalambro, A. (2014). Service network design models for two-tier city logistics. *Optimization Letters, 8*(4), 1375–1387.

Crainic, T. G., Ricciardi, N., & Storchi, G. (2004). Advanced freight transportation systems for congested urban areas. *Transportation Research Part C: Emerging Technologies, 12*(2), 119–137.

Crainic, T. G., Ricciardi, N., & Storchi, G. (2009). Models for evaluating and planning city logistics transportation systems. *Transportation Science, 43*(4), 432–454.

Crainic, T. G., Errico, F., Rei, W., & Ricciardi, N. (2012). Integrating c2e and c2c traffic into city logistics planning. In E. Taniguchi, & R. G. Thompson (Eds.), *Seventh International Conference on City Logistics* (Procedia—social and behavioral sciences) June 7–11, 2011 (Vol. 39, pp. 47–60) Mallorca, Spain: Elsevier.

Crainic, T. G., Errico, F., Rei, W., & Ricciardi, N. (2016a). Modeling demand uncertainty in two-tier city logistics tactical planning. *Transportation Science, 50*(2), 559–578.

Crainic, T. G., Nguyen, P. K., & Toulouse, M. (2016b). Synchronized multi-trip multi-traffic pickup and delivery problem in city logistics. *Transportation Research Procedia, 12*, 26–39.

Crainic, T. G., Gendreau, M., & Jemai, L. (2020). Planning hyperconnected, urban logistics systems. *Transportation Research Procedia, 47*, 19–26.

Crainic, T. G., Rei, W., Hewitt, M., & Maggioni, F. (2021). Partial Benders decomposition: General methodology and application to stochastic network design. *Transportation Science, 55*(2), 414–435.

Dellaert, N., Dashty Saridarq, F., Van Woensel, T., & Crainic, T. G. (2019). Branch & price based algorithms for the two-echelon vehicle routing problem with time windows. *Transportation Science, 53*, 463–479.

Ehmke, J. F., Meisel, S., & Mattfeld, D. C. (2012). Floating car based travel times for city logistics. *Transportation Research Part C: Emerging Technologies, 21*, 338–352

Fontaine, P., Crainic, T. G., Jabali, O., & Rei, W. (2021). Scheduled service network design with resource management for two-tier multimodal city logistics. *European Journal of Operational Research, 294*(2), 558–570.

Freemark, Y. (2011). Opportunities abound for transporting goods by tram—if properly coordinated. Retrieved April 27, 2020 from http://www.thetransportpolitic.com/2011/10/23/opportunities-abound-for-transporting-goods-by-tram-if-properly-coordinated/

Gianessi, P. (2014). Solving strategic and tactical optimization problems in city logistics. (PhD Thesis, Université Paris 13, France).

Gianessi, P., Alfandari, L., Létocart, L., & Calvo, R. W. (2016). The multicommodity-ring location routing problem. *Transportation Science, 50*(2), 541–558.

Gonzalez-Feliu, J., Semet, F., & Routhier, J. L. (Eds.) (2014). *Sustainable urban logistics: Concepts, methods and information systems*. Berlin: Springer.

Gragnani, S., Valenti, G., & Valentini, M. P. (2004). City logistics in Italy: A national project. In E. Taniguchi, & R. G. Thompson (Eds.), *Logistics systems for sustainable cities* (pp. 279–293). Amsterdam: Elsevier.

Grangier, P., Gendreau, M., Lehuédé, F., & Rousseau, L. M. (2015). An adaptive large neighborhood search for the two-echelon multiple-trip vehicle routing problem with satellite synchronization. *European Journal of Operational Research, 254*(1), 80–91.

Guerrero-Lorente, J., Gabor, A. F., & Ponce-Cueto, E. (2020). Omnichannel logistics network design with integrated customer preference for deliveries and returns. *Computers & Industrial Engineering, 144*, 106433.

Guyon, O., Absi, N., Feillet, D., & Garaix, T. (2012). A modeling approach for locating logistics platforms for fast parcels delivery in urban areas. In E. Taniguchi, & R. G. Thompson (Eds.), *Seventh International Conference on City Logistics* (Procedia - social and behavioral sciences) June 7–11, 2011 (Vol. 39, pp. 360–368). Mallorca, Spain: Elsevier.

Hemmelmayr, V. C., Cordeau, J. F., & Crainic, T. G. (2012). An adaptive large neighborhood for two-echelon vehicle routing problems arising in city logistics. *Computers & Operations Research, 39*(12), 3215–3228

Holguín-Veras, J., Leal, J. A., Sánchez-Diaz, I., Browne, M., & Wojtowicz, J. (2020a). State of the art and practice of urban freight management part I: Infrastructure, vehicle-related, and traffic operations. *Transportation Research Part A: Policy and Practice, 137*, 360–382.

Holguín-Veras, J., Leal, J. A., Sánchez-Diaz, I., Browne, M., & Wojtowicz, J. (2020b). State of the art and practice of urban freight management part II: Financial approaches, logistics, and demand management. *Transportation Research Part A: Policy and Practice, 137*, 383–410.

Hu, W., Dong, J., Hwang, B. G., Ren, R., & Chen, Z. (2020). Hybrid optimization procedures applying for two-echelon urban underground logistics network planning: A case study of Beijing. *Computers & Industrial Engineering, 144*, 106452.

Lindholm, M., & Behrends, S. (2012). Challenges in urban freight transport planning—a review in the baltic sea region. *Journal of Transport Geography, 22*, 129–136.

Liu, C., Kou, G., Zhou, X., Peng, Y., Sheng, H., & Alsaadi, F. E. (2020). Time-dependent vehicle routing problem with time windows of city logistics with a congestion avoidance approach. *Knowledge-Based Systems, 188*, 104813.

Maggioni, F., Perboli, G., & Tadei, R. (2014). The multi-path traveling salesman problem with stochastic travel costs: building realistic instances for city logistics applications. *Transportation Research Procedia, 3*, 528–536.

Mancini, S., Gonzalez-Feliu, J., & Crainic, T. G. (2014). Planning and optimization methods for advanced urban logistics systems at tactical level. In J. Gonzalez-Feliu, F. Semet, J.-L. Routhier (Eds.), *Sustainable Urban Logistics: Concepts, Methods and Information Systems* (pp. 145–164). Berlin: Springer, .

Masson, R., Trentini, A., Lehuédé, F., Malhéné, N., Péton, O., & Tlahig, H. (2017). Optimization of a city logistics transportation system with mixed passengers and goods. *EURO Journal on Transportation and Logistics, 6*(1), 81–109.

Montreuil, B. (2011). Towards a physical internet: meeting the global logistics sustainability grand challenge. *Logistics Research, 3*(2–3), 71–87.

Morana, J., Gonzalez-Feliu, J., & Semet, F. (2014). Urban consolidation and logistics pooling—planning, management and scenario assessment issues. In J. Gonzalez-Feliu, F. Semet, J. L. Routhier (Eds.), *Sustainable urban logistics: Concepts, methods and information systems* (pp. 187–210). Berlin: Springer.

Nguyen, P. K., Crainic, T. G., & Toulouse, M. (2013). A Tabu search for time-dependent multi-zone multi-trip vehicle routing problem with time windows. *European Journal of Operational Research, 231*(1), 43–56.

Nguyen, P. K., Crainic, T. G., & Toulouse, M. (2017). Multi-trip pickup and delivery problem with time windows and synchronization. *Annals of Operations Research, 253*(2), 899–934.

Oliveira, B., Ramos, A. G., & de Sousa, J. P. (2020). A classification of two-tier distribution systems based on mobile depots. *Transportation Research Procedia, 47*, 115–122.

Perboli, G., Tadei, R., & Vigo, D. (2011). The two-echelon capacitated vehicle routing problem: Models and math-based heuristics. *Transportation Science, 45*(3), 364–380.

Riemann, H. (2019). Logistiktram project. Retrieved April 27, 2020 from http://logistiktram.de/

Savelsbergh, M., & Van Woensel, T. (2016) 50th anniversary invited article—city logistics: Challenges and opportunities. *Transportation Science, 50*(2), 579–590.

Snoeck, A., Merchán, D., & Winkenbach, M. (2020). Revenue management in last-mile delivery: State-of-the-art and future research directions. *Transportation Research Procedia, 46*, 109–116.

Taniguchi, E. (2014). Concepts of city logistics for sustainable and liveable cities. *Procedia - Social and Behavioral Sciences, 151*, 310–151.

Taniguchi, E., Noritake, M., Yamada, T., & Izumitani, T. (1999). Optimal size and location planning of public logistics terminals. *Transportation Research Part E: Logistics and Transportation Review, 35*(3), 207–222.

Taniguchi, E., Thompson, R. G., Yamada, T., & van Duin, J. H. R. (2001). *City logistics: Network modelling and intelligent transport systems*. Amsterdam: Pergamon.

Taniguchi, E., Thompson, R. G., & Qureshi, A. G. (2020). Modelling city logistics using recent innovative technologies. *Transportstion Research Procedia, 46*, 3–12.

Trentini, A., & Mahléné, N. (2010). Toward a shared urban transport system ensuring passengers & goods cohabitation. *Tema Journal of Land Use, Mobility and Environment, 3*(2), 37–46.

Winkenbach, M., Kleindorfer, P. R., & Spinler, S. (2016). Enabling urban logistics services at *La Poste* through multi-echelon location-routing. *Transportation Science 50*(2), 520–540.

Chapter 17
Public Transportation

Antonio Mauttone, Héctor Cancela, and María E. Urquhart

1 Introduction

Public Transportation (PT) refers to shared transportation services (Teodorovic and Janic 2016) which operate using infrastructure like roads or rails, and vehicles like buses or trains. Usually, it includes urban public transit and intercity public transportation, both characterized by fixed routes and schedules which are available for use by all persons who pay the established fare (Vuchic 2007). PT has been gaining importance since sustainability is increasingly identified as one of the primary goals of the society. When compared against other motorized transport modes, PT exhibits higher efficiency rates in terms of energy consumption, greenhouse emissions, noise pollution and usage of public space. However, both setting and operating of PT systems involve very large expenditures. Moreover, the performance of these systems from the viewpoint of the users is a key aspect in order to offer a successful service, which reveals the need for effective planning methodologies.

The planning of PT systems offers various opportunities for optimization. The whole process can be decomposed into several planning stages which define a sequence of hierarchical decisions, namely, network design, frequency and timetable determination, and fleet and crew scheduling (Ceder and Wilson 1986; Goosens et al. 2004). According to this approach, network design plays a very relevant role within the overall planning process since it impacts in every subsequent stage, and therefore in every component cost of the system. In that context, the meaning of the term *public transport network* depends on the specific mode. For systems based on buses which share the street with regular vehicles (i.e., cars) there is no cost of infrastructure building, or it can be negligible. On the other hand, rapid

A. Mauttone (✉) · H. Cancela · M. E. Urquhart
Department of Operations Research, Universidad de la República, Montevideo, Uruguay
e-mail: mauttone@fing.edu.uy; cancela@fing.edu.uy; urquhart@fing.edu.uy

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_17

transit, rail and metro systems involve large investments due to building of exclusive corridors, railways and tunnels; we refer to these elements as the *physical network*. In addition, in PT systems there is a second network level which is defined by the services that operate over the street network for classical bus systems and over the physical network for rapid transit, rail and metro systems: the routes followed by the vehicles. This level is referred as the *route network*. The operation of these routes determines a relevant component of the operating cost of the system, in the form of vehicle and personnel cost per distance and time unit respectively. Also, both topological structure and frequency (vehicles passing per time unit) of the routes determine largely the level of service offered to the users in terms of overall travel time, which includes time spent walking (from origin and to destination stops or stations), waiting, and on-board the vehicle.

When approached as a network design problem, models of both physical and route networks represent stops and stations as network nodes, and street and rail sections as network links. Usually, the nodes are fixed and decisions are related to inclusion or exclusion of the links in the solution. In the most general case, each link has attributes like building cost, travel time and capacity. A first approach for PT network design is the general fixed charge network design model described in Chap. 2, where each commodity represents a specific group of people traveling from some origin node to another destination node. However, there are distinctive characteristics of PT systems which add particular difficulties. When modeling the design of routes, decisions are not related to single links, instead they refer to a sequence of links. Moreover, the system is composed by several routes which may overlap, i.e., they can share common links. Also, the performance evaluation of the system from the viewpoint of the users entails modeling their behavior with respect to the set of enabled links or routes. This is a very particular characteristic of PT network design models, since users behave by themselves and their interests can be conflicting with the interests of the operator, which entails considering special features in the corresponding network design models.

This chapter presents concepts, models and solution methods for PT network design. In order to precise the scope, we consider only models that include topological variables (Farahani et al. 2013), i.e., variables which represent decisions about nodes and links. However, we also consider models that include non-topological variables like frequency or vehicle size, when they represent decisions which are taken simultaneously with topological ones. These non-topological variables have great influence on PT network design, although they do not define directly its structure.

Regarding chapter organization, Sect. 2 states the main concepts and notation whereas Sect. 3 presents several models for PT network design including several problem aspects. Section 4 presents relevant solution methods, both for the models presented at Sect. 3 as for other models whose solution approaches exhibit key algorithmic concepts. Section 5 presents a compilation of the main bibliography in chronological order, whereas Sect. 6 offers perspectives for future research in the topic.

2 Background

2.1 Basic Concepts and Notation

The physical structure of PT systems suggests a direct network representation. The elements of the network have attributes which represent parameters of the users (people who use the services), the operators (companies or agencies which offer the services) and the whole society (typically related to infrastructure building).

Let $\mathscr{G}$ be a directed graph with corresponding set of nodes $\mathscr{N}$ which represents junctions, stops or stations and set of arcs $\mathscr{A}$ which represents sections of streets or rails between nodes. There are several ways to build the graph model $\mathscr{G}$ from real data; the international community has not agreed in a single standard one (Heyken Soares et al. 2019). Given that in many cases PT systems exhibit a symmetric pattern of services, we also consider an undirected variant of $\mathscr{G}$ with corresponding sets of vertices $\mathscr{V}$ and edges $\mathscr{E}$. An arc $a \in \mathscr{A}$ (edge $e \in \mathscr{E}$) can be identified by its corresponding ordered (unordered) pair of endpoint nodes (i, j) (vertices $[i, j]$). Also, for each arc $a \in \mathscr{A}$ we define user cost $c_a^u \geq 0$ which represents the cost (usually travel time) experienced by the user when traversing a, and operator cost $c_a^o \geq 0$ which represents the cost incurred by the operator due to offering a service which traverses a (usually travel time or distance). Undirected versions of both user and operator cost are defined for edges $e \in \mathscr{E}$, namely, c_e^u and c_e^o respectively. In general terms, each arc $a \in A$ has capacity $u_a \geq 0$ which states the maximum flow that can traverse a per time unit. The entities which flow over arcs can be either persons (mostly referred as users or passengers) or vehicles (buses, trains), depending on the specific context. Thus, if an arc represents a physical element (street or rail section), the flow is measured in terms of vehicles and its magnitude is directly proportional to the frequencies of the services which operate over the arc. On the other hand, if an arc represents a route section, the flow is measured in terms of passengers and its magnitude is directly proportional to the demand attracted by the route, i.e., the passengers traveling on-board the vehicles operating the route. The passenger demand for PT is modeled as a set of commodities $\mathscr{K}$. Each element $k \in \mathscr{K}$ has origin and destination nodes $O(k)$ and $D(k)$, respectively, and demand (passengers per time unit in a given time horizon) $d_k > 0$ between $O(k) \in \mathscr{N}$ and $D(k) \in \mathscr{N}$. In the context of PT systems, $\mathscr{K}$ defines an *origin-destination (OD) matrix* and each element $k \in \mathscr{K}$ is called *OD pair*. These commodities share the same network, therefore, when applicable, they are collectively constrained by arc capacities. The demand corresponding to OD pair $k \in \mathscr{K}$ is said to be *covered* if $O(k)$ and $D(k)$ are connected by the PT network, independent on its capacity. Moreover, if the network capacity allows flowing the whole amount d_k, the demand is also said to be *satisfied*.

A PT route is defined as a sequence of adjacent nodes or vertices in $\mathscr{G}$ and it has a cyclic pattern. When defined in terms of undirected edges, it is assumed that it operates in both directions. On the other hand, directed routes should be defined as cycles in $\mathscr{G}$. Let $\mathscr{R}$ be the set of all routes in $\mathscr{G}$ according to this definition. In the most general case, a route stops at every node where it passes. Therefore, passengers can access the corresponding service (either to board or to alight) in all those nodes. Each route $r \in \mathscr{R}$ has frequency $f_r \geq 0$ which expresses the number of vehicles per time unit operating the route. The special case $f_r = 0$ is sometimes used to state that route r is disabled. A route with its frequency is sometimes referred as a *line*.

The operation cost of routes depends on both distance and time. Assuming a constant average speed, the distance component of the variable cost of a route $r \in \mathscr{R}$ is proportional to its cycle time $\sum_{a \in r} c_a^o$. Moreover, the time component is proportional to its frequency f_r. A combined measure of the variable cost can be defined as $f_r \sum_{a \in r} c_a^o$, which stands for the number of vehicles that operate simultaneously in r. In the most general case, this measure is taken as a proxy for operation cost.

Regarding user cost, the PT network determines one of its main attributes: travel time. For OD pair $k \in \mathscr{K}$, it is assumed without loss of generality that users travel along the shortest path defined by the enabled arcs, which can be formulated as

$$\text{Minimize} \quad \sum_{a \in \mathscr{A}} c_a^u x_a \tag{17.1}$$

$$\text{Subject to} \quad \textstyle\sum_{a \in \mathscr{A}_n^+} x_a - \sum_{a \in \mathscr{A}_n^-} x_a = w_{nk}, \quad \forall n \in \mathscr{N}, \tag{17.2}$$

$$0 \leq x_a \leq u_a, \qquad \forall a \in \mathscr{A}, \tag{17.3}$$

where sets $\mathscr{A}_n^+ \subseteq \mathscr{A}$ and $\mathscr{A}_n^- \subseteq \mathscr{A}$ denote outgoing and incoming arcs respectively of node $n \in \mathscr{N}$ and w_{nk} is equal to d_k if $n = O(k)$, $-d_k$ if $n = D(k)$ and 0 otherwise.

Formulation (17.1)–(17.3) denotes a *minimum cost flow problem* (Ahuja et al. 1993), where decision variable x_a represents the flow of passengers over arc a. Moreover, the value u_a defines a capacity constraint for arc $a \in \mathscr{A}$. Thus, if a is enabled because its corresponding physical link is built and there is a line which operates over it, a sufficiently large value of u_a will allow the entire demand d_k to flow over a. This is the case in which all passengers follow the same (shortest) path from $O(k)$ to $D(k)$. But if u_a values are not large enough, some passengers are forced to take other paths with larger cost due to insufficient capacity in the shortest path, which gives rise to a *capacitated user equilibrium* with constant arc cost (Correa et al. 2004). This is a variation of the classical equilibrium in private car networks where the cost of each arc depends on its flow, therefore all different paths followed by the demand corresponding to the same OD pair have the same cost (Sheffi 1985). Typically, arc capacities in PT networks are defined by the capacity of the infrastructure (allowable speed, number of lanes) and the services (route frequency and vehicle capacity). In PT network design models, these elements can be either fixed parameters as well as decision variables. In any case, let $P_k \in \mathscr{A}$ ($\mathscr{E}$ in the undirected version) denote the set of arcs (edges) with flow greater than zero in the optimal solution of (17.1)–(17.3). If capacities allow the whole demand

d_k to flow over the same path, then P_k denotes the shortest path between $O(k)$ and $D(k)$. Otherwise, it denotes the set of arcs corresponding to all the paths followed by the demand. Any of these paths can represent either direct trips (using a single line) or trips with transfers (using two or more lines), depending on the modeling of the network and the hypothesis assumed regarding passenger behavior.

So far, formulation (17.1)–(17.3) represents reasonably the passenger behavior taking into account only the on-board travel time. Walking time can also be modeled using this formulation, by including specific nodes that represent trip origins and destinations (e.g., nodes representing geographical zones) and walking arcs connecting these nodes with the stops and stations. However, in some cases the waiting time should also be considered, either as an attribute for shortest path calculations or as a parameter for evaluating system performance. The waiting time at the stop is non-linearly related to the frequency of the line or set of lines which lead to destination. This phenomenon entails more complex formulations. Moreover, the effect of capacity over the waiting time and the flow distribution on lines leads to even more complex formulations with respect to passenger behavior.

2.2 Problem Nomenclature, General Formulation and Solution Approach for Public Transportation Network Design

Public transportation network design involves managing several levels of networks. In this context, we denote as PND (Physical Network Design) the problem of designing the physical network, i.e., decisions related to building dedicated bus lanes, rail or metro lines. Moreover, we denote as RND (Route Network Design) the problem of designing the routes over an existing physical network. This may comprise the design of a single route with a particular goal or the design of a complete set of routes to satisfy the whole demand of a given scenario.

In order to formulate a general optimization model for public transportation network design, firstly we can identify topological and non-topological decision variables (Farahani et al. 2013), namely, X^t and X^n respectively. In the first group there are decisions related to nodes and arcs of $\mathscr{G}$, e.g., station location, rail building or route structure. Relevant non-topological decision variables include route frequency and vehicle capacity. Moreover, in PT network design models we need variables that represent the behavior of passengers, namely, X^b. These variables are not controlled directly by the planner, however, they depend on his decisions regarding infrastructure building and service provision. For that reason, usually they are modeled explicitly since they determine a relevant component of system performance.

The objective function expresses the goal of the planner, which may take several forms. It can be either a direct formulation of the interests of both users and operators, or it can represent a more general system goal. Very often, the planner is forced to manage opposite interests. For instance, a high number of routes with high frequencies contribute to increase the level of service from the viewpoint of the user, but it causes high operation costs as well, which might not be sustainable in the economic sense. This leads to consider multiobjective formulations (Ehrgott 2005).

The modeling of passenger behavior entails considering a hierarchical process where the planner makes a decision (e.g., regarding routes) and the passengers choose their routing over those services, producing flow values which are necessary for the planner in order to fully compute its measure of system performance. Despite the fact that this hierarchical process in some cases can be modeled properly as a standard optimization problem, its most general formulation entails a multiple-level (more specifically, two-level or bilevel) formulation (Bard 1998).

Finally, the constraints can be of several types, ranging from criteria of the planners (which may include performance indicators of users, operators and the overall system) to physical constraints regarding route structure, infrastructure and vehicle capacity. Budgetary constraints imposed over infrastructure building and service operation are often included as well.

A generic formulation for the public transportation network design problem can be defined as (17.4)–(17.7). For $m > 1$ the objective function is a vector which represents several goals which should be taken into account simultaneously. Constraint (17.5) may take standard forms like equalities or inequalities. Constraint (17.6) states that passenger behavior variables X^b should take the optimal value of an additional optimization problem, where $\bar{X}^b$ are decision variables and H states the criterion of the users for traveling over the network set by the planner through fixed values X^t and X^n, constrained by function Z. An example of this second level optimization problem is the shortest path routing stated by (17.1)–(17.3).

$$\text{Minimize } \; [F_1(X^t, X^n, X^b), \ldots F_m(X^t, X^n, X^b)] \tag{17.4}$$

$$\text{Subject to} \qquad G(X^t, X^n, X^b) \leq 0, \tag{17.5}$$

$$X^b = \text{argmin } \; H(X^t, X^n, \bar{X}^b), \tag{17.6}$$

$$\text{Subject to } \; Z(X^t, X^n, \bar{X}^b) \leq 0. \tag{17.7}$$

Both PND and RND addressed as optimization problems, exhibit several sources of complexity. The underlying network design problem already has a combinatorial structure which entails high computational complexity (Johnson et al. 1978). The feasible space of topological variables of RND is huge, given the size of the set $\mathscr{R}$ of all possible routes. Passenger behavior sub-models usually included as the second level problem in (17.4)–(17.7) add complexity to the overall formulation, especially when the more complex variants of (17.1)–(17.3) are considered. The multiobjective and multilevel structure poses the need for specific resolution methods, which can be either exact or heuristic. In general terms, exact methods always rely over an explicit mathematical programming formulation. Conversely, these formulations are often used to implement heuristic methods instead of exact ones. The RND problem is approached by two different strategies in order to determine the values of topological variables X^t: (1) generating a pool of many good candidate routes (which we call *route generation*) and then selecting the optimal subset (*route selection*) and (2) generating a set of routes which constitutes a feasible solution, which may be improved in a further stage (*route set generation and improvement*). Moreover, heuristic and metaheuristic methods for RND often decouple the sub-problems of determining the optimal values for non-topological variables X^n and passenger behavior variables X^b. The resolution of these sub-problems are coded into specific sub-routines which are called appropriately during the overall optimization process.

3 Models for Public Transportation Network Optimization

In this section we present several models for both PND and RND problems. The passenger behavior appears explicitly on RND, since a full characterization of the public transportation services (lines) is modeled. For that reason Sects. 3.1–3.4 focus on RND, assuming a physical network already established. The models presented apply to different PT modes, which share common elements in the context of strategic and tactical planning, namely, networks, lines, passengers, vehicles, capacities and budgetary constraints. Differences among the general hypotheses assumed in the models presented, are mainly due to the specific transport mode under discussion. Thus, in models for intercity railway line planning (Sect. 3.1), the underlying network is sparse and the passengers are assumed to schedule their arrival to the station according to the timetable. In models for bus line planning (Sect. 3.2) the underlying network (streets) is assumed to be dense. In bus based systems including services with different characteristics regarding frequency and regularity, the modeling of waiting time is relevant (Sect. 3.3). Whenever line capacity comes into play (Sect. 3.4), the services should be designed taking into account the reaction of the users. The issue of transfers between lines appears in almost every medium to large sized scenario. Transfers have a great impact on both users (perceived level of service) and operators (number of lines, which influences operations cost), and its modeling is not straightforward.

Table 17.1 provides a list of main symbols used in this section. Moreover, the nonnegative real variable x is used to denote flow, either over arcs a, routes r and paths p, also indexed by commodity k. Similarly, the binary variable y is used to denote the decision of including a route r or line l into the solution.

Table 17.1 Definitions of main symbols

Symbol	Definition
$\mathscr{G}$	Graph representing the underlying network
$\mathscr{N}$ ($\mathscr{V}$)	Nodes (vertices) of the graph
$\mathscr{A}$ ($\mathscr{E}$)	Directed arcs (undirected edges) of the graph
$\mathscr{A}_n^+$ ($\mathscr{A}_n^-$)	Incoming (outgoing) arcs to (from) node n
c_a^u (c_e^u)	User cost of arc a (edge e)
c_a^o (c_e^o)	Operator cost of arc a (edge e)
u_a	Capacity of arc a
$\mathscr{K}$	Set of commodities (OD pairs)
$O(k)$ ($D(k)$)	Origin (destination) node of commodity k
d_k	Demand of commodity k
w_{nk}	Equal to d_k if $n = O(k)$, $-d_k$ if $n = D(k)$ and 0 otherwise
$\mathscr{R}$	Set of all routes defined over $\mathscr{G}$
$\mathscr{R}_0$	Pool of candidate routes
f_r	Frequency of route r

3.1 *User and Operator Oriented Models with Fixed Passenger Behavior*

In railway systems it is reasonable to assume that services will be provided along shortest paths from passenger viewpoint over the physical network. This allows introducing the *system-split* hypothesis, which states that passengers always travel along shortest paths in $\mathscr{G}$ (with respect to cost c_a^u) independently of the routes. Consequently, the passenger behavior can be fixed, thus simplifying the models by solving a priori problem (17.1)–(17.3) for each commodity $k \in \mathscr{K}$ and loading the corresponding flows over the network links.

Model (17.8)–(17.13) selects an optimal subset of routes with their corresponding frequencies, from a given pool $\mathscr{R}_0 \subseteq \mathscr{R}$ of routes defined over the physical network (Bussieck et al. 1997). The model adopts the undirected versions of both $\mathscr{G}$ and $\mathscr{R}$, and takes into account demand data given as an OD matrix. The system performance is represented by the amount of direct demand satisfied, denoted by x_{rk} for route $r \in \mathscr{R}_0$ and OD pair $k \in \mathscr{K}$.

Constraint (17.9) bounds the passenger flow (thus preventing infinite values) by the demand of each OD pair, while constraint (17.10) links passenger flow with the capacity of each route r, which is defined as the product of the train capacity C and the route frequency f_r. Finally, constraint (17.11) states that the sum of the frequencies of all routes passing by edge e must be equal to the load of that edge (t_e, resulting from the fixed system-split flows computed a priori) divided by the train capacity. This last constraint prevents unnecessary high frequencies by setting values which ensure route capacity. The routes included in the solution are those r such that $f_r > 0$.

$$\text{Maximize} \sum_{r \in \mathscr{R}_0} \sum_{k \in \mathscr{K}, P_k \subseteq r} x_{rk} \tag{17.8}$$

$$\begin{aligned}
\text{Subject to} \quad & \textstyle\sum_{r \in \mathscr{R}_0, P_k \subseteq r} x_{rk} \le d_k, && \forall k \in \mathscr{K}, && (17.9)\\
& \textstyle\sum_{k \in \mathscr{K}, e \in P_k \subseteq r} x_{rk} \le C f_r, && \forall e \in \mathscr{E},\ r \in \mathscr{R}_0, && (17.10)\\
& \textstyle\sum_{r \in \mathscr{R}_0, e \in r} f_r = \ulcorner t_e / C \urcorner, && \forall e \in \mathscr{E}, && (17.11)\\
& x_{rk} \ge 0, && \forall r \in \mathscr{R}_0,\ k \in \mathscr{K}, && (17.12)\\
& f_r \in \mathbb{Z}_+, && \forall r \in \mathscr{R}_0. && (17.13)
\end{aligned}$$

Note that depending on the routes included in the pool $\mathscr{R}_0$, the whole demand $d_k, \forall k \in \mathscr{K}$ will be satisfied (either directly or indirectly) or not. If for each $k \in \mathscr{K}$, the pool $\mathscr{R}_0$ includes at least one route comprising both $O(k)$ and $D(k)$, the whole demand is likely to be satisfied directly. This kind of solution does not take into account explicitly the interest of the operator, since there is not an explicit upper bound on the number of lines. For this reason, formulation (17.8)–(17.13) is referred as user oriented.

On the other hand, operator oriented models usually seek to minimize operation costs (Goosens et al. 2004). We use the concept of line to define set $\hat{\mathscr{R}}_0 = \mathscr{R}_0 \times \mathscr{F} \times \mathscr{S}$, where $\mathscr{F} \subset \mathbb{Z}_+$ denotes possible values of frequencies and $\mathscr{S} \subset \mathbb{Z}_+$ denotes possible values for number of carriages, both corresponding to each route $r \in \mathscr{R}_0$. Each element $l \in \hat{\mathscr{R}}_0$ has route r_l, frequency f_l and number of carriages s_l. Model (17.14)–(17.18) also assumes an a priori system-split loading of OD flows to each edge e of the network, which determines the required frequency f_e and number of carriages s_e. Parameter k_l states the line cost (including fixed and variable components per train and carriage), while y_l is a binary decision variable which states whether or not to include line $l \in \hat{\mathscr{R}}_0$ in the solution.

$$\text{Minimize} \sum_{l \in \hat{\mathscr{R}}_0} k_l y_l \tag{17.14}$$

$$\begin{aligned}
\text{Subject to} \quad & \textstyle\sum_{l \in \hat{\mathscr{R}}_0(e)} f_l y_l \ge f_e, && \forall e \in \mathscr{E}, && (17.15)\\
& \textstyle\sum_{l \in \hat{\mathscr{R}}_0(e)} f_l s_l y_l \ge c_e, && \forall e \in \mathscr{E}, && (17.16)\\
& \textstyle\sum_{l \in \mathscr{R}_0, r_l = r} y_l \le 1, && \forall r \in \mathscr{R}_0, && (17.17)\\
& y_l \in \{0, 1\}, && \forall l \in \hat{\mathscr{R}}_0. && (17.18)
\end{aligned}$$

Constraints (17.15) and (17.16) ensure capacity fulfillment by setting appropriate values of frequency and number of carriages, where $\hat{\mathscr{R}}_0(e) = \{l \in \hat{\mathscr{R}}_0 / e \in r_l\}$.

Constraint (17.17) ensures that for each route $r \in \mathscr{R}_0$, at most one line from $\hat{\mathscr{R}}_0$ is selected.

Note that formulation (17.14)–(17.18) minimizes operation costs, while passengers' interest is taken into account by the system-split hypothesis and the constraints which ensure sufficient capacities in the selected lines.

3.2 Explicit Modeling of Passenger Behavior

If the physical network is dense, there are many possibilities for defining routes. This is the case of bus based systems, where the physical network is defined in terms of the streets. In this scenario, the system-split approach is not a reasonable assumption. Therefore, since the demand flows cannot be fixed a priori, the passenger behavior is represented explicitly by means of specific decision variables (Borndörfer et al. 2007). For route $r \in \mathscr{R}$, let y_r be a binary (topological) variable which states whether or not r is included in the solution and let f_r be a real (non-topological) variable which represents its frequency. While routes are defined over the undirected version of $\mathscr{G}$, passenger paths are defined over its directed counterpart. Let $\mathscr{P}$ be the set of all directed passenger paths in $\mathscr{G}$ and let $\mathscr{P}(k) \subseteq \mathscr{P}$ be the set of paths from $O(k)$ to $D(k)$. The path-based formulation is defined by (17.19)–(17.25), where the behavioral variable x_p stands for the amount of flow over path p.

$$\text{Minimize} \quad \sum_{p \in \mathscr{P}} \sum_{a \in p} c_a^u x_p + \sum_{r \in \mathscr{R}} \left(k_r^f y_r + k_r^v f_r \right) \tag{17.19}$$

$$\begin{aligned}
\text{Subject to} \qquad & \textstyle\sum_{p \in \mathscr{P}(k)} x_p = d_k, && \forall k \in \mathscr{K}, && (17.20)\\
& \textstyle\sum_{p \in \mathscr{P}/a \in p} x_p \le \sum_{r \in \mathscr{R}/a \in r} C_r^p f_r, && \forall a \in \mathscr{A}, && (17.21)\\
& \textstyle\sum_{r \in \mathscr{R}(e)} f_r \le C_e^v, && \forall e \in \mathscr{E}, && (17.22)\\
& f_r \le F y_r, && \forall r \in \mathscr{R}, && (17.23)\\
& y_r \in \{0,1\},\ f_r \ge 0, && \forall r \in \mathscr{R}, && (17.24)\\
& x_p \ge 0, && \forall p \in \mathscr{P}. && (17.25)
\end{aligned}$$

Unlike the models presented in Sect. 3.1, objective function (17.19) represents simultaneously the interest of both users and operators. The first term accounts for total travel time of users while the second one groups both fixed and variable operator cost, using parameters k_r^f and k_r^v respectively for route $r \in \mathscr{R}$. Constraint (17.20) imposes flow conservation for passenger demand over paths. Line capacity is ensured by constraint (17.21), where parameter C_r^p stands for the number of places in vehicles performing route r. Similarly, constraint (17.22) ensures that lines passing by edge e (street section) do not surpass collectively its capacity (measured in terms of vehicles per time unit), stated by parameter C_e^v. Finally, constraint (17.23) states that the frequency of route r can be greater than zero only if r is part of the solution, where F is a parameter whose value should be sufficiently high.

Formulation (17.19)–(17.25) denotes a multicommodity flow problem with capacities imposed to both route frequencies and passenger flows. The first term of the objective function ensures that passengers follow the shortest path over the network resulting from the enabled routes. Moreover, two issues are worth to be mentioned. First, due to constraint (17.21), the flow of a given OD pair $k \in \mathscr{K}$ may be split into several paths with different cost due to insufficient capacity on the shortest path (capacitated user equilibrium). Second, transfers between lines are ignored, since the flow over a specific path is enabled by constraint (17.21) if each of its arcs belongs to at least one route enabled by constraint (17.23). This means that in the optimal solution, passengers may be forced to perform an arbitrary number of transfers between routes. The first issue is further discussed in Sect. 3.4 while the second one may be approached by using the expanded network $\hat{\mathscr{G}}(\mathscr{R}_0)$ shown in Fig. 17.1, where each node of $\mathscr{G}$ is replicated for each $r \in \mathscr{R}_0 \subseteq \mathscr{R}$. Each arc is also replicated for each route, which allows to model different costs for different lines passing by the same arc. Transfer arcs are added to connect nodes which represent the same stop or station for different lines. Finally, boarding and alighting arcs are added to connect origins and destinations with stops or stations. By using this expanded network, transfers between routes can be weighted and counted in the optimization models.

3.3 Including Waiting Time

In public transportation systems, waiting time is recognized as one of the most onerous components of the user total travel time. In some cases, ignoring waiting time in the modeling may be justified reasonably. For instance, users of intercity services with low frequency can schedule their arrivals to the stop or station, assuming that timetable information is available and reliable. Moreover, users of metro or rapid transit systems may experience low waiting time due to availability of high frequency services. However, in other systems like most of bus based ones, modeling of waiting time is relevant in order to state a realistic scenario.

To do that, an expanded network as shown in Fig. 17.1 (without transfer arcs) is used (Cancela et al. 2015), where $\mathscr{A}^b \subset \mathscr{A}$ denotes the set of boarding arcs.

Over this network, we can formulate the problem of selecting the optimal subset of routes from a provided pool $\mathscr{R}_0$ and setting the frequency for each selected route, taken from a discrete set of values $\mathscr{F} = \{F_1, \dots F_m\}$ indexed by q. This discretization of frequencies is introduced to obtain a linear formulation. Each element of $\mathscr{F}$ (therefore, each possible value of frequency) has its own boarding

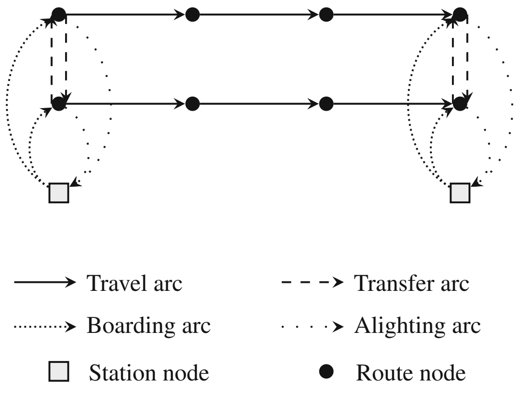


Fig. 17.1 Expanded network comprising two routes passing by two common stations

arc in the network. Let y_r be a binary topological variable which expresses whether route $r \in \mathscr{R}_0$ is selected and f_{rq} be a non-topological binary variable which states that frequency F_q is assigned to route r. Moreover, let x_{ak} be the amount of demand corresponding to OD pair k which flows over arc a and let z_{nk} be the waiting time multiplied by the flow of OD pair k at node n (both x and z are behavioral variables). The maximum number of available vehicles is denoted by parameter B.

$$\text{Minimize} \sum_{k\in\mathscr{K}} \left(\sum_{a\in\mathscr{A}} c_a^u x_{ak} + \sum_{n\in\mathscr{N}} z_{nk} \right) \tag{17.26}$$

$$\text{Subject to} \quad \textstyle\sum_{r\in\mathscr{R}_0} 2 \sum_{q\in\mathscr{F}} F_q f_{rq} \sum_{e\in r} c_e^o \le B, \quad \forall k \in \mathscr{K}, \tag{17.27}$$

$$\textstyle\sum_{a\in\mathscr{A}_n^+} x_{ak} - \sum_{a\in\mathscr{A}_n^-} x_{ak} = w_{nk}, \quad \forall n \in \mathscr{N},\ k \in \mathscr{K}, \tag{17.28}$$

$$x_{ak} \le \mathscr{F}(a) z_{nk}, \quad \forall a \in \mathscr{A}^{b+},\ n \in \mathscr{N},\ k \in \mathscr{K}, \tag{17.29}$$

$$x_{ak} \le d_k y_{\mathscr{R}_0(a)}, \quad \forall a \in \mathscr{A},\ k \in \mathscr{K}, \tag{17.30}$$

$$x_{ak} \le d_k f_{\mathscr{R}_0(a)\mathscr{F}(a)}, \quad \forall a \in \mathscr{A}^b,\ k \in \mathscr{K}, \tag{17.31}$$

$$\textstyle\sum_{q\in\mathscr{F}} f_{rq} = y_r \quad \forall r \in \mathscr{R}_0, \tag{17.32}$$

$$x_{ak} \ge 0, \quad \forall a \in \mathscr{A},\ k \in \mathscr{K}, \tag{17.33}$$

$$z_{nk} \ge 0, \quad \forall n \in \mathscr{N},\ k \in \mathscr{K}, \tag{17.34}$$

$$y_r \in \{0, 1\}, \quad \forall r \in \mathscr{R}_0, \tag{17.35}$$

$$f_{rq} \in \{0, 1\}, \quad \forall r \in \mathscr{R}_0,\ q \in \mathscr{F}. \tag{17.36}$$

Formulation (17.26)–(17.36) minimizes user total travel time, including on-board and waiting components. Constraint (17.27) imposes a limit on the number of vehicles used, thus representing the interest of the operator, while constraint (17.28) is a typical flow conservation condition. Activation constraint (17.30) states that demand can flow only over arcs of enabled routes, while a similar activation constraint (17.31) states that demand can flow only over arcs corresponding to the frequency assigned to each route. In these expressions, $\mathscr{R}_0(a)$ and $\mathscr{F}(a)$ denote the route from $\mathscr{R}_0$ and the frequency of $\mathscr{F}$ respectively, corresponding to arc $a \in \mathscr{A}$. Constraint (17.32) states that only one value of frequency from $\mathscr{F}$ can be assigned to each route. Finally, constraint (17.29) models the fact that passengers corresponding to OD pair k waiting at node n are distributed among the set of most convenient lines (in the sense of overall expected travel time) that lead to their destination. For fixed values of variables y and f, the result corresponds to the *optimal strategies* passenger behavior model (Spiess and Florian 1989). That model assumes that: (1) users seek to minimize the expected total travel time along the network, (2) the waiting time is inversely proportional to the sum of the frequencies of lines which lead to destination, and (3) the distribution of demand among these lines is proportional to their frequencies. A direct formulation of these assumptions followed by a series of algebraic transformations (Spiess and Florian 1989) allow to observe that the formulation of this passenger behavior model corresponds to a variation of the shortest path problem (17.1)–(17.3), where the waiting time term is added in the objective function and the flow-splitting constraint (17.29) distributes the demand flow among different routes passing by the same stop.

3.4 *Multiple Objectives and Levels of Decisions*

The models presented in previous sections consider decisions of different stakeholders within a single formulation having a standard structure. In some cases this can be a reasonable modeling approach, however, there are situations where a more structured formulation is needed in order to model properly particular characteristics of the problem, namely:

- Different stakeholders may have conflicting objectives, therefore it is impossible to arrive to the best solution from a single point of view. In public transportation systems we can observe this interplay between users and operators, which reveals the multiobjective nature of the problem (Ehrgott 2005). The models presented in Sect. 3.1 are biased by definition towards some of these specific objectives. The models of Sects. 3.2 and 3.3 formulate implicitly multiobjective problems and allow for exploring different compromise solutions by weighting and constraining objectives.
- Some stakeholders may require to know the reaction of subordinate ones, in order to fully determine their decisions. Since most public transportation network design models are conceived to support decisions of the planners, the way in which passengers use the routes should be modeled in order to know its consequence over the system performance. This modeling requires considering different levels of decisions, where there is a leader (the planner) who restrains

decisions of a follower (the passengers), in order to arrive to an optimal solution for the whole system. This characteristic of many passenger transportation problems entails formulating a two-level (bilevel) optimization problem (Bard 1998). The models presented in Sects. 3.2 and 3.3 include variables which represent decisions of the planner (y and f) and the passengers (x and z), which are pushed jointly towards the same direction by the objective functions and constraints.

Model (17.37)–(17.43) optimizes simultaneously the objectives of users and operators while ensuring sufficient capacity in the lines that passengers decide to use (Goerigk and Schmidt 2017). Lines are taken from a provided set $\mathscr{R}_0$. Let C_r^o be the operation cost (e.g., length) of route $r \in \mathscr{R}_0$ and $C \in \mathbb{N}$ be the capacity of vehicles, expressed in number of passengers. The remaining symbols are defined as in previous sections.

The existence of two objective functions implies that the optimal solution is the set of all *efficient* (or *Pareto optimal*) solutions, instead of a single optimal solution. That set represents the whole range of optimal trade-off levels (in terms of routes and frequencies) between both objectives of vector (17.37). The lower-level problem (17.40)–(17.43) states that passengers move along the shortest path defined by the routes enabled by the upper-level, i.e., those with $f_r > 0$. Equation (17.40) states that variable x_{ak} of the upper-level must take optimal values from its lower-level counterpart $\bar{x}_{ak}$. Constraint (17.38) determines frequencies in order to allow passengers moving along shortest paths with sufficient capacity. This means that passengers perceive unlimited capacity in routes, therefore, for each OD pair $k \in \mathscr{K}$ the demand d_k is not split. Note that by eliminating objective (17.40) and moving constraints (17.41)–(17.43) to the upper-level, we would obtain a (single level) relaxation of the original problem where the routing of passengers follows a capacitated user equilibrium.

$$\text{Minimize } [\sum_{k\in\mathscr{K}} \sum_{a\in\mathscr{A}} c_a^u x_{ak}, \sum_{r\in\mathscr{R}_0} f_r C_r^o] \tag{17.37}$$

$$\text{Subject to} \quad \textstyle\sum_{k\in\mathscr{K}} x_{ak} \leq f_{\mathscr{R}_0(a)} C, \quad \forall a \in \mathscr{A}, \tag{17.38}$$

$$f_r \in \mathbb{N}, \quad \forall r \in \mathscr{R}_0, \tag{17.39}$$

$$x_{ak} \in \text{argmin } \textstyle\sum_{a\in\mathscr{A}} c_a^u \bar{x}_{ak}, \tag{17.40}$$

$$\text{Subject to } \textstyle\sum_{a\in\mathscr{A}_n^+} \bar{x}_{ak} - \sum_{a\in\mathscr{A}_n^-} \bar{x}_{ak} = w_{nk}, \quad \forall n \in \mathscr{N}, k \in \mathscr{K}, \tag{17.41}$$

$$\bar{x}_{ak} \leq d_k f_{\mathscr{R}_0(a)}, \quad \forall a \in \mathscr{A}, k \in \mathscr{K}, \tag{17.42}$$

$$\bar{x}_{ak} \in \mathbb{N}, \quad \forall a \in \mathscr{A}, k \in \mathscr{K}. \tag{17.43}$$

3.5 Other Relevant Models

The problem of route design in bus rapid transit systems exhibits particular characteristics. First, routes are defined over predefined corridors with linear structure, unlike the mesh-like structure of the street network used by regular bus based systems. Moreover, for a given corridor comprising n stations or stops, the number of possible routes is 2^n since limited-stop services are under consideration in order to reduce travel time. Thus, several parallel routes can be defined over the same corridor, each of them having a different set of stops. To model this feature, an expanded network similar to the one shown in Fig. 17.1 can be used, where each station is replicated for each route (Walteros et al. 2015). Both on-board (travel) and walking arcs are considered, including arcs which model access to the stations, walking inside the stations and changing of routes at the same station. Whenever a route skips a station, the travel time between its previous and next stations must fulfil the triangular inequality, thus modeling the fact that there is no delay due to skipping intermediate stations. The domain of topological variables is defined by all possible routes over all corridors. Typical constraints include arc capacity given by the capacity of the stations and the lines. Also, frequencies can be included as decision variables, which are bounded by a total number of available vehicles (Schmid 2014).

The PND problem involves decisions regarding infrastructure building of metro and rapid transit systems, namely, the construction of stations and tracks or corridors. Even though decisions regarding routes is not a primary concern in the context of this problem, they are taken into account due to their relevance regarding system performance. A typical way of addressing this problem is to choose a small number of routes, maximizing the coverage of a given demand between a set of fixed points, subject to a maximum available construction budget (Laporte et al. 2007). The binary variables s_{rv} and y_{re} state whether route r uses station $v \in \mathscr{V}$ and edge $e \in \mathscr{E}$ respectively. The passenger behavior is modeled with binary variables z_k and x_{ek} which state whether OD pair $k \in \mathscr{K}$ uses the public transportation network and whether it employs edge $e \in \mathscr{E}$ from that network, respectively. The formulation aims at the maximization of trips attracted to the public network, where the demand is split according to parameters which express the user cost of traversing each edge by using the public mode or the private one (typically, the car mode). An extension to this model considers the incremental building of the network across a set $\mathscr{T}$ of given periods (Marín and Jaramillo 2008). In this context, some problem data depend on the specific period $t \in \mathscr{T}$, namely, the OD matrix, construction costs, available budget and user cost within the public network. Clearly, in order to support multistage long-term planning, the dimensionality of the model is increased.

A different modeling approach for incremental building of the physical network proposes the design of single routes, which can be used as building block for obtaining a complete system made by different routes (Dufourd et al. 1996). In this case, decisions are the location of a single route, while maximizing population coverage under constraints of number of stations and inter-station spacing. A route is defined as a sequence of potential stations $s \in \mathscr{S}$ taken from a grid which represents a discretization of the study region. Each potential station s has coordinates in the Euclidean space, which are used to estimate its population catchment based in concentric geometrical shapes and the distance between the station and squares of the grid which intersect with the shape. This is a variation of covering-path like problems, which results in a non-linear integer mathematical program. Moreover, variations of this model consider the coverage of origin-destination trips instead of the maximization of population catchment (Laporte et al. 2005). This is done by replacing the original objective function by an expression which relates coverage areas of pairs of stations. Furthermore, this value is multiplied by a logit factor in order to determine the share of demand that is attracted by the public network, which is assumed to compete against a private mode. Construction costs are represented in these models by constraints on maximum route length and number of stations.

4 Solution Approaches

In this section, we present an overview of solution methods for public transportation network design problems, either related to models of Sect. 3 or to other ones which exhibit relevant algorithmic ideas. In a first level, methods are classified into mathematical programming and heuristic based ones, depending on whether they are based on an explicit mathematical formulation.

4.1 Mathematical Programming Based Methods

Several problems related to PT network design are formulated as mathematical programs, usually mixed integer linear ones (MILP). In most cases, small problem instances can be solved by using commercial MILP software developed by third parties. However, for larger instances some solution methods involve specific algorithmic developments. These methods are strongly determined by the mathematical formulation, since they exploit its properties. They can be classified into branch-and-bound-and-cut and decomposition methods.

4.1.1 Branch-and-Bound-and-Cut Methods

Problem (17.8)–(17.13) is solved by using branch and bound with three problem specific improvements: a relaxation obtained by aggregating variables x_{rk} across all routes $r \in \mathscr{R}_0$, cutting planes induced by constraints (17.10) and (17.11) in the relaxed problem, and upper and lower bounds derived by using the relaxed problem. It is worth noting that solutions of the relaxed problem ensure demand satisfaction by all lines collectively but they disregard the capacities of individual lines, therefore they cannot be easily transformed into feasible solutions of the original problem.

Moreover, problem (17.14)–(17.18) is solved firstly by applying a formulation strengthening through preprocessing, which involves coefficient reduction, variable reduction linked to the coefficient reduction and constraint reduction using dominance rules. Next, the branch and bound is enriched with cutting planes derived from constraints (17.15) and (17.16), several branching rules and a primal heuristic which builds a solution based in the resolution of the linear relaxation.

In order to find the set of efficient solutions for the multiobjective problem (17.37)–(17.43) the ϵ-constraint method is applied with respect to the second objective. This means that the second component of vector (17.37) is transformed into a constraint, which enables to find efficient solutions by varying its right-hand side. Moreover, the bilevel structure of the problem is eliminated by substitution of the lower-problem (17.40)–(17.43) by its optimality conditions. This can be done by combining duality, specific properties of the shortest-path problem and linearization techniques.

4.1.2 Decomposition Methods

Problem (17.19)–(17.25) is solved by a column generation approach, given the super-polynomial number of variables. In a first step, the linear relaxation is solved by iteratively pricing passenger and line path variables until no improvement is found. The pricing of passenger variables is a polynomial-time solvable shortest path problem. On the other hand, the pricing of line variables is a *NP*-hard maximum weighted path problem. In a second step, the algorithm builds an integer solution from the set of routes having nonzero frequencies in the optimal solution of the linear relaxation. This is done by a greedy procedure which deletes routes as long as all OD pairs are covered and the objective value decreases.

The PND problem of choosing a small number of routes while maximizing demand coverage can be solved by applying the Benders decomposition (Marín and Jaramillo 2009). The problem is partitioned into the master (which involves variables related to infrastructure building) and the sub-problem (which deals with passenger behavior). At each iteration of the algorithm, dual variables of the sub-problem define optimality or feasibility cuts which are added to the constraints of the master problem. Moreover, several extensions are introduced in order to improve the performance of the method, namely, separation of the sub-problem by OD pair, elimination of inactive cuts and specific shortest path algorithms to solve the sub-problem. These improvements allow for solving realistic size instances of the problem.

4.2 Heuristic Based Methods

We refer as heuristic methods for PT network design to those which are not driven by an explicit mathematical programming formulation. The algorithms presented apply to variants of models presented in Sect. 3. The objectives to be optimized can be, among others: user benefit maximization, operator cost minimization or total welfare maximization (Kepaptsoglou and Karlaftis 2009). Moreover, the heuristic methods for RND are classified into: (1) route generation and route selection, and (2) route set generation and improvement. The first approach considers the generation of single routes (route generation), which also can be used to compose a pool of candidate routes from which an optimal subset will be then selected (route selection). The second approach generates a complete solution in a first stage (route set generation), which can be then improved (route set improvement).

4.2.1 Route Generation and Selection

In the context of RND, the route generation and selection approach entails generating firstly a pool of many good candidate routes, from which the optimal (or best possible) subset is selected in a second stage. When generating the pool, usually the following criteria are taken into consideration: (1) the candidate routes should be good, both for users and operators, (2) each element of the pool has to fulfil some constraints which can be verified at route level individually, e.g., route length, duration and circuity, overlapping with existing routes, (3) a compromise between a small pool concentrated in few routes and a larger pool which provides more diversity should be managed. The usual way for generating candidate routes is based on shortest paths between node pairs of $\mathscr{G}$, which can include origins and destinations of OD pairs given by set $\mathscr{K}$ or all possible node pairs taken from $\mathscr{N} \times \mathscr{N}$. These routes are expected to provide a good level of service in terms of travel time from the users viewpoint. But since this pool could be very restrictive, additional routes are usually generated. To do this, different ideas can be applied: (1) taking a route generated from a shortest path P and generating additional similar routes by successively eliminating each edge from P and recomputing the shortest path, (2) generating k-shortest paths for every node pair of $\mathscr{G}$ (as we increase the value of k, a larger and more diverse pool can be obtained). Since routes generated from shortest paths could be biased towards the interest of users, alternative ways of generating routes biased towards operator's interest are taken into consideration, for instance, including in the pool routes generated by analyzing the concentration of demand flow in the arcs of $\mathscr{G}$ (Cipriani et al. 2012). To do that, a system-split like procedure is first run, which produces the aggregated flow from all OD pairs $k \in \mathscr{K}$ over each arc $a \in \mathscr{A}$. Then, routes are generated by selecting highly loaded arcs and adding links until specific termination criteria involving route constraints are met. So far, the candidate routes generated by these methods do not collectively ensure the fulfilment of global constraints at the route selection level, like demand coverage (in the topological sense) and demand satisfaction (in terms of capacity). This issue can be addressed by a model based pool generation (over minimal spanning trees) which ensures capacity fulfilment (Gattermann et al. 2016).

Heuristics based on route generation and selection involve a second phase where the best possible subset of routes is selected from the pool of candidate routes. This entails solving a set covering like problem, with a large number of variables. We identify two approaches to solve this problem heuristically for RND: (1) genetic algorithms based search, and (2) neighborhood based search. In the first group, usually the route identifiers are coded into a chromosome which can be of either fixed or variable length, thus allowing solutions with different number of routes. The individuals (sets of routes, i.e., solutions to RND) are then evolved using classical genetic operators like one point or two point crossover, and mutation. Note that crossing two individuals entails exchanging routes between solutions. Regarding neighborhood based search, a set of neighbors of a given solution to RND can be defined by replacing each route by one of its contiguous (similar) elements in the pool. Note that the structure of the routes defined during the pool generation does not change due to the search process. Moreover, since the pool does not necessary guarantee demand coverage of all demand OD pairs, the unsatisfied demand can be included in the objective function to penalize this fact.

4.2.2 Route Set Generation and Improvement

The route set generation approach produces a complete solution for RND. Usually, feasibility at both route and solution level is ensured. Most algorithms perform an incremental construction, which can be either biased or unbiased. In the first group, the main idea is to build some skeleton routes which are then enlarged by inserting nodes until the whole demand given by set $\mathscr{K}$ is covered. Skeletons are built by connecting high demand OD pairs, either enumerating and selecting the best sequence of intermediate nodes or computing shortest paths. Then, additional demand is covered by inserting nodes into the initial skeletons. However, the node insertion should discard cases where the resulting route becomes too large, circuitous or overloaded. The solutions generated by these methods are expected to be good by construction, however, they can be improved in a further stage. On the other hand, the unbiased approach aims at generating initial solutions which need to be improved in a second stage. In this case, the route set generation method should ensure diversity, while the route set improvement should ensure a comprehensive exploration of the search space. Usually, the construction is performed by selecting randomly an initial node and then adding randomly additional nodes. The solution should guarantee minimal levels of demand coverage and connectivity. To do that, usually all nodes of $\mathscr{G}$ should be reached by routes, and a reasonable number of route intersections (which enable transfers) should be ensured.

The route set improvement entails either modifying existing routes or generating new ones. We again identify two different approaches depending on the adoption of genetic or neighborhood search. In the first group, problem specific genetic operators can be applied to the initial solution, namely, add/delete arc, route merge, route break, route sprout and route crossover. Regarding neighborhood search, a typical approach applies simple arc add/delete operators to each route of the solution. A more complex neighborhood structure involves exploring alternative deviating paths from an initial one, which can be modified at given points (Zhao and Zeng 2008). It is worth noting that whatever the neighborhood structure is adopted, any method for escaping from local optima can be used, e.g., simulated annealing or tabu search.

A related methodology which falls within this category is the generation and improvement of a single route in the context of PND. This is done by considering the grid-based set of potential nodes and constructing either a random walk along one of the two diagonals of the square grid (Dufourd et al. 1996) or a greedy biased initial solution (Bruno et al. 2002). Then, local search is applied, where the neighborhood of the solution is obtained by moving one of its stations to a contiguous position in the grid.

4.2.3 Handling Specific Problem Features

Heuristic methods for PT network design often have to deal with two distinctive problem characteristics: (1) the multiobjective structure due to existence of conflicting objectives, and (2) the bilevel structure resulting from the passenger behavior model.

The treatment of multiple objectives is sometimes performed implicitly, where algorithms are conceived to balance the different objectives during solution construction (Baaj and Mahmassani 1995; Mauttone and Urquhart 2009a). In this case, the output is a single solution but, by changing appropriately some parameters, different trade-off solutions can be obtained. A different approach consists of solving heuristically a model which weights the different objectives into a single function (Pattnaik et al. 1998). By changing the weights, different trade-off solutions can be obtained. Finally, some other algorithms produce in a single run, an entire set of trade-off solutions (Israeli and Ceder 1995; Mauttone and Urquhart 2009b; Oliveira and Barbieri 2015). This is attained by means of specific operators and parameter settings.

In the context of heuristics, whenever a solution is changed due to local move or genetic operator, usually the passenger behavior model should be run in order to evaluate the system performance under the new conditions. This entails solving variants of the shortest path problem (17.1)–(17.3). In absence of capacity constraints, the computation is equivalent to solving $|\mathscr{K}|$ independent shortest path problems. But, if passengers are restrained by vehicle capacity, the resulting multicommodity flow problem is more difficult to solve. In this case, specific accelerating techniques can help to reduce computation time (Walteros et al. 2015). Although more complex and detailed passenger behavior models exist in the literature (Desaulniers and Hickman 2007), their complexity in PT network optimization models and algorithms must be kept bounded, given the strategic and tactical characteristics of the problems involved. In any case, the computational effort spent by calling the passenger behavior model is significant with respect to the overall execution time of PT network design algorithms.

5 Bibliographical Notes

Some relevant surveys are worth to be mentioned before discussing the specific literature on public transportation network design. In Schöbel (2012), the RND problem is discussed from a mathematical programming perspective, providing formalization for several concepts including the notion of user and operator oriented models. Kepaptsoglou and Karlaftis (2009) review models and algorithms for RND, proposing the classification of solution approaches into (1) candidate route generation and route configuration and (2) route construction and improvement. Laporte and Mesa (2015) review methodologies for PND, including the location of stations, design of a single route and of the entire network. In Farahani et al. (2013), an overview of methodologies for several urban transportation network design problems is provided, including both private and public modes. The study focuses in models and algorithms dealing with topological variables, presents a general bilevel formulation for the problems and identifies problem instances reported in the literature up to the year of publication. More recently, Iliopoulou et al. (2019) review metaheuristic approaches to RND, identifying relevant algorithmic aspects like route representation, repair and recombination.

Early work in public transportation network optimization consists of heuristics and it can be traced from Lampkin and Saalmans (1967), where the heuristic for route set construction based in skeletons and further node insertion is proposed. This method was later extended by Silman et al. (1974), who include a route deletion procedure and consider transfers between routes when computing demand coverage and travel time. Dubois et al. (1979) tackle both PND and RND problems for bus systems. Actually, the PND does not entail infrastructure building, instead it refers to selecting the set of streets which will be used by the bus routes, which reduces the size of the underlying network used as input in the RND problem. Unlike the methods mentioned above, which allow for generating a route set from an empty solution, the concept of route set improvement is developed by Mandl (1980). That author proposes a method that applies insertion and deletion of nodes in routes and interchange of parts between routes of an already existing solution. The route set construction based on skeletons is resumed by Baaj and Mahmassani (1995), who enrich the procedure for node selection and insertion. Also, these authors propose the idea of building skeletons based on k-shortest paths instead of the shortest one, in order to diversify the search during construction. Mauttone and Urquhart (2009a) modify the node insertion procedure by proposing a pair insertion which seeks to cover high demand OD pairs directly. More recently, Islam et al. (2019) propose a greedy algorithm inspired by the work of Baaj and Mahmassani (1995), which builds routes between high demand OD pairs by appending shortest route segments that consider both travel cost and demand coverage.

The first studies which apply metaheuristics to RND consist of genetic algorithms. Pattnaik et al. (1998) solve the route selection problem by encoding the identifiers of routes taken from a predefined pool, into a chromosome which can be of either fixed or variable length. Further developments propose extensions to include frequency encoding (Tom and Mohan 2003) and parallel implementations (Agrawal and Tom 2004). Another relevant application of metaheuristics to route selection in the context of RND is due to Fan and Machemehl (2006), who apply simulated annealing to select the best subset of routes from a predefined pool of candidates. Fan and Mumford (2010) apply simulated annealing for searching on the space of route structure. A different application of genetic algorithms to RND is proposed by Ngamchai and Lovell (2003) for the route set improvement problem, implementing several problem specific operators which modify the structure of the routes of the initial solution. More recent applications of metaheuristics involve the use of ant and bee colony optimization (Nikolic and Teodorovic 2014; Szeto and Jiang 2014; Yu et al. 2012) and particle swarm optimization (Kechagiopoulos and Beligiannis 2014), either for construction or for improvement of solutions.

Mathematical programming approaches are more recent and they were firstly applied to passenger rail transportation. Regarding RND, Bussieck et al. (1997) proposed the user oriented model and the corresponding solution method based in relaxations, cutting planes and bounds. The operator oriented model is due to Claessens et al. (1998), who also present complexity results and a solution method based in reformulation and lower bounding. Their work is resumed by Goosens et al. (2004), who propose a solution method based on formulation strengthening and branch and cut. Goosens et al. (2006) extend the model to allow lines with different stopping patterns. While the studies mentioned above consider fixed passenger behavior, Borndörfer et al. (2007) introduce a path-based model which generates the routes. They also present complexity results and a solution method based on column generation. Other formulations which include explicit modeling of passenger behavior have been proposed by Guan et al. (2006) and Cancela et al. (2015). Finally, other relevant work include the study of Schöbel and Scholl (2006), who proposed the expanded network to account for transfers, previously adopted by Spiess and Florian (1989) and later improved by Goerigk and Schmidt (2017). Regarding PND problems, Laporte et al. (2007) propose a base formulation, which is then extended by Marín and Jaramillo (2008). The single route location problem is due to Dufourd et al. (1996), then extended by Laporte et al. (2005). Latest developments in this line are due to Gutiérrez-Jarpa et al. (2017).

We should mention several studies which are relevant due to the treatment of particular aspects of public transportation network design problems. Lee and Vuchic (2005) model elastic demand by allowing a variable share of public transportation demand from a given fixed overall demand and they study the influence of several parameters over the resulting networks. Szeto and Jiang (2014) use information from the mathematical formulation to reduce the number of calls to the passenger behavior model in the context of a metaheuristic solving method. Bagloee and Ceder

(2011) handle large size networks comprising up to 13,487 nodes, 52,742 arcs and 142,041 OD pairs. Mumford (2013) makes an effort to establish a set of benchmark instances for public transportation network optimization.

Finally, it is worth noting the existence of a complementary stream of publications dealing with public transportation network design from a structural point of view. Laporte et al. (2000) identify several network structures (star, cartwheel, triangle and grid) and evaluate their effectiveness according to indexes that represent the interest of passengers. In this line, more recently Fielbaum et al. (2018) apply some of the models discussed in this chapter to cities with different structures (monocentric, polycentric and dispersed) paying attention to the role of transfers, thus, filling the gap between the different streams of research on the same topic.

6 Conclusions and Perspectives

Public transportation network design problems have been studied since more than five decades ago. Several problem aspects are well explained by the existing models. Moreover, several solution methods have been tested and documented, constituting a rich basis for developing new ones. In the following, we identify current challenges and future perspectives of this area of research.

Mathematical programming approaches face the challenge of solving huge mixed integer linear problems (MILP). The small city used as test case for RND in (Borndörfer et al. 2007) seems to establish the limit on the size of solvable instances using this approach. Newer MILP solvers should be evaluated regarding exact resolution, even considering formulations which incorporate additional problem features like transfers (Schöbel and Scholl 2006), waiting time (Cancela et al. 2015) and capacities (Goerigk and Schmidt 2017).

Metaheuristics have shown to be the most effective methods for solving medium and large-sized problem instances. Nevertheless, they face the main challenge of minimizing the calls to the passenger behavior model, which is the most critical algorithmic component of the overall solution methods. Techniques like the one proposed by Szeto and Jiang (2014), which attempts to discard unnecessary solution evaluations should be explored. Regarding experimental evaluation of accuracy of metaheuristics, the lack of a well established set of benchmark instances with reference values is a weakness of the field. This situation needs to be addressed, a remarkable contribution in this sense is the work of Mumford (2013). Recent works have shown progress on the field (Iliopoulou et al. 2019), mainly in the computational aspect of the methods that tackle the combinatorial complexity of the problem, where more elaborated experiments are conducted regarding parameter tuning, benchmark and reproducibility.

The modeling of passenger behavior under vehicle capacity constraints is a very relevant aspect of public transportation network design models. The capacitated user equilibrium modeled in Borndörfer et al. (2007) assumes that some users are willing to choose longer paths with respect to other users traveling from the same origin to the same destination. This could be questionable in the context of real systems. A different approach is adopted in the bilevel model of Goerigk and Schmidt (2017), which ensures sufficient capacity for all users. However, this entails taking into account two other issues: (1) solving the more complex bilevel formulation, and (2) discussing whether real systems are able to implement these solutions, mainly due to high frequency requirements. While the design of uncongested public transportation systems can be a reasonable goal, sometimes it is necessary to recognize that congestion plays a role in network design due to limitations on the available resources. The effect of congestion over passengers (especially over waiting time and route choice) leads to complex models (Gendreau 1984) which have been successfully addressed at the descriptive level (Cepeda et al. 2006), i.e., models that represent passenger behavior given a fixed set of routes. However, normative models for congested public transportation network design are much more difficult to solve, since they turn into mathematical programs with equilibrium constraints (Colson et al. 2007).

Finally, other aspects of public transportation network design like stochastic demand (An and Lo 2016) and integration of stages (Canca et al. 2017) also deserve attention, due to their relevance at the practical level as for the challenge they pose at both modeling and algorithmic levels. In fact, these issues have been recently approached by the research community, as part of the effort to build models which are able to better represent real life systems.

References

Agrawal, J., & Tom, V. M. (2004). Transit route network design using parallel genetic algorithm. *Journal of Computing in Civil Engineering, 18*(3), 248–256.

Ahuja, R. K., Magnanti, T. L., & Orlin, J. B. (1993). *Network flows: Theory, algorithms, and applications*. Englewood Cliffs: Prentice-Hall

An, K., & Lo, H. K. (2016). Two-phase stochastic program for transit network design under demand uncertainty. *Transportation Research Part B: Methodological, 84*, 157–181.

Baaj, M. H., & Mahmassani, H. S. (1995). Hybrid route generation heuristic algorithm for the design of transit networks. *Transportation Research Part C: Emerging Technologies, 3*(1), 31–50.

Bagloee, S. A., & Ceder, A. (2011). Transit-network design methodology for actual-size road networks. *Transportation Research Part B: Methodological, 45*(10), 1787–1804.

Bard, J. F. (1998). *Practical bilevel optimization, algorithms and applications*. Berlin: Springer.

Borndörfer, R., Grötschel, M., & Pfetsch, M. E. (2007). Column-generation approach to line planning in public transport. *Transportation Science, 41*(1), 123–132.

Bruno, G., Gendreau, M., & Laporte, G. (2002). A heuristic for the location of a rapid transit line. *Computers and Operations Research, 29*(1), 1–12.

Bussieck, M. R., Kreuzer, P., & Zimmermann, U. T. (1997). Optimal lines for railway systems. *European Journal of Operational Research, 96*(1), 54–63.

Canca, D., De-Los-Santos, A., Laporte, G., & Mesa, J. A. (2017). An adaptive neighborhood search metaheuristic for the integrated railway rapid transit network design and line planning problem. *Computers and Operations Research, 78*, 1–14.

Cancela, H., Mauttone, A., & Urquhart, M. E. (2015). Mathematical programming formulations for transit network design. *Transportation Research Part B: Methodological, 77*, 17–37.

Ceder, A., & Wilson, N. H. M. (1986). Bus network design. *Transportation Research Part B: Methodological, 20*(4), 331–344.

Cepeda, M., Cominetti, R., & Florian, M. (2006). A frequency-based assignment model for congested transit networks with strict capacity constraints: Characterization and computation of equilibria. *Transportation Research Part B: Methodological, 40*(6), 437–459.

Cipriani, E., Gori, S., & Petrelli, M. (2012). Transit network design: A procedure and an application to a large urban area. *Transportation Research Part C: Emerging Technologies, 20*(1), 3–14.

Claessens, M. T., van Dijk, N. M., & Zwaneveld, P. J. (1998). Cost optimal allocation of rail passenger lines. *European Journal of Operational Research, 110*(3), 474–489.

Colson, B., Marcotte, P., & Savard, G. (2007). An overview of bilevel optimization. *Annals of Operations Research, 153*(1), 235–256.

Correa, J. R., Schulz, A. S., & Stier-Moses, N. E. (2004). Selfish routing in capacitated networks. *Mathematics of Operations Research, 29*(4), 961–976.

Desaulniers, G., & Hickman, M. D. (2007). Public transit. In G. Laporte & C. Barnhart (Eds.), *Transportation, handbooks in operations research and management science* (Vol. 14, pp. 69–127). Amsterdam: Elsevier

Dubois, D., Bel, G., & Llibre, M. (1979). A set of methods in transportation network synthesis and analysis. *Journal of the Operational Research Society, 30*(9), 797–808.

Dufourd, H., Gendreau, M., & Laporte, G. (1996). Locating a transit line using tabu search. *Location Science, 4*(12), 1–19.

Ehrgott, M. (2005). *Multicriteria optimization*. Berlin: Springer.

Fan, W., & Machemehl, R. B. (2006). Using a simulated annealing algorithm to solve the transit route network design problem. *Journal of Transportation Engineering, 132*(2), 122–132.

Fan, L., & Mumford, C. (2010). A metaheuristic approach to the urban transit routing problem. *Journal of Heuristics, 16*(3), 353–372.

Farahani, R., Miandoabchi, E., Szeto, W. Y., & Rashidi, H. (2013). A review of urban transportation network design problems. *European Journal of Operational Research, 229*(2), 281–302.

Fielbaum, A., Jara-Díaz, S., & Gschwender, A. (2018). Transit line structures in a general parametric city: The role of heuristics. *Transportation Science, 52*(5), 1092–1105.

Gattermann, P., Harbering, J., & Schöbel, A. (2016). Line pool generation. *Public Transport, 9*(1), 7–32.

Gendreau, M. (1984). *Etude approfondie d'un modèle d'équilibre pour l'afectation de passagers dans les réseaux de transports en commun*. Ph.d. thesis, Université de Montréal, Publication CRT-384.

Goerigk, M., & Schmidt, M. (2017). Line planning with user-optimal route choice. *European Journal of Operational Research, 259*(2), 424–436.

Goossens, J.-W., van Hoesel, S., & Kroon, L. (2004). A branch-and-cut approach for solving railway line-planning problems. *Transportation Science, 38*(3), 379–393.

Goossens, J.-W., van Hoesel, S., & Kroon, L. (2006). On solving multi-type railway line planning problems. *European Journal of Operational Research, 168*(2), 403–424.

Guan, F., Yang, H., & Wirasinghe, S. C. (2006). Simultaneous optimization of transit line configuration and passenger line assignment. *Transportation Research Part B: Methodological, 40*(10), 885–902.

Gutiérrez-Jarpa, G., Laporte, G., Marianov, V., & Moccia, L. (2017). Multi-objective rapid transit network design with modal competition: The case of Concepción, Chile. *Computers and Operations Research, 78*, 27–43.

Heyken Soares, P., Mumford, C., Amponsah, K., & Mao, Y. (2019). An adaptive scaled network for public transport route optimisation. *Public Transport, 11*, 379–412.

Iliopoulou, C., Kepaptsoglou, K., & Vlahogianni, E. (2019). Metaheuristics for the transit route network design problem: A review and comparative analysis. *Public Transport, 11*, 487–521.

Islam, K., Moosa, I., Mobin, J., Nayeem, M., & Rahman, M. (2019). A heuristic aided Stochastic Beam Search algorithm for solving the transit network design problem. *Swarm and Evolutionary Computation, 46*, 154–170.

Israeli, Y., & Ceder, A. (1995). Transit route design using scheduling and multiobjective programming techniques. In J. Daduna, I. Branco, & J. P. Paixão (Eds.), *Computer-Aided Transit Scheduling: Proceedings of the Sixth International Workshop on Computer-Aided Scheduling of Public Transport. Lecture Notes in Economics and Mathematical Systems* (pp. 56–75). Berlin: Springer.

Johnson, D. S., Lenstra, J. K., & Kan, A. H. G. R. (1978). The complexity of the network design problem. *Networks, 8*, 279–285.

Kechagiopoulos, P., & Beligiannis, G. (2014). Solving the urban transit routing problem using a particle swarm optimization based algorithm. *Applied Soft Computing, 21*, 654–676.

Kepaptsoglou, K., & Karlaftis, M. (2009). Transit route network design problem: Review. *Journal of Transportation Engineering, 135*(8), 491–505.

Lampkin, W., & Saalmans, P. D. (1967). The design of routes, service frequencies, and schedules for a municipal bus undertaking: A case study. *Operational Research Quarterly, 18*(4), 375–397.

Laporte, G., Marín, A., Mesa, J. A., & Ortega, F. (2007). An integrated methodology for the Rapid Transit Network Design Problem. In F. Geraets, L. Kroon, A. Schöbel, D. Wagner, & C. Zaroliagis (Eds.), *International Dagstuhl Workshop, Dagstuhl Castle, Germany, June 20-25, 2004, 4th International Workshop, ATMOS 2004, Bergen, Norway, September 16–17, 2004, Revised Selected Papers* (pp. 187–199). Berlin: Springer.

Laporte, G., & Mesa, J. A. (2015). The design of rapid transit networks. In G. Laporte, S. Nickel, & F. Saldanha da Gama (Eds.), *Location science* (pp. 581–594). Cham: Springer.

Laporte, G., Mesa, J. A., & Ortega, F. (2000). Optimization methods for the planning of rapid transit systems. *European Journal of Operational Research, 122*(1), 1–10.

Laporte, G., Ortega, F., Mesa, J. A., & Sevillano, I. (2005). Maximizing trip coverage in the location of a single rapid transit alignment. *Annals of Operations Research, 136*, 49–63.

Lee, Y. J., & Vuchic, V. (2005). Transit network design with variable demand. *Journal of Transportation Engineering, 131*(1), 1–10.

Mandl, C. E. (1980). Evaluation and optimization of urban public transportation networks. *European Journal of Operational Research, 5*(6), 396–404.

Marín, A., & Jaramillo, P. (2008). Urban rapid transit network capacity expansion. *European Journal of Operational Research, 191*(1), 45–60.

Marín, A., & Jaramillo, P. (2009). Urban rapid transit network design: Accelerated Benders decomposition. *Annals of Operations Research, 169*(1), 35–53.

Mauttone, A., & Urquhart, M. E. (2009a). A route set construction algorithm for the transit network design problem. *Computers and Operations Research, 36*(8), 2440–2449.

Mauttone, A., & Urquhart, M. E. (2009b). A multi-objective metaheuristic approach for the transit network design problem. *Public Transport, 1*(4), 253–273.

Mumford, C. (2013). New heuristic and evolutionary operators for the multi-objective urban transit routing problem. In *Proceedings of the 2013 IEEE Congress on Evolutionary Computation* (pp. 939–946).

Ngamchai, S., & Lovell, D. (2003). Optimal time transfer in bus transit route network design using a genetic algorithm. *Journal of Transportation Engineering, 129*(5), 510–521.

Nikolic, M., & Teodorovic, D. (2014). A simultaneous transit network design and frequency setting: Computing with bees. *Expert Systems with Applications, 41*(16), 7200–7209.

Oliveira, R., & Barbieri, C. (2015). Efficient transit network design and frequencies setting multi-objective optimization by alternating objective genetic algorithm. *Transportation Research Part B: Methodological, 81*(2), 355–376.

Pattnaik, S. B., Mohan, S., & Tom, V. M. (1998). Urban bus transit route network design using genetic algorithm. *Journal of Transportation Engineering, 124*(4), 368–375.

Schmid, V. (2014). Hybrid large neighborhood search for the bus rapid transit route design problem. *European Journal of Operational Research, 238*(2), 427–437.

Schöbel, A. (2012). Line planning in public transportation: Models and methods. *OR Spectrum, 34*(3), 491–510.

Schöbel, A., & Scholl, S. (2006). Line planning with minimal traveling time. In L. G. Kroon & R. H. Möhring (Eds.), *5th Workshop on Algorithmic Methods and Models for Optimization of Railways (ATMOS'05)*. Wadern: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik

Sheffi, Y. (1985). *Urban transportation networks: Equilibrium Analysis With Mathematical Programming Methods*. Englewood Cliffs: Prentice-Hall

Silman, L. A., Barziliy, Z., & Passy, U. (1974). Planning the route system for urban buses. *Computers and Operations Research, 1*(2), 201–211.

Spiess, H., & Florian, M. (1989). Optimal strategies: A new assignment model for transit networks. *Transportation Research Part B: Methodological, 23*(2), 83–102.

Szeto, W. Y., & Jiang, Y. (2014). Transit route and frequency design: Bi-level modeling and hybrid artificial bee colony algorithm approach. *Transportation Research Part B: Methodological, 67*, 235–263.

Teodorovic, D., & Janic, M. (2016). *Transportation engineering, theory, practice and modeling*. Oxford: Butterworth-Heinemann

Tom, V. M., & Mohan, S. (2003). Transit route network design using frequency coded genetic algorithm. *Journal of Transportation Engineering, 129*(2), 186–195.

Vuchic, V. R. (2007). *Urban transit, systems and technology*. New York: Wiley.

Walteros, J. L., Medaglia, A. L., & Riaño, G. (2015). Hybrid algorithm for route design on bus rapid transit systems. *Transportation Science, 49*(1), 66–84.

Yu, B., Yang, Z.-Z., Jin, P.-H., Wu, S.-H., & Yao, B.-Z. (2012). Transit route network design-maximizing direct and transfer demand density. *Transportation Research Part C: Emerging Technologies, 22*, 58–75.

Zhao, F., & Zeng, X. (2008). Optimization of transit route network, vehicle headways and timetables for large-scale transit networks. *European Journal of Operational Research, 186*(2), 841–855.

Chapter 18
Hub Network Design

Ivan Contreras

1 Introduction

Hub networks are frequently employed in many transportation, telecommunication and computer systems to efficiently route commodities between many origins and destinations. A distinguishing feature of hub networks is the use of transshipment, consolidation, or sorting points for commodities, called *hub facilities*, to connect a large number of origin/destination (O/D) pairs by using a small number of links. Commodities having the same origin but different destinations are consolidated when routed to the hubs and are then combined with other commodities having different origins but the same destination. The use of hub facilities helps centralize commodity handling and sorting operations, reduce set-up costs, and achieve economies of scale on routing costs through the consolidation of flows. Hub networks can be seen as hierarchical networks which, in their most basic form, contain two levels: an *access-level* network connecting O/D nodes to hubs, and a *hub-level* network connecting hub nodes between them. The design of hub networks involves selecting nodes to place hub facilities, determining the arcs to connect O/D nodes and hubs, and selecting the paths to route commodities.

Hub network design problems (HNDPs) lie at the heart of network design planning in transportation and telecommunication systems. Application areas of HNDPs in transportation include air freight and passenger travel, postal delivery, express package delivery, trucking, liner shipping, public transportation, and rapid transit systems. Demand corresponds to passengers, mail, express packages, or goods carried by airplanes, trucks, trains, or vessels moved on physical networks such as roads and railways or through the air or water. Hub facilities are sorting

I. Contreras (✉)
Concordia University and Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT), Montreal, QC, Canada
e-mail: icontrer@encs.concordia.ca; ivan.contreras@concordia.ca

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_18

centers or transportation terminals in which one or more transportation modes interact. Hubs are used as intermediate facilities to consolidate flows, perform an activity to commodities (i.e., sort, assemble, label), or transfer them to other modes of transportation. Consolidation of flows at hubs enables economies of scale on transportation costs, not only on the routing of flows between hubs but also between O/D nodes and hubs. Sometimes hubs are required route commodities. Some other times hubs are not required but desirable for economical reasons.

Applications of HNDPs in telecommunications arise in the design of distributed data networks, where commodities correspond to electronic data that are routed over a variety of physical links such as co-axial cables and fiber optic links or through the air via satellite channels and microwave links. Hub facilities correspond to hardware such as switches, concentrators, and multiplexors which help to provide efficient connections between tributary and backbone networks. Large set-up costs for hub facilities and communication links, in combination with economies of scale in data transmissions and network utilization, motivate the use of hub-and-spoke architectures.

HNDPs constitute a challenging class of network optimization problems involving two types of design decisions: (1) the location of hub facilities at nodes of an underlying network, and (2) the activation of various classes of links to connect origin, destinations, and hubs. Given the inherent complexity of the interaction between these two types of decisions, HNDPs were first studied from a facility location perspective. In particular, the so-called *hub location problems* (HLPs) focus on the interaction between hub facilities and consider the location of hubs as the key decision. Most HLPs use a set of assumptions that simplify the design and routing decisions to the point of being completely determined by the allocation decisions of O/D nodes to hubs. When such simplifying assumptions are relaxed, HNDPs are more closely related to *multicommodity network design problems* (MNDPs). In fact, HNDPs are a particular class of MNDPs in which node selection decisions are taken into account. A specific class of such general HNPDs, denoted as *hub arc location problems* (HALPs), have also been studied in which a set of hub arcs, and their associated hub nodes, need to be selected. In this case, the modeling of O/D paths become more involved as the allocation of nodes to hubs does no longer determine the routing of flow through the hub network. HALPs retain some assumptions of HLPs, specially the ones regarding the design of the access-level network. The rich variety of applications has also given rise to HNDPs with specific hub network topologies and to more general models involving design decisions on both hub and access levels as well as additional node selection decisions.

This chapter studies HNPDs from a network design perspective. We focus on the role network design and routing decisions play in the formulation and solution of various classes of HNDPs. Section 2 starts with some preliminaries, including the key features of hub networks, the types of decisions that can be taken into account, and how these decisions interact between them. We also describe commonly considered assumptions and properties of HNDPs and how these impact their formulation. In particular, Sect. 3 introduces different formulations for various classes of HLPs problems considering three allocation patterns: multiple, single, and

r-allocation. Section 4 presents more complex HNDPs such as HALPs and other problems with specific hub network topologies such as tree-star, start-star, ring-star, and hub line networks. For all these classes of problems, we highlight their most relevant applications and describe some formulations which have been developed and exploited in combination with decomposition methods to solve them. Section 5 provides a historical review of key references on HNDPs together with some of the most significant milestones in the field. Conclusions and perspectives follow in Sect. 6.

2 Preliminaries

A generic hub network design problem can be described as follows. Consider a complete graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ is the set of nodes representing the origins and destinations of flows as well as the set of potential hub locations, and $\mathcal{E}$ is the set of edges. For each node pair (i, j), let $W_{ij} \geq 0$ and $d_{ij} \geq 0$ denote the amount of flow to be routed and the distance, respectively, from the origin $i \in \mathcal{N}$ to the destination $j \in \mathcal{N}$. For each node $i \in \mathcal{N}$, f_i is the fixed set-up cost for locating a hub, whereas for each $e \in \mathcal{E}$, g_e denotes the fixed set-up cost for activating an (undirected) hub arc. A hub arc $e = (i, j) \in \mathcal{E}$ connects two different hub nodes i and j and has a unit flow cost of αd_{ij}. The parameter α $(0 \leq \alpha \leq 1)$ is used as a discount factor to provide reduced unit flow costs on hub arcs to reflect economies of scale resulting from consolidation of flows between hubs. The unit flow cost between O/D pairs is given by the length of the path between the origin and destination nodes in the solution network. Each O/D path has a *collection* leg from the origin node to the first hub, possibly a *transfer* leg between the first and the last hubs, and a *distribution* leg from the last hub to the destination node.

Depending on the assumptions and considered application, the solution network of a HNDP consists of up to four types of arcs: (1) *hub arcs* connecting two hubs with a discounted flow cost, (2) *bridge arcs* connecting also two hub nodes but without benefiting from the reduced unit flow cost of a hub arc, (3) *access arcs* connecting non-hub nodes and hubs, and (4) *direct arcs* connecting two non-hub nodes. A generic HNDP consists of locating a set of hub facilities, activating a set of arcs, and of determining the routing of flows through the hub network, with the objective of minimizing the total set-up and flow cost.

HLPs are a class of HNDPs which have been most studied in the literature. They focus on the location of a set of hub facilities and the assignment of O/D nodes to these facilities. Arc selection and routing decisions are mainly determined by the assumptions made on the cost structure and the assignment pattern. In particular, there are four assumptions underlying most HLPs: (1) commodities have to be routed via a set of hubs, (2) hub, access and bridge arcs have no set-up cost, (3) the discount factor α is the same for all hub arcs and does not depend on the amount of flow routed on each hub arc, (4) distances d_{ij} satisfy the triangle inequality. The following properties are a direct consequence of these assumptions:

- *O/D paths with hubs*: Assumption 1 prohibits direct connections between O/D nodes that are not hubs and hence, O/D paths must include at least one hub node. Note that this assumption is rather mild, as it is always possible to add a dummy hub and associated flow costs to represent direct connections between non-hub nodes.
- *Fully-interconnected hubs*: Assumption 2 allows hubs to be interconnected at no extra cost and, together with Assumptions 3 and 4, an important resulting property is that the set of hub arcs define a complete subgraph on the set of hub nodes. As a consequence, hub arc selection decisions become trivial once the location of hub nodes is known.
- *one-hub-arc O/D paths*: Another important property obtained when combining all assumptions is that O/D paths contain at least one and at most two hubs. However, it is important to note that whenever Assumption 2 or 4 are not satisfied, paths may contain more than two hubs and more than one hub arc.

The above properties do not only simplify the network design decisions in HLPs, as they are completely determined by the location and assignment decisions, but most importantly, they significantly reduce the number of O/D paths that need to be considered on a hub network. In HLPs, O/D paths include either a single hub node and no hub arc, or two hub nodes and a single hub arc. Moreover, because of Assumptions 2 and 4, each collection and distribution leg, if present, contains only one access arc. O/D paths are thus of the form (i, k, m, j), where $(k, m) \in \mathscr{N} \times \mathscr{N}$ is the ordered pair of hubs to which i and j are allocated, respectively. The flow cost of routing W_{ij} along the path (i, k, m, j) is then given by $W_{ij}\left(\chi d_{ik} + \alpha d_{km} + \delta d_{mj}\right)$, where χ, α, and δ represent the collection, transfer and distribution costs along the path. To reflect economies of scale between hubs, we assume that $\alpha < \chi$ and $\alpha < \delta$. Note that these paths contain one, two or at most three arcs, depending on the number of visited hubs and on the function of origins and destinations (i.e., hub or non-hub nodes). As a consequence, there are only $O(n^2)$ paths for each O/D pair. As we will show in Sect. 3, this allows the development of tight *path-based formulations* with $O(n^4)$ variables that explicitly consider all these paths and for some allocation patterns, they do not even require the use of flow conservation constraints.

In the case of more general HNDPs that do not satisfy some of the above mentioned assumptions, the modeling of O/D paths becomes more involved given that hub nodes are not necessarily fully interconnected and due to the presence of bridge arcs. O/D paths may contain more than three arcs and visit more than two hub nodes. The transfer leg can use several bridge and hub arcs, depending on whether additional assumptions on the structure of O/D paths are considered or not. This means that a much larger number of O/D paths exist. In fact, for the case of a complete graph the number of paths between all pairs of nodes is given by $\sum_{i=0}^{n-2}(n-2)!/(n-2-i)!$ As a consequence, path-based formulations for HNDP would have up to $O(n^{n-2})$ variables. Flow conservation constraints are now needed when extending arc-based formulations of HLPs which contain only $O(n^4)$

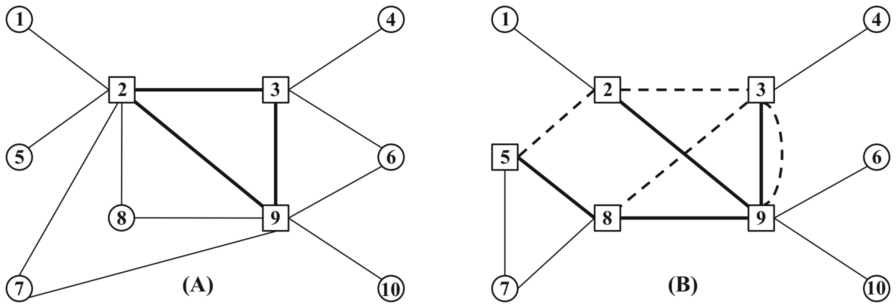


Fig. 18.1 Solution network of a hub location problem (**a**) and a hub network design problem (**b**)

variables. In Sect. 4, we highlight the added complexity in formulating and solving HNDPs where non-trivial arc selection and routing decisions need to be made.

Figure 18.1a shows an example of a solution network of a HLP in which different structures on O/D paths arise (squares represent hub nodes and circles represent non-hub nodes). The path $(5, 8, 3, 4)$ is a two-hub path formed by the access arcs $(5, 8)$, $(4, 3)$ and the hub arc $(8, 3)$. The path $(9, 9, 2, 1)$ is also a two-hub path but containing only the access arc $(1, 2)$ and the hub arc $(2, 9)$. The path $(3, 3, 9, 9)$ is yet another two-hub path formed only by the hub arc $(3, 9)$. The path $(4, 3, 3, 6)$ is a one-hub path containing only the access arcs $(4, 3)$ and $(6, 3)$. The path $(5, 8, 8, 8)$ is also a one-hub path containing the single access arc $(5, 8)$.

Figure 18.1b shows an example of a solution network of a more general HNDP in which different structures on O/D paths arise (dashed lines represent bridge arcs). The path $(5, 8, 9, 6)$ is a three-hub path formed by the bridge arc $(5, 8)$, the hub arc $(8, 9)$, and the access arc $(9, 6)$. The path $(1, 2, 8, 9, 3, 4)$ is a four-hub path containing the access arcs $(1, 2)$, $(4, 3)$ and the hub arcs $(2, 8)$, $(8, 9)$, and $(9, 3)$.

3 Hub Location Problems

HLPs focus on the location of hub facilities and the assignment of O/D nodes to open hubs. At the hub-level network, hub arc selection decisions are completely determined by the location of the hubs, given that they are full-interconnected with hub arcs. At the access-level network, arc selection decisions are given by the allocation of O/D nodes to hubs. There are three possible allocation strategies: multiple assignments, single assignments, and r-allocation. In the case of HLPs in which there is no set-up cost for the activation of access arcs, once the hub locations are known, the flow cost is minimized by finding a shortest path on the network induced by the selected hubs for each O/D pair, resulting in a multiple allocation pattern of O/D nodes to hubs. That is, a O/D node may be directly connected to more than one hub facility. A multiple assignment pattern simplifies the routing decisions and provides greater flexibility on hub networks, allowing lower flow cost solutions. However, they may considerably increase the network design cost as a larger number of access links must be activated. Applications in which it would be reasonable to consider multiple assignments arise mainly in transportation, in particular in air freight and passenger travel, public transportation, and rapid transit systems. In these cases, access arcs either do not correspond to physical links or they are associated with existing physical infrastructure (i.e., roads or highways) and hence, there is no set-up cost associated with them.

In a single assignments strategy, each O/D node must be connected to exactly one hub facility. All commodities with the same origin (or destination) are thus routed via the same access arc. Applications of a single assignment strategy arise in telecommunications, where access arcs correspond to physical links having significant set-up costs which need to be installed to provide connection and communication services to terminal nodes. Other applications arise in transportation, in particular in express package and postal delivery where commodities are usually consolidated at O/D nodes to be sent to the same sorting facility. Finally, in an r-allocation strategy each O/D node can be connected to at most r hubs. This strategy generalizes both single and multiple assignment strategies and, at the same time, provides the flexibility of allowing nodes to be allocated to two or more hubs while keeping some control on the number of access arcs on the solution network. In what follows, we describe the most relevant formulations that have been introduced to model each of the allocation strategies. We also point out to the most relevant solution algorithms developed for each of these classes of problems.

3.1 Multiple Assignments

We can use the so-called *flow-based formulations* to model HLPs with multiple assignments. They use continuous variables to determine the amount of flow routed on a particular arc originated at a given node. In the case of multiple assignments, we need three sets of flow variables to model the collection, transfer, and distribution legs in an O/D path. In particular, for the collection leg we define the continuous variables $U_{ik}, i, k \in \mathcal{N}$, equal to the amount of flow from origin node i sent directly to hub k via access arc (i, k). For the transfer leg, let Y_{ikm}, $i, j, k \in \mathcal{N}$, be equal to the amount of flow originated at node i and passing through hub arc (k, m). Finally, for the distribution leg let X_{ijm}, $i, j, m \in \mathcal{N}$, be equal to the amount of flow from origin i sent from hub m directly to destination j via access arc (m, j). We also define binary location variables z_i, $i \in \mathcal{N}$, equal to 1 if and only if a hub is located at node i. Using these sets of decision variables, we can formulate HLPs with multiple assignments as follows:

$$\text{minimize} \quad \sum_{k\in\mathcal{N}} f_k z_k + \sum_{i,k\in\mathcal{N}} \chi d_{ik} U_{ik} + \sum_{i,k,m\in\mathcal{N}} \alpha d_{km} Y_{ikm} + \sum_{i,j,m\in\mathcal{N}} \delta d_{mj} X_{ijm}$$

subject to
$$\sum_{k\in\mathcal{N}} U_{ik} = O_i \qquad i \in \mathcal{N} \tag{18.1}$$
$$\sum_{m\in\mathcal{N}} X_{ijm} = W_{ij} \qquad i, j \in \mathcal{N} \tag{18.2}$$
$$U_{ik} + \sum_{m\in\mathcal{N}} Y_{imk} = \sum_{m\in\mathcal{N}} Y_{ikm} + \sum_{j\in\mathcal{N}} X_{ijk} \qquad i, k \in \mathcal{N} \tag{18.3}$$
$$U_{ik} \le O_i z_k \qquad i, k \in \mathcal{N} \tag{18.4}$$
$$X_{ijm} \le W_{ij} z_m \qquad i, j, m \in \mathcal{N} \tag{18.5}$$
$$U_{ik}, Y_{ijk}, X_{ijk} \ge 0 \qquad i, j, k \in \mathcal{N} \tag{18.6}$$
$$z_k \in \{0, 1\} \qquad k \in \mathcal{N}. \tag{18.7}$$

Constraints (18.1)–(18.3) correspond to the flow conservation equations for a network flow problem for each origin node i. In particular, for each node $i \in \mathcal{N}$ there is a network with $2n + 1$ nodes. The first node is the source with a supply of O_i and then, there are n transshipment nodes, one for each possible hub node $k \in \mathcal{N}$. Finally, the demand at each of the n destination nodes j is given as W_{ij}. Constraints (18.4)–(18.5) ensure that flows are routed via open hubs. The above formulation contains $O(n^3)$ variables and $O(n^3)$ constraints. If the flow requirements are symmetric, i.e., $W_{ij} = W_{ji}$, $\forall i, j \in \mathcal{N}$, and if the collection and distribution cots are equal ($\chi = \delta$), then the U_{ik} variables can be eliminated from the formulation by using:

$$U_{ik} = \sum_{j\in\mathcal{N}} X_{jik} \qquad \forall i, k \in \mathcal{N}.$$

Arc-based formulations can also be adapted for the case of HLPs with multiple assignments. For each $i, j, k, m \in \mathcal{N}$, we define binary variables x_{ijkm} equal to 1 if and only if the flow originated at i and destination j is routed via hub arc (k, m). Using the same set of location variables z_i in combination with the arc variables x_{ijkm}, the problem can be stated as follows:

minimize
$$\sum_{k\in\mathcal{N}} f_k z_k + \sum_{i,j,k,m\in\mathcal{N}} W_{ij} \left(\chi d_{ik} + \alpha d_{km} + \delta d_{mj}\right) x_{ijkm}$$
subject to
$$\sum_{k,m\in\mathcal{N}} x_{ijkm} = 1 \qquad i, j \in \mathcal{N} \tag{18.8}$$
$$\sum_{m\in\mathcal{N}} x_{ijkm} + \sum_{m\in\mathcal{N}\setminus\{k\}} x_{ijmk} \le z_k \qquad i, j, k \in \mathcal{N} \tag{18.9}$$
$$x_{ijkm} \ge 0 \qquad i, j, k, m \in \mathcal{N} \tag{18.10}$$
$$z_k \in \{0, 1\} \qquad k \in \mathcal{N}. \tag{18.11}$$

Constraints (18.8) state that exactly one hub arc must be selected to route the flow from origin i to destination j. Constraints (18.9) ensure that O/D paths (i, k, m, j) use only open hubs. This formulation has $O(n^4)$ variables and $O(n^3)$ constraints and usually provides tight LP bounds. In addition, we note that this *arc-based formulation* is equivalent to a *path-based formulation* given that all O/D paths are completely characterized by the arc variables x_{ijkm}.

Given that O/D nodes can be connected to more than one hub facility, we can exploit some properties on the structure of O/D paths to do preprocessing in order to significantly reduce the number of required variables in the formulation. In particular, it is known that every flow uses at most one direction of a hub arc, the one with lower flow cost. We thus define an *undirected* flow cost F_{ije} for each $e = (k, m) \in \mathcal{E}$ and $i, j \in \mathcal{N}$ as $F_{ije} = \min\{F_{ijkm}, F_{ijmk}\}$. The number of variables can be further reduced by defining a set of candidate hub arcs E_{ij} for each O/D pair. This is done by using the property that no flow will be routed through a hub arc containing two hubs whenever it is cheaper to route it through only one of them.

The x_{ijkm} can be projected out from the *arc-based formulation* via Benders decomposition to obtain a valid formulation in the space of the binary variables z_i. The Benders reformulation of the arc-based formulation is:

$$\text{minimize} \quad \sum_{k \in \mathcal{N}} f_k z_k + \eta$$

$$\text{subject to} \quad \sum_{k \in \mathcal{N}} z_k \geq 1 \tag{18.12}$$

$$\eta \geq \sum_{i \in \mathcal{N}} a_i^r z_i \qquad r = 1, \ldots, |Q_D|, \tag{18.13}$$

$$z_k \in \{0, 1\} \qquad k \in \mathcal{N}, \tag{18.14}$$

where Q_D is the set of extreme points of the dual subproblem associated with constraints (18.8)–(18.9). Non-dominated Benders cuts (18.13) can be efficiently generated with ad hoc algorithms that resort on the solution of linear and network flow problems.

3.2 Single Assignments

Flow-based formulations can also be adapted to model HLPs with single assignments. Similarly to the case of multiple assignments, we use continuous variables to compute the amount of flow routed on a particular arc originated at a given node. However, in the case of single assignments, we only need to use the set of flow variables associated with the hub arcs (Y_{ikm}). For each pair $i, k \in \mathcal{N}$, we also define binary location/allocation variables z_{ik}, equal to one if and only if node i is assigned to hub k. When $i = k$, variable z_{kk} represents the establishment or not of a hub at node k. HLPs with single assignments can be formulated as follows:

$$\text{minimize} \quad \sum_{k \in \mathcal{N}} f_k z_{kk} + \sum_{i,k \in \mathcal{N}} (\chi O_i + \delta D_i)\, d_{ik} z_{ik} + \sum_{i,k,m \in \mathcal{N}} \alpha d_{km} Y_{ikm}$$

$$\text{subject to} \quad \sum_{m \in \mathcal{N}} Y_{imk} + O_i z_{ik} = \sum_{j \in \mathcal{N}} W_{ij} z_{jk} + \sum_{m \in \mathcal{N}} Y_{ikm} \qquad i, k \in \mathcal{N} \tag{18.15}$$

$$\sum_{k \in \mathcal{N}} z_{ik} = 1 \qquad i \in \mathcal{N} \tag{18.16}$$

$$z_{ik} \le z_{kk} \qquad i, k \in \mathcal{N} \tag{18.17}$$

$$z_{ik} \in \{0, 1\} \qquad i, k \in \mathcal{N} \tag{18.18}$$

$$Y_{ikm} \ge 0 \qquad i, k, m \in \mathcal{N}. \tag{18.19}$$

Constraints (18.15) state that the flow entering to hub k either directly from node i or via other hubs m has to be equal to the flow leaving to either other hubs m or to destination nodes j. Constraints (18.16) ensure that each O/D node is assigned to exactly one hub node. Finally, constraints (18.17) guarantee O/D nodes are assigned to open hubs. The above formulation contains $O(n^3)$ variables and $O(n^2)$ constraints.

HLPs with single assignments are closely related to classical discrete location problems. In fact, they can be modeled as facility location problems with additional quadratic costs associated with the interaction of O/D nodes. HLPs with single assignments can be stated as the following quadratic binary integer program:

$$\text{minimize} \quad \sum_{k \in \mathcal{N}} f_k z_{kk} + \sum_{i,k \in \mathcal{N}} (\chi O_i + \delta D_i)\, d_{ik} z_{ik} + \sum_{i,j,k,m \in \mathcal{N}} \alpha W_{ij} d_{km} z_{ik} z_{jm} \tag{18.20}$$

$$\text{subject to} \quad (18.16)\text{–}(18.18).$$

Note that constraints (18.16)–(18.18) define the set of feasible solutions to the so-called *uncapacitated facility location problem* (UFLP). In fact, when the quadratic term of the objective (18.20) is removed, the HLP with single assignments reduces to the UFLP. However, contrary to the UFLP, integrality conditions on the allocation variables z_{ik} need to be explicitly stated to have a valid formulation. This is mainly due to the fact that objective (18.20) is non-convex.

We now discuss different approaches that have been considered to handle the quadratic term of the objective (18.20). The first one is to use the *reformulation linearization technique* of Adams and Sherali (1990), to obtain the following linear MIP formulation:

$$\text{minimize} \quad \sum_{k \in \mathcal{N}} f_k z_{kk} + \sum_{i,k \in \mathcal{N}} (\chi O_i + \delta D_i)\, d_{ik} z_{ik} + \sum_{i,j,k,m \in \mathcal{N}} \alpha W_{ij} d_{km} x_{ijkm}$$

subject to (18.16)–(18.18)

$$\sum_{m \in \mathcal{N}} x_{ijkm} = z_{ik} \qquad i, j, k \in \mathcal{N} \tag{18.21}$$

$$\sum_{k \in \mathcal{N}} x_{ijkm} = z_{jm} \qquad i, j, m \in \mathcal{N} \tag{18.22}$$

$$x_{ijkm} \geq 0 \qquad i, j, k, m \in \mathcal{N}. \tag{18.23}$$

where x_{ijkm}, $i, j, k, m \in \mathcal{N}$, are variables equal to 1 if and only if the flow originated at i and destination j transits via hub arc (k, m). This formulation can be seen as an *arc-based formulation* in which constraints (18.21)–(18.22) are flow conservation equations for n^2 networks, each of which associated with an O/D pair (i, j). In addition, it contains $O(n^4)$ variables and $O(n^3)$ constraints and is known to provide tight LP bounds. Moreover, constraints (18.16) can be replaced by

$$\sum_{k,m \in \mathcal{N}} x_{ijkm} = 1 \qquad \forall i, j \in \mathcal{N}, \tag{18.24}$$

to obtain an alternative valid formulation. This highlights that, due to the particular structure of a fully interconnected hub-level network, this formulation can also be seen as a *path-based formulation* given that it uses path variables x_{ijkm} to characterize all O/D paths visiting either one or two hub nodes. In this case, constraints (18.24) correspond to the convexity constraints associated with O/D pairs. These arc/path-based formulations have been used in combination with decomposition methods to develop adhoc solution algorithms for efficiently solving various HLPs with single assignments (see, Sect. 5).

It is possible to use projection methods to eliminate the path variables x_{ijkm} of arc-based formulations to obtain MIP formulations with fewer variables. The first one is a direct method used in Mirchandani (2000) to project out flow variables for network loading problems. The second one is an indirect method used in Rardin and Wolsey (1993) for uncapacitated fixed charge network flow problems. Labbé and Yaman (2004) apply the direct projection method on an arc-based formulation and analyze the strength and dominance of these projection inequalities. The authors prove that a subset of these projection inequalities are facet-defining and that some others, are dominated by other families of facet-defining inequalities. Labbé et al. (2005) show that the projection inequalities defined by a subset of the extreme rays of the projection cone are sufficient to provide a valid formulation for HLPs with single assignments. In particular, HLPs with single assignments can be formulated as

$$\text{minimize} \sum_{k \in \mathcal{N}} f_k z_k + \sum_{i,k \in \mathcal{N}} (\chi O_i + \delta D_i) d_{ik} z_{ik} + \sum_{k,m \in \mathcal{N}} \alpha d_{km} y_{km}$$

subject to (18.16)–(18.18)

$$y_{km} \geq \sum_{(i,j)\in K} W_{ij}\left(z_{ik} + z_{jm} - 1\right) \qquad k, m \in \mathcal{N}, K \subseteq \mathcal{N} \times \mathcal{N} \quad (18.25)$$

$$y_{km} \geq 0 \qquad k, m \in \mathcal{N}, \quad (18.26)$$

where y_{km}, $k, m \in \mathcal{N}$ are an additional set of continuous variables equal to the amount of flow routed on hub arc (k, m). For each arc (k, m), constraints (18.25) and (18.26) imply

$$y_{km} = \max_{K\subseteq\mathcal{N}\times\mathcal{N}} \sum_{(i,j)\in K} W_{ij}\left(z_{ik} + z_{jm} - 1\right) = \sum_{(i,j)\in K_{km}} W_{ij}\left(z_{ik} + z_{jm} - 1\right),$$

where K_{km} is the set of all demands which are routed on hub arc (k, m). This formulation contains only $O(n^2)$ variables but an exponential number of constraints. Constraints (18.25) are a particular case of a more general class of facet defining inequalities which can be separated in polynomial time.

An alternative to project out the path variables x_{ijkm} is by using Benders decomposition (BD) to obtain a valid reformulation in the space of the original z_{ik} variables. In particular, the *Benders reformulation* of the arc-based formulation is:

$$\text{minimize} \quad \sum_{k\in\mathcal{N}} f_k z_{kk} + \sum_{i,k\in\mathcal{N}} (\chi O_i + \delta D_i) d_{ik} z_{ik} + \eta$$

$$\text{subject to} \quad (18.16)\text{–}(18.18)$$

$$\sum_{k\in\mathcal{N}} z_{kk} \geq 1 \quad (18.27)$$

$$\eta \geq \sum_{i,k\in\mathcal{N}} a^r_{ik} z_{ik} \qquad r = 1, \ldots, |P_D|, \quad (18.28)$$

where P_D is the set of extreme points of the dual subproblem associated with constraints (18.21)–(18.22). Even though there is an exponential number of constraints (18.28), non-dominated cuts can be efficiently separated with ad hoc algorithms that resort on the solution of linear and network flow problems.

3.3 r-Allocation

The r-allocation strategy provides flexibility in the design of hub networks without explicitly considering set-up costs on access arcs. It has as particular cases both single and multiple assignment strategies. Flow-based and arc-based formulations can also be adapted to model HLPs with r-allocation.

In the case of the flow-based formulation, we combine the location/allocation variables z_{ik} from the single assignments variant with the flow variables U_{ik}, Y_{ikm}, and X_{ijm} from the multiple assignments variant to model the collection, transfer, and distribution legs, respectively. Similarly to the multiple assignments strategy, we also need the U_{ik} variables for the collection leg, as it is no longer possible to model it using the allocation variables z_{ik}. Using these sets of variables, we obtain the following flow-based formulation:

$$\begin{aligned}
\text{minimize} \quad & \sum_{k \in \mathcal{N}} f_k z_{kk} + \sum_{i,k \in \mathcal{N}} \chi d_{ik} U_{ik} + \sum_{i,k,m \in \mathcal{N}} \alpha d_{km} Y_{ikm} + \sum_{i,j,m \in \mathcal{N}} \delta d_{mj} X_{ijm} \\
\text{subject to} \quad & (18.1)\text{–}(18.3),\ (18.6),\ (18.17)\text{–}(18.18) \\
& \sum_{k \in \mathcal{N}} z_{ik} \leq r \qquad i \in \mathcal{N} && (18.29) \\
& U_{ik} \leq O_i z_{ik} \qquad i,k \in \mathcal{N} && (18.30) \\
& X_{ijm} \leq W_{ij} z_{jm} \qquad i,j,m \in \mathcal{N}. && (18.31)
\end{aligned}$$

Constraints (18.29) ensure that each O/D node is allocated to at most r hub facilities, whereas constraints (18.30) and (18.31) state that flow can be routed on access arcs (i, k) and (j, m) only if they have been activated, respectively. These constraints are equivalent to constraints (18.4) and (18.5) from the multiple assignment variant but yield stronger bounds. Note that in order to model HLPs considering an r-allocation strategy, it is needed to combine not only the set of variables but also the set of constraints (18.1)–(18.3), (18.6) from the multiple assignments variant with constraints (18.17)–(18.18) from the single assignments variant.

In the case of the arc-based formulation, the location/allocation z_{ik} variables and the routing variables x_{ijkm} from the single assignments variant are enough to model the problem. The formulation is as follows:

$$\begin{aligned}
\text{minimize} \quad & \sum_{k \in \mathcal{N}} f_k z_{kk} + \sum_{i,j,k,m \in \mathcal{N}} W_{ij} \left(\chi d_{ik} + \alpha d_{km} + \delta d_{mj}\right) x_{ijkm} \\
\text{subject to} \quad & (18.17)\text{–}(18.18),\ (18.29) \\
& \sum_{k \in \mathcal{N}} \sum_{m \in \mathcal{N}} x_{ijkm} = 1 \qquad \forall\, i,j \in \mathcal{N} && (18.32) \\
& \sum_{m \in N} x_{ijkm} \leq z_{ik} \qquad i,j,k \in \mathcal{N} && (18.33) \\
& \sum_{k \in N} x_{ijkm} \leq z_{jm} \qquad i,j,m \in \mathcal{N} && (18.34) \\
& x_{ijkm} \geq 0 \qquad i,j,k,m \in \mathcal{N}. && (18.35)
\end{aligned}$$

Constraints (18.32) state that the flow associated with each node pair must be routed using one O/D path. Constraints (18.33) and (18.34) state that only O/D paths associated with active access arcs can be used to route commodities.

As expected, these formulations considering such flexible allocation strategies tend to be more difficult to solve as compared to the best formulations presented in Sects. 3.1 and 3.2 for specific multiple and single allocation variants.

4 Hub Network Design Problems

Full interconnection between hub nodes may be prohibitive in applications where there is a considerable set-up cost associated with the hub arcs. To overcome this drawback of standard HLPs, several problems considering incomplete hub networks have been studied. Formulating and solving more general HNDPs represent a bigger challenge as compared to standard HLPs. This is due to the fact that HNDPs involve additional design decisions such as link activation of hub, access and bridge arcs as well as non-trivial routing decisions. It is no longer possible to state HNDPs as quadratic extensions of facility location problems but rather as extensions of MNDPs in which node selection (i.e., location) decisions need to be taken into account. HNDPs are a class of network optimization problems known to be significantly more difficult to solve in practice as compared to facility location problems. One of the main reasons is that O/D paths may contain more than three arcs and visit more than two hubs. As a consequence, they cannot longer be mainly determined by the allocation decisions. Flow conservation constraints and additional design variables for arc selection decisions are now needed to explicitly model O/D paths in both flow and arc-based formulations. This has a negative impact in the quality of the LP bounds associated with these formulations when compared to the LP bounds obtained with standard HLPs.

In the first part of this section we concentrate on a particular class of HNLPs, referred to as *hub arc location problems* (HALPs), which have as key decisions the location of hub arcs. These problems retain some of the assumptions used in hub location models, specially the ones that relate to the cost structure and allocation patterns to simplify the design decisions at the access level network and to focus on the design decisions at the hub level network. In the second part we study HALPs that consider specific hub network topologies arising from various applications and highlight how these topologies impact the routing decisions.

4.1 Hub Arc Location Problems

A fundamental difference between HALPs and HLPs is that solution networks may not longer have a fully interconnected hub-level network. HALPs explicitly consider link activation decisions in hub and bridge arcs. Additional restrictions may be imposed on the topology of hub-level networks. However, an important simplifying assumption that is retained from HLPs, as compared with more general HNDPs, is that they do not involve non-trivial link activation decision on access arcs. That is, similar to HLPs, assignment patterns determine the design decisions in the access-level network. As a result, both single, multiple, and r-allocation HALPs variants can be considered.

In HALPs hubs are not necessarily fully interconnected due to the set up cost on the hub arcs or because additional conditions on the network topology are imposed. This causes O/D paths to become more involved, since they may use more than three arcs and visit more than two hubs. Similar to HLPs, because of Assumptions 2 and 4, each collection and distribution leg, if present, employs either one access arc or one bridge arc. However, the transfer leg can now use several bridge and hub arcs, depending on the particular assumptions considered on the structure of O/D paths.

To simplify the added complexity of the routing decisions in HALPs an additional assumption, referred to as the *one-hub-arc O/D path* assumption, can be considered. It states that O/D paths must contain at most one hub arc on the transfer leg. In turn, this limits paths to have at most three arcs, being the first and last ones either access or bridge arcs and the intermediate arc, if it exists, a hub arc. This assumption is used to duplicate the level of service obtained in HLPs and is also consistent with practice. In air transportation, for example, it ensures that a passenger will never have to change flights more than twice. In ground transportation, it is convenient to restrict the number of break-bulk terminals that each commodity has to pass through so as to reduce handling and congestion at terminals and to provide a form of performance guarantee. O/D paths are once more of the form (i, k, m, j), and we can thus define their associated flow costs as $W_{ij}\left(\chi d_{ik} + \alpha d_{km} + \delta d_{mj}\right)$.

In what follows we first describe some HALPs that consider the one-hub-arc assumption. We then discuss other more general HALPs that do not consider any assumption on the structure of O/D paths. In particular, we show how the routing the decisions become more involved given that it is needed to determine whether a discount is perceived between two hub nodes or not.

4.1.1 Models with One-Hub-Arc O/D Paths

These problems do not consider set-up costs on the activation of hub nodes and hub arcs. Instead, they considered a cardinality constraint on the number of hub arcs in the solution network. The selected hub arcs induce a set of hub nodes, but there is no limit on the number of activated hubs. These HALPs consider multiple assignments and the goal is to minimize the total flow cost.

Given that in this case bridge arcs can only exist in the collection or distribution legs, a flow-based formulation can be obtained by using the same set of flow variables U_{ik}, Y_{ikm}, and X_{ijm} used in HLPs to model flows passing on the collection, transfer, and distribution legs, respectively. In addition, for $(k, m) \in \mathscr{E}$, we define binary variables y_{km} equal to one if and only if hub arc (k, m) is selected. Using these sets of variables, we can formulate the problem as follows:

$$\text{minimize} \sum_{i,k \in \mathcal{N}} \chi d_{ik} U_{ik} + \sum_{i,k,m \in \mathcal{N}} \alpha d_{km} Y_{ikm} + \sum_{i,j,m \in \mathcal{N}} \delta d_{mj} X_{ijm}$$

subject to (18.1)–(18.2), (18.4)–(18.7)

$$\sum_{(k,m) \in \mathcal{E}} y_{km} = q \tag{18.36}$$

$$z_k \leq \sum_{(k,m) \in \mathcal{E}} y_{km} \qquad k \in \mathcal{N} \tag{18.37}$$

$$U_{ik} = \sum_{m \in \mathcal{N}} Y_{ikm} \qquad i,k \in \mathcal{N} \tag{18.38}$$

$$\sum_{m \in \mathcal{N}} Y_{imk} = \sum_{j \in \mathcal{N}} X_{ijk} \qquad i,k \in \mathcal{N} \tag{18.39}$$

$$Y_{ikk} \leq O_i z_k \qquad i,k \in \mathcal{N} \tag{18.40}$$

$$Y_{ikm} + Y_{imk} \leq M_{ikm} y_{km} \qquad i \in \mathcal{N}, (k,m) \in \mathcal{E} \tag{18.41}$$

$$y_{km} \in \{0,1\} \qquad (k,m) \in \mathcal{E}, \tag{18.42}$$

where $M_{ikm} \leq O_i$ is an upper bound on the amount of flow originated at i that can be routed via hub arc (k,m). Constraints (18.36) force the number of selected hub arcs to be equal to q, whereas constraints (18.37) ensure that a location variable z_k is activated only if there exist at least one hub arc incident to node k. These constraints, in combination with (18.4)–(18.7), and (18.40), ensure that flow variables U_{ik}, Y_{ikm}, and X_{ijm} are used only at open hubs. Constraints (18.41) guarantee that flow is routed via two different hub nodes with a discounted cost only if the associated hub arc is selected. Finally, constraints (18.38) and (18.39) are flow conservation constraints for each node i and each potential hub k. This formulation contains $O(n^3)$ variables and $O(n^3)$ constraints.

A more general class of HALPs with multiple assignments has also been studied. In particular, these problems contain both set-up costs and cardinality constraints on hub arcs and hub nodes. For each $e \in \mathcal{E}$, g_e denotes the set-up cost for selecting hub arc e. This class of HALPs consist of locating a set of at most q hub arcs ($q \geq 1$), that induce a set of at most p hub nodes ($p \geq 2$), and of determining the routing of commodities through the hub network, with the objective of minimizing the total set-up and flow cost.

Taking into account the one-hub-arc assumption, we define the cost for routing W_{ij} when using hub arc $e = (k,m)$ as, $F_{eij} = W_k \min\left\{F^1_{eij}, F^2_{eij}, F^3_{eij}, F^4_{eij}\right\}$, where

$$F^1_{eij} = \chi d_{ik} + \alpha d_{km} + \delta d_{mj}; \qquad F^2_{eij} = \chi d_{im} + \alpha d_{mk} + \delta d_{jk};$$

$$F^3_{eij} = \chi d_{ik} \qquad +\delta d_{kj}; \qquad F^4_{eij} = \chi d_{im} + \delta d_{mj}.$$

Note that the definition of F_{eij} uses some properties of the considered multiple assignments pattern and cost structure. First, every commodity uses at most one direction of a hub arc, the one with lower flow cost. It is thus possible to know a priori how the end hub nodes of a given hub arc e would be connected with origin i and destination j, in case such commodity is routed via hub arc e. Second, no commodity will be routed through a hub arc whenever it is cheaper to route it through only one of its hub nodes. Therefore, some O/D paths may not contain a transfer leg (i.e., a hub arc).

For each $i, j \in \mathscr{N}$ and $e \in \mathscr{E}$ we define (undirected) routing variables x_{eij} equal to 1 if and only if demand originated at i and destination j is routed via hub arc e. Using these variables, an arc-based formulation for this class of HALPs can be obtained as follows:

$$\text{minimize} \quad \sum_{i \in \mathscr{N}} f_i z_i + \sum_{e \in \mathscr{E}} g_e y_e + \sum_{i,j \in \mathscr{N}} \sum_{e \in \mathscr{E}} F_{eij} x_{ije}$$

$$\text{subject to} \quad \sum_{e \in \mathscr{E}} y_e \le p \tag{18.43}$$

$$\sum_{i \in \mathscr{N}} z_i \le q \tag{18.44}$$

$$\sum_{e \in \mathscr{E}} x_{ije} = 1 \qquad \forall i, j \in \mathscr{N} \tag{18.45}$$

$$x_{ije} \le y_e \qquad \forall e \in \mathscr{E}, i, j \in \mathscr{N} \tag{18.46}$$

$$y_e \le z_k \qquad \forall e = (k, m) \in \mathscr{E} \tag{18.47}$$

$$y_e \le z_m \qquad \forall e = (k, m) \in \mathscr{E} \tag{18.48}$$

$$y_e, z_i \in \{0, 1\} \qquad \forall e \in \mathscr{E}, i \in \mathscr{N} \tag{18.49}$$

$$x_{ije} \ge \qquad \forall e \in \mathscr{E}, i, j \in \mathscr{N}. \tag{18.50}$$

Constraints (18.43) and (18.44) state the maximum cardinality constraint on the hub arcs and hub nodes, respectively. Constraints (18.45) guarantee that every commodity is assigned to exactly one hub arc, whereas (18.46) allow commodities to be routed only via selected hub arcs. Constraints (18.47) and (18.48) ensure that the end nodes of hub arcs are open hub nodes. This formulation has $O(n^4)$ variables and $O(n^4)$ constraints.

This general class of HALPs can be stated as the minimization of a real-valued supermodular set function. This fundamental property, which is also known for other types of facility location problems (Wolsey 1983), can be exploited to develop formulations. In particular, using supermodular properties, it is possible to completely eliminate the routing variables x_{ijkm} from the above formulation. For each $i, j \in \mathcal{N}$, we order the elements of $\mathcal{E}$ by non-decreasing values of their coefficients F_{ije}, and we denote e_{rk} to the r-th element according to that ordering. That is, $F_{ije_1} \leq F_{ije_2} \leq \cdots \leq F_{ije_{|\mathcal{E}|}} \leq F_{ije_{|\mathcal{E}|+1}}$, where $F_{ije_{|\mathcal{E}|+1}} = F_{ije^*}$ is the cost for the fictitious edge e^* such that (1) $F_{ije^*} > \max_{e\in\mathcal{E}} F_{ije}$, for all $i, j \in \mathcal{N}$; and (2) $\sum_{i,j\in\mathcal{N}} F_{ije*} > \max_{e\in\mathcal{E}}(f_e + \sum_{i,j\in\mathcal{N}} F_{ije})$. This assumption guarantees that at least one hub variable y_e is at value one in any optimal solution. A formulation for this class of HALPs is as follows:

$$\begin{aligned}
\text{minimize} \quad & \sum_{i\in\mathcal{N}} f_i z_i + \sum_{e\in\mathcal{E}} g_e y_e + \sum_{i,j\in\mathcal{N}} \eta_{ij} \\
\text{subject to} \quad & (18.43)\text{–}(18.44),\ (18.47)\text{–}(18.49) \\
& \eta_{ij} \geq F_{ije_r} + \sum_{e\in\mathcal{E}} (F_{ije} - F_{ije_r})^{-} y_e \quad r = 1, \ldots, |\mathcal{E}|+1,\ i, j \in \mathcal{N},
\end{aligned}$$

$$(18.51)$$

where η_{ij} are continuous decision variables used to evaluate the flow cost of O/D pair (i, j) and $(x)^{-} = \min\{0, x\}$. Constraints (18.51) are the so-called supermodular constraints computing the flow cost for each O/D pair by only taking into account the set of open hub arcs. This formulation has only $O(n^2)$ variables and $O(n^4)$ constraints.

4.1.2 Models with Arbitrary O/D Paths

We now focus on a general class of HALPs that relax the one-hub-arc O/D path assumption and allow paths to contain more than one hub/bridge arc on the transfer leg. A flow-based formulation can be obtained by using the same set of flow variables U_{ik}, Y_{ikm}, and X_{ijm} as before plus an additional set of flow variables B_{ikm}, $i, j, k \in \mathcal{N}$, equal to the amount of flow originated at node i and passing through bridge arc (k, m). Let β denote the unit flow cost of bridge arcs, where $\beta > \alpha$. A flow-based formulation can be stated as follows:

$$\begin{aligned}
\text{minimize} \quad & \sum_{i,k\in\mathcal{N}} \chi d_{ik} U_{ik} + \sum_{i,k,m\in\mathcal{N}} d_{km} (\alpha Y_{ikm} + \beta B_{ikm}) + \sum_{i,j,m\in\mathcal{N}} \delta d_{mj} X_{ijm} \\
\text{subject to} \quad & (18.1)\text{–}(18.2),\ (18.4)\text{–}(18.7),\ (18.36),\ (18.37),\ (18.40),\ (18.41) \\
& U_{ik} + \sum_{m\in\mathcal{N}} (Y_{imk} + B_{imk}) \\
& = \sum_{m\in\mathcal{N}} (Y_{ikm} + B_{ikm}) + \sum_{j\in\mathcal{N}} X_{ijk} \quad i, k \in \mathcal{N} && (18.52) \\
& B_{ikm} \leq M_{ikm} z_k \qquad i, k, m \in \mathcal{N} && (18.53) \\
& B_{ikm} \leq M_{ikm} z_m \qquad i, k, m \in \mathcal{N} && (18.54)
\end{aligned}$$

$$B_{ikm} \geq 0 \qquad i, k, m \in \mathcal{N}.$$

Constraints (18.52) correspond to the flow conservation equations for each origin i and potential hub node k. Note that the flow entering into a hub can come either directly from the origin i or from other hub nodes via hub arcs or bridge arcs. Similarly, the flow leaving the node can go either directly to destination nodes j or to other hub nodes via hub arcs and bridge arcs. Constraints (18.53) and (18.54) ensure that bridge arcs are used only between open hub nodes. This formulation contains $O(n^3)$ variables and $O(n^3)$ constraints.

Arc-based formulations can also be adapted for this class of HALPs. Given that O/D paths cannot longer be characterized by using only the routing variables x_{ijkm}, as it is the case in HLPs with multiple assignments and HALPs with one-hub-arc O/D paths, we need to combine them with other variables to properly model O/D paths. In particular, we use the U_{ik} and X_{ijm} variables used in previous flow-based formulations to model the collection and distribution legs, respectively, together with the routing variables x_{ijkm} that state whether the hub arc (k, m), and its associated discounted cost, is used to route the demand associated with node pair i, j. In addition, we define the (non discounted) routing variables b_{ijkm} equal to one if and only if the flow originated at i and destination j uses bridge arc (k, m). Note that both x_{ijkm} and b_{ijkm} are required to properly model the transfer leg. An arc-based formulation can be stated as follows:

$$\text{minimize} \quad \sum_{i,j \in \mathcal{N}} W_{ij} \left(\sum_{k \in \mathcal{N}} \chi d_{ik} U_{ijk} + \sum_{k,m \in \mathcal{N}} d_{km} \left(\alpha x_{ijkm} + \beta b_{ijkm} \right) + \sum_{m \in \mathcal{N}} \delta d_{mj} X_{ijm} \right)$$

subject to (18.7), (18.36) and (18.37)

$$\sum_{k \in N} U_{ijk} = 1 \qquad i, j \in \mathcal{N} \tag{18.55}$$

$$\sum_{m} X_{ijm} = 1 \qquad i, j \in \mathcal{N} \tag{18.56}$$

$$U_{ijk} + \sum_{m \in \mathcal{N}} \left(x_{ijmk} + b_{ijmk} \right)$$

$$= \sum_{m \in \mathcal{N}} \left(x_{ijkm} + b_{ijkm} \right) + \sum_{j \in \mathcal{N}} X_{ijk} \quad i, j, k \in \mathcal{N} \tag{18.57}$$

$$U_{ijk} \leq z_k \qquad i, j, k \in \mathcal{N} \tag{18.58}$$

$$X_{ijm} \leq z_m \qquad i, j, m \in \mathcal{N} \tag{18.59}$$

$$x_{ijkm} + x_{ijmk} \leq y_{km} \qquad i, j \in \mathcal{N}, (k, m) \in \mathcal{E} \tag{18.60}$$

$$\sum_{m \in \mathcal{N}} b_{ijkm} \leq z_k \qquad i, j, k \in \mathcal{N} \tag{18.61}$$

$$\sum_{k \in \mathcal{N}} b_{ijkm} \leq z_m \qquad i, j, m \in \mathcal{N} \tag{18.62}$$

$$U_{ijk}, Y_{ijk} \geq 0 \qquad i, j, k \in \mathcal{N}.$$

$$x_{ijkm}, b_{ijkm} \geq 0 \qquad i, j, k, m \in \mathscr{N}.$$

Constraints (18.55)–(18.57) correspond to the flow conservation equations for each node pair and potential hub. Constraints (18.58)–(18.59) forces the flow on the access arcs to be routed via open hubs. Constraints (18.60) ensure that discounted costs are perceived on the transfer leg only for the selected hub arcs, whereas (18.61) and (18.62) allow bridge arcs to be used only between open hub nodes. This formulation has $O(n^4)$ variables and $O(n^4)$ constraints.

Note that in none of the HLPs and HALPs discussed until now, there has been a need to add flow conservation constraints for the case of arc-based formulations. All previous formulations exploited in one way or the other the property (or assumption) that O/D paths can be characterized by the hubs to which origins and destinations are assigned to. Note that not only the required number variables has doubled, but also several additional constraints are need to model feasible O/D paths.

4.2 *Specific Hub Network Topologies*

We now focus on HNDPs that consider specific hub network topologies emerging from various applications in transportation and telecommunications. In particular, we study four topologies: star-start hub networks, tree-start hub networks, cycle-star hub networks, and hub line networks. We describe the main applications associated with these topologies and provide formulations that exploit their structure.

4.2.1 Star-Star Hub Networks

A start-start hub network consists of a set of hub nodes directly connected to a central hub node (i.e., a hub-level network is a star). Each O/D node is connected to a hub node, creating a set of stars at the access-level network (see Fig. 18.2a). Applications of such networks arise in the design of satellite communication networks (Helme and Magnanti 1989), where homing stations (hub facilities) containing an earth station and a local switch are used in combination with terrestrial and satellite links to connect node pairs. Nodes connected to the same homing station communicate through the local switch, whereas nodes connected to different homing stations use their assigned earth stations and the satellite. Other applications of start-start hub networks arise in the area of cargo delivery. Yaman (2008) provides a concrete application associated with one of the largest cargo delivery companies in Turkey, in which a star-star hub network with central hub located in Ankara is used. Commodities originated at a city are sent to a single hub. At the hub, cargo arriving from different cities are collected and sorted. If the destination is served by the same hub, the cargo is routed directly to its destination. Otherwise, the cargo is sent to a central hub facility where it is further routed to the hub of the destination

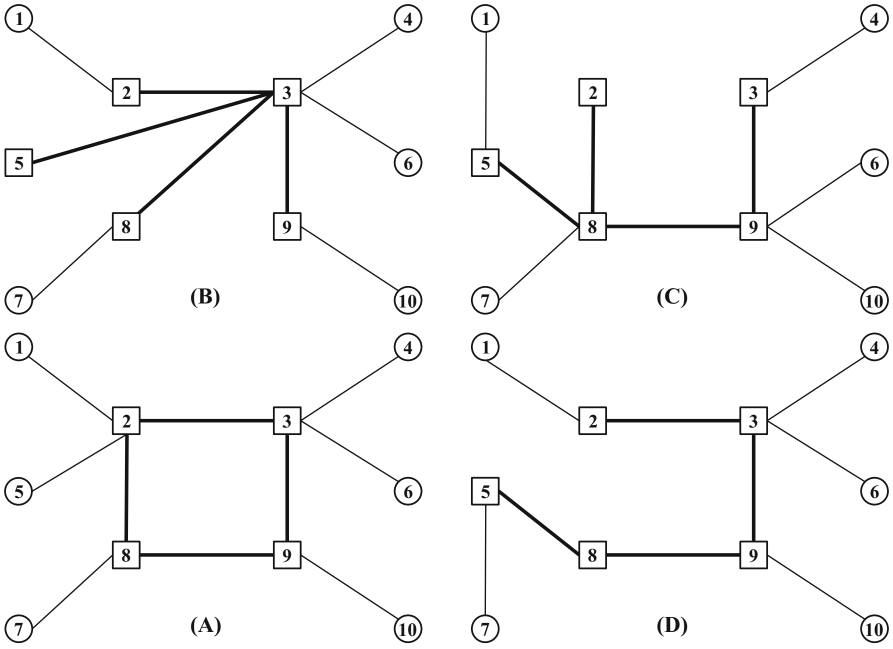


Fig. 18.2 Structure of (**a**) cycle-star, (**b**) star-star, (**c**) tree-star, and (**d**) line hub network

and eventually to its destination. Hub arcs are served with higher capacity trucks or cargo airplanes.

Let 0 denote the central hub which has already been located and let d_{k0} denote the unit flow cost between hub k and node 0. Assuming a star structure at the hub-level network simplifies, to some extent, the hub arc selection and routing decisions. For instance, arc selection decisions are determined by the location decisions—if a hub is located at node k, the hub arc $(k, 0)$ will be activated. Moreover, exactly two possible paths exist to connect node pairs. On the one side, if two nodes i and j are assigned to the same hub k, then the flow from node i to node j will follow the path (i, k, j), containing only two access arcs and no hub arcs. On the other side, if node i is assigned to hub k and node j is assigned to hub $m \neq k$, then the flow from i to j will follow the path $(i, k, 0, m, j)$. That is, it will contain two access arcs and two hub arcs. This means that in order to compute the flow cost for each O/D pair we only need to know which type of path will be used. It is possible to exploit this feature to model star-star hub networks as follows. For each $k \in \mathcal{N}$ and $i, j \in \mathcal{N}$, $i \neq j$, we define the variable u_{ijk} equal to one if and only if one of the nodes i and j is assigned to hub k. That is, when u_{ijk} is equal to one that means that flows between nodes i and j (in both directions) are routed on the hub arc $(k, 0)$. Using these variables in combination with the location/allocation variables z_{ik} we obtain the following formulation:

$$\begin{aligned}
\text{minimize} \quad & \sum_{k\in\mathcal{N}} f_k z_{kk} + \sum_{i,k\in\mathcal{N}} (\chi O_i + \delta D_i) d_{ik} z_{ik} + \sum_{k\in\mathcal{N}} \sum_{i,j\in\mathcal{N}: i\neq j} \left(W_{ij}+W_{ji}\right) \alpha d_{k0} u_{ijk} \\
\text{subject to} \quad & (18.16)\text{–}(18.18) \\
& u_{ijk} \geq z_{ik} - z_{jk} \quad k \in \mathcal{N}, i, j\mathcal{N}, i \neq j \qquad (18.63) \\
& u_{ijk} \geq z_{jk} - z_{ik} \quad k \in \mathcal{N}, i, j\mathcal{N}, i \neq j. \qquad (18.64)
\end{aligned}$$

Note that constraints (18.63)–(18.64) are used to model the nonlinear term $u_{ijk} = |z_{ik} - z_{jk}|$ that makes u_{ijk} variables equal to one whenever two O/D nodes are assigned to different hubs. This formulation contains $O(n^3)$ variables and $O(n^3)$ constraints.

An alternative formulation can be obtained by using projection inequalities similar to the ones introduced in Sect. 3.2 for HLPs with single assignments. In particular, for each $j \in \mathcal{N}$ we define continuous variables y_k equal to the amount of flow routed on hub arc $(k, 0)$. We can then formulate this problem as:

$$\begin{aligned}
\text{minimize} \quad & \sum_{k\in\mathcal{N}} f_k z_{kk} + \sum_{i,k\in\mathcal{N}} (\chi O_i + \delta D_i) d_{ik} z_{ik} + \sum_{k\in\mathcal{N}} \alpha d_{k0} y_k \\
\text{subject to} \quad & (18.16)\text{–}(18.18) \\
& y_k \geq \sum_{(i,j)\in K} \left(W_{ij} + W_{ji}\right)\left(z_{ik} - z_{jk}\right) \qquad k \in \mathcal{N}, K \subseteq \mathcal{N} \times \mathcal{N}, \qquad (18.65)
\end{aligned}$$

were constraints (18.65) have the same interpretation as (18.25), i.e., to compute the amount of flow routed on hub arc $(k, 0)$. This formulation has only $O(n^2)$ variables but an exponential number of constraints.

4.2.2 Tree-Star Hub Networks

A tree-star hub network consists of a set of hub nodes connected via a spanning tree (i.e., hub-level network is a tree). Each O/D node is assigned to exactly one hub, creating a set of stars at the access-level network (see Fig. 18.2b). Potential applications of such networks arise in the design of digital data service networks (Lee et al. 1996), where private service networks are constructed for individual organizations by connecting customer sites to digital switching offices (hub facilities) with bridging capabilities. These hubs are connected with fiber optic links and given that there is a very high set-up cost associated with these links, service providers usually consider tree topologies to minimize the number of required links to provide connection services between customer sites. Other applications of tree-star hub networks arise in the design of rapid transit systems. Contreras et al. (2010) give a concrete example in the design of the high-speed train network in Spain, which has been designed with a tree structure and it is intended that, when finished, each city (O/D node) with more than 10,000 inhabitants will be within 50 km of some high-speed train station (hub facilities). Kim and Tcha (1992) provides additional applications of tree-star networks in the design of community access television systems (CATV).

Contrary to star-star topologies, arc selection and routing decisions become more involved in the case of tree-star topologies as these cannot be determined by the location/allocation decisions. In fact, even if the location of hubs and allocation of O/D nodes to hubs is given, the problem is still NP-hard as it reduces to the well-known *optimum communication spanning tree problem* (Hu 1974; Zetina et al. 2019). Before discussing formulations, we define the graph of flows $\mathscr{G}_F = (\mathscr{N}, \mathscr{E}_F)$, as the undirected graph with node set $\mathscr{N}$ and an edge associated with each pair $(i, j) \in \mathscr{N} \times \mathscr{N}$ such that $W_{ij} + W_{ji} > 0$. We assume that $\mathscr{G}_F$ is made up of a single connected component since otherwise the problem can be decomposed into several independent ones, one for each connected component in $\mathscr{G}_F$. Whenever a particular application requires a single tree and the graph of flows contains more than one connected component, we can replace the flows of value zero with $W_{ij} = \epsilon > 0$ sufficiently small.

Hub arc variables y_{km} used in HALPs can also be employed to construct the tree of hubs. Moreover, flow conservation constraints are explicitly included in formulations to model O/D paths. A flow-based formulation for the design of tree-star hub networks can be stated as follows:

$$\text{minimize} \quad \sum_{i,k \in \mathscr{N}} (\chi O_i + \delta D_i)\, d_{ik} z_{ik} + \sum_{i,k,m \in \mathscr{N}} \alpha d_{km} Y_{ikm}$$

$$\text{subject to} \quad (18.15)\text{–}(18.19),\ (18.41)\text{–}(18.42)$$

$$\sum_{k \in \mathscr{N}} z_{kk} = p \tag{18.66}$$

$$\sum_{(k,m) \in \mathscr{E}} y_{km} = p - 1 \tag{18.67}$$

$$z_{km} + y_{km} \le z_{mm} \qquad (k, m) \in \mathscr{E} \tag{18.68}$$

$$z_{mk} + y_{km} \le z_{kk} \qquad (k, m) \in \mathscr{E}. \tag{18.69}$$

Constraints (18.66) and (18.67) ensure that exactly p hub nodes and $p - 1$ hub arcs are selected, respectively, which in combination with flow conservation constraints (18.15) guarantee that the selected $p - 1$ hub arcs define a single connected component associated with the p selected hubs, i.e., a tree spanning all hubs. Constraints (18.68) and (18.69) are a stronger version of the standard linking constraints (18.47) and (18.48). Finally, note that the assumption that the graph of flows $\mathscr{G}_F$ contains a single connected component, together with (18.66), (18.67) and (18.15), eliminates the need for subtour elimination constraints. However, when direct connections are allowed between non-hub nodes the following set of constraints need to be included to obtain a valid formulation.

$$\sum_{(k,m)\in S\times S} y_{km} \leq \sum_{k\in S\setminus\{s\}} z_k \qquad \forall S \subseteq \mathcal{N}, s \in S. \tag{18.70}$$

An arc-based formulation can also be used to design tree-star hub networks:

$$\begin{aligned} \text{minimize} \quad & \sum_{i,k\in\mathcal{N}} (\chi O_i + \delta D_i) d_{ik} z_{ik} + \sum_{i,j,k,m\in\mathcal{N}} \alpha W_{ij} d_{km} x_{ijkm} \\ \text{subject to} \quad & (18.16)\text{–}(18.18),\ (18.66)\text{–}(18.69) \\ & z_{ik} + \sum_{m\in\mathcal{N}} x_{ijmk} = \sum_{m\in\mathcal{N}} x_{ijkm} + z_{jk} \quad i,j,k \in \mathcal{N}. \end{aligned} \tag{18.71}$$

Constraints (18.71) are the flow conservation equations stating that for each node pair (i, j) and potential hub k, the flow from i to j may enter node k either directly from its origin via access arc (i, k) or through another hub via a hub arc (m, k). Similarly, the flow may exit node k either directly to its destination via access arc (m, j) or through another hub via a hub arc (k, m).

4.2.3 Cycle-Star Hub Networks

A cycle-star hub network consists of a set of hub nodes connected with a set of hub arcs by means of a cycle. Each O/D node must be connected to exactly one hub node, creating a set of stars at the access-level network (see Fig. 18.2c). Potential applications of cycle-star hub networks arise in the design of telecommunication networks (Lee et al. 1993; Xu et al. 1999) where a number of tributary networks are connected to a backbone network via a set of hubs. Given the large set-up costs associated with the installation of a set of links, network planners usually consider the design of a network containing the minimum number of links. Although tree-star and line-star topologies are attractive network topologies for this goal, these may not be appropriate for telecommunications networks where there are requirements for the backbone network to guarantee the existence of at least one path between O/D nodes in case a backbone link fails. A cycle-star hub network ensures connectivity of the network in such disruptive scenario while minimizing the set-up cost. Additional applications arise in the design of rapid transit systems. Network planners may be interested in the extension of public transportation networks in a metropolitan areas by installing a circular rapid transit line, such as a subway, a tram or an express lane. Examples of circular lines are the Moscow Underground, the Melbourne Circular Tram Line, and some of the Montreal bus lines (e.g., 33, 55, and 470). In some situations, a cycle is desirable not only due to reliability requirements but also because it offers an alternative path which can considerably reduce the travel time between node pairs.

Similar to tree-star topologies, arc selection and routing decisions are more involved as these cannot be determined by location/allocation decisions. In fact, even if the location of hubs and allocation of O/D nodes is given, the problem is still NP-hard as it reduces to the so-called *minimum flow cost Hamiltonian cycle problem* (Ortiz-Astorquiza et al. 2015). Given that hub arcs are undirected and uncapacitated, for each pair of hub nodes there exist exactly two possible paths on the cycle and the flows associated with the O/D nodes allocated to such hubs will be routed through the least cost path containing an undetermined number of hub arcs. A flow-based formulation for the design of cycle-star hub networks can be stated as follows:

$$\text{minimize} \quad \sum_{i,k \in \mathscr{N}} (\chi O_i + \delta D_i)\, d_{ik} z_{ik} + \sum_{i,k,m \in \mathscr{N}} \alpha d_{km} Y_{ikm}$$

$$\text{subject to} \quad (18.15)\text{–}(18.19),\ (18.41)\text{–}(18.42),\ (18.68)\text{–}(18.69)$$

$$\sum_{k \in \mathscr{N}} z_{kk} = p \tag{18.72}$$

$$\sum_{(k,m) \in \mathscr{E}} y_{km} = p \tag{18.73}$$

$$\sum_{(k,m) \in \mathscr{E}} y_{km} = 2 z_k \qquad k \in \mathscr{N}. \tag{18.74}$$

Constraints (18.72)–(18.74) together with flow conservation equations (18.15) guarantee that the set of selected hub arcs form a cycle of hubs. Similar to tree-star hub networks, subtour elimination constraints (18.70) need to be incorporated when considering direct connections between non-hub nodes to obtain a valid formulation.

An arc-based formulation can be stated as follows:

$$\text{minimize} \quad \sum_{i,k \in \mathscr{N}} (\chi O_i + \delta D_i) d_{ik} z_{ik} + \sum_{i,j,k,m \in \mathscr{N}} \alpha W_{ij} d_{km} x_{ijkm}$$

$$\text{subject to} \quad (18.16)\text{–}(18.18),\ (18.71),\ (18.68)\text{–}(18.69),\ (18.72)\text{–}(18.74).$$

4.2.4 Hub Line Networks

A hub line network consists of a set of hub nodes connected by means of a path (or line). In this case, each O/D node can be assigned to more than one hub node, i.e., a multiple allocation pattern (see Fig. 18.2d). Potential applications for hub lines arise in public transportation planning, in particular in the design of rapid transit systems and highway networks (Martins de Sá et al. 2015). Network planners may consider the expansion of an existing network in a metropolitan region to improve users' travel time by installing a rapid transit line, such as a subway, tram, or light rail line or an express bus lane. Hubs correspond to central stations such as subway, tram, bus or train stations. The aim is to minimize the total travel time between node pairs. Additional applications of hub line topologies appear in the design of road networks, where network planners may be interested in extending current road network in urban, suburban, or rural regions when constructing a new path-shaped highway or express lane. Hub nodes can be seen as a set of interchanges between highways and other existing roads (Lari et al. 2008).

Similar to tree-star and cycle-star hub networks, arc selection and routing decisions are not determined by the location decisions. Hub arc variables y_{km} are needed to construct the (undirected) line of hubs. Also, flow variables U_{ijk} and X_{ijk} used in HALPs are required to model the routing at the access-level network. An arc-based formulation for the design of hub line networks is as follows:

$$\text{minimize} \sum_{i,j} W_{ij} \left(\sum_{k \in \mathscr{N}} \chi d_{ik} U_{ijk} + \sum_{k,m \in \mathscr{N}} d_{km} \alpha x_{ijkm} + \sum_{m \in \mathscr{N}} \delta d_{mj} X_{ijm} \right)$$

subject to (18.7), (18.55)–(18.56), (18.58)–(18.60)

$$\sum_{k \in \mathscr{N}} z_k = p \tag{18.75}$$

$$\sum_{(k,m) \in \mathscr{E}} y_{km} = p - 1 \tag{18.76}$$

$$\sum_{(k,m) \in \mathscr{E}} y_{km} \le 2 z_k \qquad k \in \mathscr{N} \tag{18.77}$$

$$U_{ijk} + \sum_{m \in \mathscr{N}} x_{ijmk} = \sum_{m \in \mathscr{N}} x_{ijkm} + X_{ijk} \quad i, j, k \in \mathscr{N}. \tag{18.78}$$

Constraints (18.75)–(18.77) limit the number of hub nodes, hub arcs and degree of each hub node to at most two, respectively. Constraints (18.78) are the flow conservation equation which properly account for flows whenever a hub k is used. All these constraints together ensure that the hub-level is connected, forming a hub line. Similar to tree-star and cycle-star hub networks subtour elimination constraints (18.70) need to be added when considering direct connections between non-hub nodes to obtain a valid formulation.

5 Bibliographical Notes

The study of HLPs began with the pioneering work of O'Kelly (1986a), for continuous models, and O'Kelly (1986b), for discrete models. Campbell (1994a), Klincewicz (1998), and Bryan and O'Kelly (1999) provide early reviews and focus on classification schemes, fundamentals, and models with applications in the areas of telecommunications and air transportation. Campbell et al. (2001) wrote a comprehensive survey in which the location of hubs is the key decision. Alumur and Kara (2008) provided a classification scheme and review of the growing literature on network hub location models before 2008. Campbell and O'Kelly (2012) provided an insight into early motivations for analyzing hub location models and highlighted recent research directions. Contreras and O'Kelly (2019) wrote a concise overview of the main developments and most recent trends in hub location such as flow dependent discounted costs, capacitated models, uncertainty, dynamic and multi-modal models, and competition and collaboration.

In what follows, we highlight some of the most relevant references with respect to the development of mathematical models and solution algorithms for solving hub location and hub network design problems.

5.1 Hub Location Problems

O'Kelly (1987) provided the first formulation for a hub location problem with single assignments. O'Kelly formulated this problem as a discrete location problem with additional quadratic costs associated with the interaction of O/D nodes. The study of hub location models with multiple assignments originated in Campbell (1992), in which various formulations for this class of problems were presented.

The work of Skorin-Kapov et al. (1997) and Ernst and Krishnamoorthy (1996, 1998b) presented the first generation of tight arc-based formulations and useful flow-based formulations for single and multiple allocation variants of HLPs. However, the tightest arc-based formulation known so far for multiple assignment variants is the one independently introduced by Hamacher et al. (2004) and Marín (2005a). The former obtains this formulation by lifting facet-defining inequalities of the well-known uncapacitated facility location problem whereas the latter obtains the same set of facet-defining constraints as well as other facets by reformulating the problem as a set packing problem and identifying maximum cliques in an auxiliary graph. Boland et al. (2004) presented preprocessing procedures to reduce the number of variables and constraints for flow-based formulations, as well as some valid inequalities that improve LP relaxation bounds of capacitated variants.

Contreras et al. (2009b) used the Benders reformulation of Sect. 3.1 to develop and exact algorithm for uncapacitated HLPs with multiple assignments that, in combination with other algorithmic features such as preprocessing, a heuristic, and elimination tests, provided solutions for large-scale instances with up to 500 nodes. This Benders reformulation has also been extended to solve multi-level capacitated instances with up to 300 nodes (Contreras et al. 2012), and stochastic problems dealing with uncertainties in both demand flows and transportation costs (Contreras et al. 2011a).

Arc-based formulations have also been used to develop ad hoc solution algorithms for various HLPs with single assignments. Pirkul and Schilling (1998) use a Lagrangean relaxation (LR) in which constraints (18.16), (18.21)–(18.22) are relaxed to approximately solve p-hub median problems with single assignments. Contreras et al. (2009a) and Elhedhli and Wu (2010) use LRs in which constraints (18.21)–(18.22) are relaxed to solve capacitated HLP variants. Contreras et al. (2011b) use the LR of Contreras et al. (2009a) to solve integer restricted master problems containing a small subset of the x_{ijkm} variables and generating more if needed with a column generation procedure. This lower bounding procedure is embedded within a branch-and-price algorithm to solve capacitated instances with up to 200 nodes. Contreras et al. (2010, 2017) presented some families of extended cut-set inequalities that can help improve the LP bounds associated with flow-based formulations. Labbé et al. (2005) developed a branch-and-cut algorithm that uses several families of projection inequalities to solve quadratic capacitated variants with up to 50 nodes.

Camargo and Miranda (2012) and Camargo et al. (2011) were the first works to introduce Benders reformulations for solving HLPs with single assignments. The authors use a hybrid outer-approximation / Benders decomposition algorithm for dealing with the nonlinearity caused by functions used to represent congestion at hubs. Contreras et al. (2021) recently used a Benders reformulation within a branch-and-cut framework to optimally solve uncapacitated and capacitated instances with up to 900 nodes.

5.2 Hub Network Design Problems

O'Kelly and Miller (1994) is the first work discussing the need of including additional design decisions in hub location models. The authors provide a classification of hub network topologies based on protocols that consider the allocation pattern of O/D nodes, the interconnection between hub nodes, and the possibility of allowing direct connections between O/D nodes.

Campbell et al. (2005a) introduced HALPs and provided a classification scheme for them that accounts for assumptions on hub-level network decisions, access-level network decisions, and O/D path decisions. In a follow-up paper, Campbell et al. (2005b) presented an exact enumeration-based algorithm to solve instances with up to 25 nodes and $q = 6$ for several classes of HALPs. Contreras and Fernández (2014) showed how a general class of HALPs can be stated as the minimization of a real-valued supermodular set function and developed a branch-and-cut algorithm using supermodular cuts to solve various particular cases of HALPs for instances with up to 125 nodes.

More complex HALPs have been studied where additional features need to be taken into account. In the context of air passenger transportation, Sasaki et al. (2014) study competitive HALPs with multiple assignments in a Stackelberg framework. Gelareh et al. (2010) deals with another competitive HALP with multiple assignments arising in liner shipping. The authors extend path-based formulations for this variant and present a LR algorithm to obtain bounds for instances with up to 20 nodes. Tanash et al. (2017) focus on HALPs with single assignments in which flow dependent costs are considered. The authors propose a branch-and-bound algorithm that uses an arc-based formulation to solve instances with up to 50 nodes. Gelareh and Nickel (2011) study HALPs with multiple assignments arising in urban transport and liner shipping. A Benders decomposition algorithm is proposed to solve the considered problem. A multi-period extension of this problem is presented in Gelareh et al. (2015), where a Benders decomposition is also used to solve it. Rothenbcher et al. (2016) deals with HALPs with multiple assignments in which there exist capacities on hub arcs. A branch-and-price algorithm that uses a path-based formulation is developed to solve the problem. The pricing of the path variables is NP-hard as it corresponds to solving a *shortest path problem with resource constraints*. Camargo et al. (2017) focus on HALPs with multiple assignments in which flow-dependent discounted flow costs and hop-constraints are considered. The authors present a Benders reformulation and a branch-and-cut algorithm to solve it. After a number of algorithmic features are employed to generate non-dominated cuts, instances with up to 80 nodes can be optimally solved.

In the case of star-star hub network topologies, Labbé and Yaman (2008) performed a polyhedral analysis and show that inequalities (18.63)–(18.65) are facet-defining. Using a LR algorithm based on the above formulation, the authors solved instances with up to 150 nodes. Tree-start hub networks seem to be more challenging to solve. Martins de Sá et al. (2013) presented a Benders decomposition algorithm that uses the arc-based formulation to optimally solve instances with up to 100 nodes. Contreras et al. (2017) developed a branch-and-cut algorithm for HNDPs with a cycle-star topology using a flow-based formulation in combination with a general class of mixed-dicut inequalities to solve instances with up to 100 nodes. In the case of hub lines, Martins de Sá et al. (2015) introduced a Benders reformulation based on an arc-based formulation and developed a branch-and-cut algorithm to solve instances with up to 100 nodes.

6 Conclusions and Perspectives

We have provided an overview of hub network design problems in which both location and arc selection are key decisions. We focused on the role network design and routing decisions play in the formulation and solution of various classes of hub network design problems of increasing complexity. We pointed out how the assumptions and properties presented in Sect. 2 simplify the network design decisions, giving rise to a first generation of hub location models dealing mostly with the location of hubs and the assignment of O/D nodes to open hubs. We have also highlighted how network design decisions become more involved when removing some of these assumptions, leading to the study of a second generation of models sharing more features to the more complex multi-commodity network design problems than to discrete facility location problems.

Although substantial progress has been made by researchers and practitioners in the area of hub network design, there is still significant work ahead. In many practical applications additional design and tactical decisions need to be taken into account to accurately model the associated systems. For instance, some applications require the design of more complex access networks that are no longer determined by an assignment pattern of O/D nodes to hubs. Klincewicz (1998) reviews various models arising in the design of telecommunications networks in which tributary trees are used. Yaman et al. (2007) considers a concrete application in cargo delivery systems in which multi-stop access paths visiting more than one O/D node in the way to a hub node are used to route commodities. Camargo et al. (2013), Rodríguez-Martín et al. (2014), Cardoso Lopes et al. (2016), and Kartal et al. (2017) among others, study models arising in freight transportation and express delivery in which collection, transfer or distribution tours have to be designed. The formulation and solution of such complex problems is far more challenging as compared to standard HLPS and even to HALPs.

Other applications, such as in airline and ground transportation, require additional design decisions associated with the nodes and served commodities (Alibeyg et al. 2016, 2018). For example, in the case of airline companies network planners have to design their network when entering into the market, or may have to modify already established networks through alliances, merges and acquisitions. The decisions are to determine the cities that will be part of their network and which O/D flights to activate in order to offer air travel services to passengers between cities so as to maximize the profit.

Finally, another interesting facet of hub networks which has been rarely studied is the integration of network design with scheduling decisions. Yaman et al. (2012) studies a concrete application arising in cargo delivery systems for next-day delivery in which the goal is to simultaneously design a hub network and to decide on the release times of trucks from each demand center so that the total cargo guaranteed to be delivered by the next day is above a threshold while minimizing the flow cost. Masaeli et al. (2018) study another model arising in parcel delivery systems in which the number of dispatches to operate on the hub network as well as the time period of dispatching each vehicle from a hub are taken into account while designing the hub network.

References

Adams, W. P., & Sherali, H. D. (1990). Linearization strategies for a class of zero-one mixed integer programming problems. *Operations Research, 38*, 217–226.

Alibeyg, A., Contreras, I., & Fernández, E. (2016). Hub network design with profits. *Transportation Research Part E: Logistics and Transportation Review 96*, 40–59.

Alibeyg, A., Contreras, I., & Fernández, E. (2018). Exact solution of hub network design problems with profits. *European Journal of Operational Research, 266*, 57–71.

Alumur, S., & Kara, B. Y. (2008). Network hub location problems: The state of the art. *European Journal of Operational Research, 190*, 1–21.

Boland, N., Krishnamoorthy, M., Ernst, A. T., & Ebery, J. (2004). Preprocessing and cutting for multiple allocation hub location problems. *European Journal of Operational Research, 155*, 638–653.

Bryan, D. L., & O'Kelly, M. E. (1999). Hub-and-spoke networks in air transportation: An analytical review. *Journal of Regional Science, 39*, 275–295.

Camargo, R. S., & Miranda, Jr G. (2012). Single allocation hub location problem under congestion: Network owner and user perspectives. *Expert Systems with Applications, 39*, 3385–3391.

Camargo, R. S., Miranda, Jr G., & Ferreira, R. P. M. (2011). A hybrid outer-approximation/Benders decomposition algorithm for the single allocation hub location problem under congestion. *Operations Research Letters, 39*, 329–337.

Camargo, R. S., Miranda, Jr G., & Lokketagen, A. (2013). A new formulation and an exact approach for the many-to-many hub location-routing problem. *Applied Mathematical Modelling, 37*, 12–13.

Camargo, R. S., Miranda, Jr G., O'Kelly, M., & Campbell, J. F. (2017). Formulations and decomposition methods for the incomplete hub location problem with and without hop-constraints. *Applied Mathematical Modelling, 51*, 274–301.

Campbell, J. F. (1992). Location and allocation for distribution systems with transshipments and transportation economies of scale. *Annals of Operations Research, 40*, 77–99.

Campbell, J. F. (1994a). A survey of network hub location. *Studies in Locational Analysis, 6*, 31–43.

Campbell, J. F., Ernst, A. T., & Krishnamoorthy, M. (2001). Hub location problems. In: Z. Drezner, & H. W. Hamacher (Eds.), *Facility location*. Applications and Theory (pp. 373–408). Heidelberg: Springer.

Campbell, J. F., Ernst, A. T., & Krishnamoorthy, M. (2005a). Hub arc location problems: Part I Introduction and results. *Management Science, 51*, 1540–55.

Campbell, J. F., Ernst, A. T., & Krishnamoorthy, M. (2005b). Hub arc location problems: Part II formulations and optimal algorithms. *Management Science, 51*, 1556–71.

Campbell, J. F., & O'Kelly, M. E. (2012). Twenty-five years of hub location research. *Transportation Science, 46*, 153–169.

Cardoso Lopes, M., Eduardo de Andrade, C. E., Alves de Queiroz, T., Resende, M. G. C., & Miyazawa, F. K. (2016). Heuristics for a hub location-routing problem. *Networks, 68*, 54–90.

Contreras, I., Cordeau, J.-F., & Laporte, G. (2011a). Stochastic uncapacitated hub location. *European Journal of Operational Research, 212*, 518–528.

Contreras, I., Cordeau, J.-F., & Laporte, G. (2012). Exact solution of large-scale hub location problems with multiple capacity levels. *Transportation Science, 46*, 439–459.

Contreras, I., Díaz, J. A., & Fernández, E. (2009a). Lagrangean relaxation for the capacitated hub location problem with single assignment. *OR Spectrum, 31*, 483–505.

Contreras, I., Díaz, J. A., & Fernández, E. (2011b). Branch and price for large-scale capacitated hub location problems with single assignment. *INFORMS Journal on Computing, 23*, 41–55.

Contreras, I., & Fernández, E. (2014). Hub location as the minimization of a supermodular set function. *Operations Research, 62*, 557–570.

Contreras, I., Fernández, E., & Marín, A. (2009b). Tight bounds from a path based formulation for the tree of hubs location problem. *Computers & Operations Research, 36*, 3117–3127.

Contreras, I., Fernández, E., & Marín, A. (2010). The tree of hubs location problem. *European Journal of Operational Research, 202*, 390–400.

Contreras, I., O'Kelly, M. E. (2019). Hub location problems. In G. Laporte, S. Nickel & F. Saldanha da Gama (Eds.), *Location science*. Heidelberg: Springer.

Contreras, I., Tanash, M., & Vidyarthi, N. (2017). Exact and heuristic approaches for the cycle hub location problem. *Annals of Operations Research, 258*, 655–677.

Contreras, I., Zetina, C., Jayawasal, S., & Vidyarthi, N. (2021). An exact algorithm for large-scale non-convex quadratic capacitated facility location. Submitted.

Elhedhli, S., & Wu, H. (2010). A Lagrangean heuristic for hub-and-spoke system design with capacity selection and congestion. *INFORMS Journal on Computing, 22*, 282–296.

Ernst, A. T., & Krishnamoorthy, M. (1996). Efficient algorithms for the uncapacitated single allocation p-hub median problem. *Location Science, 4*, 139–154.

Ernst, A. T., & Krishnamoorthy, M. (1998b). Exact and heuristic algorithms for the uncapacitated multiple allocation p-hub median problems. *European Journal of Operational Research, 104*, 100–112.

Gelareh, S., Monemi, R. N., & Nickel, S. (2015). Multi-period hub location problems in transportation. *Transportation Research Part E: Logistics and Transportation Review, 75*, 67–94.

Gelareh, S., & Nickel, S. (2011). Hub location in transportation networks. *Transportation Research Part E: Logistics and Transportation Review, 47*, 1092–1111.

Gelareh, S., Nickel, S., Pisinger, D. (2010). Liner shipping hub network design in a competitive environment.*Transportation Research Part E: Logistics and Transportation Review, 46*, 991–1004.

Hamacher, H. W., Labbé, M., Nickel, S., & Sonneborn, T. (2004). Adapting polyhedral properties from facility to hub location problems. *Discrete Applied Mathematics, 145*, 104–116.

Helme, M. P., & Magnanti, T. L. (1989). Designing satellite communication networks by zero-one quadratic programming. *Networks, 19*, 427–450.

Hu, T. C. (1974). Optimum communication spanning trees. *SIAM Journal on Computing, 3*, 188–195.

Kartal, Z., Hasgul, S., & Ernst, A. T. (2017). Single allocation p-hub median location and routing problem with simultaneous pick-up and delivery. *Transportation Research Part E: Logistics and Transportation Review, 108*, 141–159.

Kim, J.-G., & Tcha, D.-W. (1992). Optimal design of a two-level hierarchical network with tree-star configuration. *Computers & Industrial Engineering, 22*, 273–281.

Klincewicz, J. G. (1998). Hub location in backbone/tributary network design: A review. *Location Science, 6*, 307–335.

Labbé, M., & Yaman, H. (2004). Projecting the flow variables for hub location problems. *Networks, 44*, 84–93.

Labbé, M., & Yaman, H. (2008). Solving the hub location problem in a start-start network. *Networks, 51*, 19–33.

Labbé, M., Yaman, H., & Gourdin, É. (2005). A branch and cut algorithm for hub location problems with single assignment. *Mathematical Programming, 102*, 371–405.

Lari, I., Ricca, F., & Scozzari, A. (2008). Comparing different metaheuristic approaches for the median path problem with bounded length. *European Journal of Operational Research, 20*, 625–637.

Lee, C.-H., Ro, H.-B., & Tcha, D.-W. (1993). Topological design of a two-level network with ring-star configuration. *Computers & Operations Research 20*, 625–637.

Lee, Y., Lim, B. L., & Park, J. S. (1996). A hub location problem in designing digital data service networks: Lagrangian relaxation approach. *Location Science, 4*, 185–194.

Marín, A. (2005a). Uncapacitated Euclidean hub location: Strengthened formulation, new facets and a relax-and-cut algorithm. *Journal of Global Optimization, 33*, 393–422.

Martins de Sá, E., Contreras, I., Cordeau, J.-F., de Camargo, R. S., & de Miranda, R. (2015). The hub line location problem. *Transportation Science, 49*, 500–518.

Martins de Sá, E., de Camargo, R. S., & de Miranda, R. (2013). An improved Benders decomposition algorithm for the tree of hubs location problem. *European Journal of Operational Research, 226*, 185–202.

Masaeli, M., Alumur, S. A., & Bookbinder, J. H. (2018). Shipment scheduling in hub location problems. *Transportation Research Part B: Methodological, 115*, 126–142.

Mirchandani, P. (2000). Projections of the capacitated network loading problem. *European Journal of Operational Research, 122*, 534–560.

O'Kelly, M. E. (1986a). The location of interacting hub facilities. *Transportation Science, 20*, 92–106.

O'Kelly, M. E. (1986b). Activity levels at hub facilities in interacting networks. *Geographical Analysis, 18*, 343–356.

O'Kelly, M. E. (1987). A quadratic integer program for the location of interacting hub facilities. *European Journal of Operational Research, 32*, 393–404.

O'Kelly, M. E., & Miller, H. J. (1994). The hub network design problem: A review and synthesis. *Journal of Transport Geography, 2*, 31–40.

Ortiz-Astorquiza, C., Contreras, I., & Laporte, G. (2015). The minimum flow cost Hamiltonian cycle problem: a comparison of formulations. *Discrete Applied Mathematics, 187*, 140–154.

Pirkul, H., & Schilling, D. A. (1998). An efficient procedure for designing single allocation hub and spoke systems. *Management Science, 44*, 235–242.

Rardin, R. L., & Wolsey, L. A. (1993). Valid inequalities and projecting the multicommodity extended formulation for uncapacitated fixed charge network flow problems. *European Journal of Operational Research, 71*, 95–109.

Rodríguez-Martín, I., Salazar-González, J. J., & Yaman, H. (2014). A branch-and-cut algorithm for the hub location and routing problem. *Computers & Operations Research, 50*,161–174.

Rothenbcher, A.-K., Drexl, M., & Irnich, S. (2016). Branch-and-price-and-cut for a service network design and hub location problem. *European Journal of Operational Research, 255*, 935–947.

Sasaki, M., Campbell, J. F., Krishnamoorthy, M., & Ernst, A. T. (2014). A Stackelberg hub arc location model for a competitive environment. *Computers & Operations Research, 47*, 27–41.

Skorin-Kapov, D., Skorin-Kapov, J., & O'Kelly, M. E. (1997). Tight linear programming relaxations of uncapacitated p-hub median problems. *European Journal of Operational Research, 94*, 582–593.

Tanash, M., Contreras, I., & Vidyarthi, N. (2017). An exact algorithm for the modular hub location problem with single assignments. *Computers & Operations Research, 85*, 32–44.

Xu, J., Chiu, S. Y., & Glover, F. (1999). Optimizing a ring-based private line telecommunication network using tabu search. *Management Science, 45*, 330–345.

Wolsey, L. A. (1983). Fundamental properties of certain discrete location problems. In J. F. Thisse, & H. G. Zoller (Eds.), *Locational analysis of public facilities* (pp. 331–355). Amsterdam: North-Holland.

Yaman, H. (2008). Star p-hub median problem with modular arc capacities. *Computers & Operations Research, 35*, 3009–3019.

Yaman, H., Kara, B. Y., & Tansel, B. Ç. (2007). The latest arrival hub location problem for cargo delivery systems with stopovers. *Transportation Research Part B: Methodological, 41*, 906–919.

Yaman, H., Karasan, O. E., & Kara, B. Y. (2012). Release time scheduling and hub location for next-day delivery. *Operations Research, 60*, 906–917.

Zetina, C., Contreras, I., Fernández, E., & Luna-Mota, C. (2019). Solving the optimum communication spanning tree problem. *European Journal of Operational Research, 273*, 108–117.

Chapter 19
Logistics Network Design

Jean-François Cordeau, Walid Klibi, and Stefan Nickel

1 Introduction

The design of logistics networks is one of the most important areas of application for multicommodity network design models. Logistics networks (or supply chains) connect suppliers, manufacturing plants, warehouses, distribution centers and customers to coordinate the acquisition of raw materials and components, their transformation into finished products and the delivery of these products to the customers. The design of these networks is complex and involves making a large number of interdependent decisions concerning the selection of suppliers, the location of production and distribution facilities, the assignment of products to the facilities, the selection of transportation modes, and the determination of the flows of raw materials, components and finished products in the network. Figure 19.1 illustrates the structure of a classical logistics network with suppliers providing raw materials to production plants which, in turn, deliver finished products to warehouses. Finally, these warehouses are responsible for serving the demand of the customers. Nodes that are not connected to others represent potential facilities that are not currently part of the network.

Logistics network design is of course strategic in nature and concerns the long term. However, it is essential when designing a network to take into account the

J.-F. Cordeau (✉)
HEC Montréal and CIRRELT, Montréal, QC, Canada
e-mail: jean-francois.cordeau@hec.ca

W. Klibi
Kedge Business School, Talence, France
e-mail: walid.klibi@kedgebs.com

S. Nickel
Karlsruhe Institute of Technology, Karlsruhe, Germany
e-mail: stefan.nickel@kit.edu

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_19

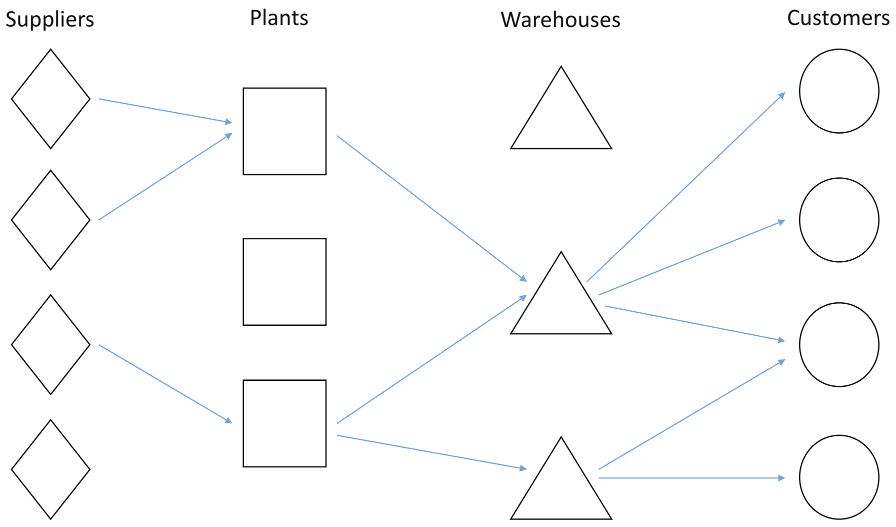


Fig. 19.1 Example of a three-echelon logistics network

tactical and operational repercussions of the strategic decisions. For instance, the selection of proper locations and capacity levels for production plants depends to a large extent on the selection of suppliers, on the choice of which products will be made in each plant, on the transportation modes used to connect the various nodes in the network, and even on the amount of flow circulating on the arcs. Of course, tactical and operational decisions will be revised more frequently as demand and other parameters change in the environment of the firm.

In addition to combining multiple types of intertwined decisions, logistics network design problems usually involve several categories of costs: fixed costs associated with facility locations, acquisition costs for raw materials and components, production costs, transportation costs, inventory holding costs, etc. Not surprisingly, several studies have shown that very important savings can be achieved by taking the design decisions in an integrated way (see, e.g. Arntzen et al. 1995; Fleischmann et al. 2006; Ulstein et al. 2006). Companies often benefit from the re-optimization of their logistics networks following changes in their business environment: new market opportunities, changes in production or transportation costs, new technologies, changes in trade regulations, etc. Mergers and acquisitions also often create the need for a firm to revise the structure of its logistics network. Accordingly, logistics networks are rarely designed from scratch and optimization models usually aim at finding the best way to adapt an existing network to new market conditions.

Over the last 40 years, the realism of logistics network design models has greatly improved and efficient solution methods have been developed to solve these models. There is now a vast literature on the topic with a very large number of models addressing the many problem variants encountered in practice (see, e.g. Melo et al.

2009; Martel and Klibi 2016). The purpose of this chapter is to provide a general modeling framework that can be used to express many of these variants and to give a brief overview of the main solution methodologies. We also devote attention to two important and recent trends: the treatment of risk and uncertainty in the design of logistics networks and the incorporation of environmental, sustainability and reverse logistics aspects.

The rest of the chapter is organized as follows. Section 2 introduces the modeling framework and discusses various extensions. This is followed by a discussion of risk and uncertainty concepts in Sect. 3 and of reverse logistics, environmental and sustainability aspects in Sect. 4. Section 5 is devoted to solution methods. Finally, Sect. 6 provides bibliographic notes and Sect. 7 concludes the chapter.

2 A General Modeling Framework for Logistics Network Design

The purpose of this section is to introduce a general formulation that captures the fundamental aspects of logistics network design and can also serve as a basis to incorporate various extensions that are often required in practical applications. This formulation is itself based on the model proposed by Cordeau et al. (2006) but it has been generalized to consider multiple time periods as well as arbitrary network structures and bills of materials.

2.1 Notation

We denote by $\mathscr{K}$ be the set of all item types circulating in the logistics network. This set can be partitioned into a subset $\mathscr{R}$ of raw materials, a subset $\mathscr{A}$ of assemblies and a subset $\mathscr{F}$ of finished products such that $\mathscr{K} = \mathscr{R} \cup \mathscr{A} \cup \mathscr{F}$. We assume here that raw materials are acquired from external suppliers whereas finished products are delivered to customers. Assemblies are intermediate components that are made either from raw materials or from other assemblies. For every item $k \in \mathscr{A} \cup \mathscr{F}$, let $\mathscr{B}^k \subseteq \mathscr{R} \cup \mathscr{A}$ denote the subset of items that are needed to make item k. Similarly, for every raw material or assembly $\ell \in \mathscr{R} \cup \mathscr{A}$, let $K^\ell \subseteq \mathscr{A} \cup \mathscr{F}$ denote the assemblies and finished products that require item ℓ. For every $k \in \mathscr{K}$ and $\ell \in \mathscr{B}^k$, we denote by $b^{k\ell}$ the amount of item ℓ required for the production of one unit of item k. The set $\mathscr{B}^k$ and values $b^{k\ell}$ define the bill of materials for item k.

The set of potential suppliers is denoted by $\mathscr{S}$ and, for every raw material $r \in \mathscr{R}$, we let $\mathscr{S}^r \subseteq \mathscr{S}$ represent the subset of suppliers that may provide r. We also let $\mathscr{P}$ and $\mathscr{W}$ denote the sets of potential locations for plants and warehouses, respectively. For every item $k \in \mathscr{K}$, we let $\mathscr{P}^k$ denote the subset of plants where item k can be produced or used in the production of other items. Similarly, $\mathscr{W}^k$ represents the warehouses where item k can be stored. Finally, we denote by $\mathscr{C}$ the set of customer locations. In most applications, a customer c would not represent an individual customer or a specific company but rather an aggregation of the demand of a given region.

To simplify the writing of the model, we let $\mathscr{O} = \mathscr{S} \cup \mathscr{P} \cup \mathscr{W}$ and $\mathscr{D} = \mathscr{P} \cup \mathscr{W} \cup \mathscr{C}$ denote the sets of possible origins and destinations for the items circulating in the network. For every $k \in \mathscr{K}$, we also define $\mathscr{O}^k \subseteq \mathscr{O}$ and $\mathscr{D}^k \subseteq \mathscr{D}$ as the sets of possible origins and destinations for item k, respectively. For any node $i \in \mathscr{O}$, let also $\mathscr{K}_i = \{k \in \mathscr{K} | i \in \mathscr{O}^k \cup \mathscr{D}^k\}$ be the set of items which may originate from or be destined to node i.

The planning horizon is divided into a set of consecutive time periods denoted by $\mathscr{T}$. For every $c \in \mathscr{C}$, $f \in \mathscr{F}$ and $t \in \mathscr{T}$, let d_c^{ft} be the demand of customer c for finished product f in period t.

The model uses three types of binary variables to represent decisions related to the selection of nodes in the network and the assignment of items to these nodes. For every node $i \in \mathscr{O}$, we define a binary variable y_i equal to 1 if and only if the node is selected, and we let c_i be the fixed cost of selecting this node. For every item $k \in \mathscr{K}$ and every node $i \in \mathscr{O}^k$, let also v_i^k be a binary variable, with cost c_i^k, taking value 1 if and only if item k is assigned to node i. These variables represent the decisions to acquire raw materials from certain suppliers or to make and store products in certain plants and warehouses. Finally, for every $k \in \mathscr{K}$, $i \in \mathscr{O}^k$ and $j \in \mathscr{D}^k$, let w_{ij}^k be a binary variable, with cost c_{ij}^k, equal to 1 if and only if origin i provides item k to destination j.

For every node $i \in \mathscr{O}$, let q_i be the output capacity of this node, and for every $k \in \mathscr{K}_i$, let u_i^k be the amount of capacity required by one unit of item k at node i. For every $k \in \mathscr{K}$ and $i \in \mathscr{O}^k$, let also q_i^k be the capacity of node i for item k and q_{ij}^k be the maximum that can be provided to destination $j \in \mathscr{D}^k$.

We assume that a set $\mathscr{M}_{ij}$ of potential transportation modes is associated to every origin-destination pair $(i, j) \in \mathscr{O} \times \mathscr{D}$. For every $m \in \mathscr{M}_{ij}$, we then define a binary variable z_{ij}^m equal to 1 if and only if transportation mode m is used between origin i and destination j. We use the notation c_{ij}^m to represent the fixed cost of using mode m and let q_{ij}^m be its capacity in each time period. For every $k \in \mathscr{K}$, $i \in \mathscr{O}^k$ and $j \in \mathscr{D}^k$, $\mathscr{M}_{ij}^k \subseteq \mathscr{M}_{ij}$ is the subset of feasible transportation modes between i and j for item k, and u^{km} is the unit capacity consumption for item k in mode m.

Finally, the model uses two types of continuous variables to represent acquisition, production, storage and transportation decisions. For every $m \in \mathscr{M}_{ij}^k$ and $t \in \mathscr{T}$, we define a non-negative variable x_{ij}^{kmt}, with cost c_{ij}^{kmt}, representing the number of units of item k transported from i to j using mode m in period t. Unit costs c_{ij}^{kmt} should include not only transportation expenses but also relevant acquisition, production and handling costs at the origin node i. This does not cause any loss of generality since the total flow going through a node can be computed as the sum of the flows on the arcs leaving that node. For every item $k \in \mathscr{K}$ and every node $w \in \mathscr{W}^k$, we also let I_w^{kt} denote the inventory of item k in node w at the end of period t and we let

Table 19.1 Summary of notation for sets

$\mathscr{A}$	Set of assemblies
$\mathscr{B}^k$	Set of items needed to make item k
$\mathscr{C}$	Set of customers
$\mathscr{C}^f$	Set of customers that require product f
$\mathscr{D}$	Set of destinations
$\mathscr{D}^k$	Set of potential destinations for item k
$\mathscr{F}$	Set of finished products
$\mathscr{K}$	Set of all items
$\mathscr{K}^\ell$	Set of items that require item ℓ
$\mathscr{K}_i$	Set of items that may originate or be destined to i
$\mathscr{M}_{ij}$	Set of transportation modes between i and j
$\mathscr{M}_{ij}^k$	Set of transportation modes between i and j for item k
$\mathscr{O}$	Set of origins
$\mathscr{O}^k$	Set of potential origins for item k
$\mathscr{P}$	Set of potential plant locations
$\mathscr{P}^k$	Set of plant locations where item k can be produced or used
$\mathscr{R}$	Set of raw materials
$\mathscr{S}$	Set of potential suppliers
$\mathscr{S}^\ell$	Set of potential suppliers providing raw material ℓ
$\mathscr{T}$	Set of time periods
$\mathscr{W}$	Set of potential warehouse locations
$\mathscr{W}^k$	Set of warehouse locations where item k can be stored

g_w^{kt} be the cost of holding one unit of item k at node w in period t. We assume here that inventory is held only at warehouse nodes. If a plant possesses storage areas for assemblies or finished products, it can be modeled by multiple nodes connected among themselves and representing the various functions of the plant.

Tables 19.1, 19.2 and 19.3 provide a summary of the notation.

2.2 Formulation

The *logistics network design problem* (LNDP) consists in minimizing the following objective function, which comprises all the fixed costs associated with the binary design variables and unit costs associated with the flow and inventory variables:

$$\text{Minimize} \sum_{i \in \mathscr{O}} \left[c_i y_i + \sum_{j \in \mathscr{D}} \sum_{m \in \mathscr{M}_{ij}} c_{ij}^m z_{ij}^m \right] +$$

Table 19.2 Summary of notation for parameters

$b^{k\ell}$	Amount of item ℓ needed in one unit of item k
c_i	Fixed cost of selecting origin i
c_i^k	Fixed cost of assigning item k to origin i
c_{ij}^k	Fixed cost of providing item k to destination j from origin i
c_{ij}^m	Fixed cost of using transportation mode m between i and j
c_{ij}^{kmt}	Unit cost for providing item k to j from i with mode m in period t
d_c^{ft}	Demand of customer c for product f in period t
g_w^{kt}	Cost of holding one unit of item k at node w during period t
q_i	Capacity of node i in equivalent units
q_{ij}^m	Capacity of mode m between i and j in equivalent units
q_i^k	Upper limit on the amount of item k shipped from node i
q_{ij}^k	Upper limit on the amount of item k shipped from i to j
u_i^k	Amount of capacity required by one unit of item k at node i
u^{km}	Amount of capacity required by one unit of item k in mode m

Table 19.3 Summary of notation for variables

I_w^{kt}	Inventory of item k in location w at the end of period t
x_{ij}^{kmt}	Amount of item k shipped from i to j with mode m in period t
y_i	$= 1$ if node i is selected
v_i^k	$= 1$ if item k is assigned to origin i
w_{ij}^k	$= 1$ if node i provides item k to destination j
z_{ij}^m	$= 1$ if mode m is selected between i and j

$$\sum_{k\in\mathcal{K}}\sum_{i\in\mathcal{O}^k}\left[c_i^k v_i^k + \sum_{j\in\mathcal{D}^k}\left[c_{ij}^k w_{ij}^k + \sum_{m\in\mathcal{M}_{ij}^k}\sum_{t\in\mathcal{T}} c_{ij}^{kmt} x_{ij}^{kmt}\right]\right] + \sum_{k\in\mathcal{K}}\sum_{w\in\mathcal{W}^k}\sum_{t\in\mathcal{T}} g_w^{kt} I_w^{kt}. \tag{19.1}$$

The first group of constraints comprises equations related to the flow of items in the network:

$$\sum_{i\in\mathcal{O}^\ell}\sum_{m\in\mathcal{M}_{ip}^\ell} x_{ip}^{\ell mt} - \sum_{k\in\mathcal{K}^\ell}\sum_{j\in\mathcal{D}^k}\sum_{m\in\mathcal{M}_{pj}^k} b^{k\ell} x_{pj}^{kmt} = 0 \qquad \ell\in\mathcal{R}\cup\mathcal{A};\ p\in\mathcal{P}^\ell;\ t\in\mathcal{T} \tag{19.2}$$

$$\sum_{i\in\mathcal{O}^k}\sum_{m\in\mathcal{M}_{iw}^k} x_{iw}^{kmt} - \sum_{j\in\mathcal{D}^k}\sum_{m\in\mathcal{M}_{wj}^k} x_{wj}^{kmt} + I_w^{k,t-1} - I_w^{kt} = 0 \qquad k\in\mathcal{K};\ w\in\mathcal{W}^k;\ t\in\mathcal{T} \tag{19.3}$$

$$\sum_{i \in \mathscr{O}^f} \sum_{m \in \mathscr{M}_{ic}^f} x_{ic}^{fmt} = d_c^{ft} \quad f \in \mathscr{F}; c \in \mathscr{C}^f; t \in \mathscr{T}. \tag{19.4}$$

Constraints (19.2) force the amount of raw material or assembly ℓ shipped to plant p in period t to be equal to the amount required by all assemblies and finished products made at this plant during the same period. Constraints (19.3) ensure that the amount of item k entering warehouse w during period t plus the inventory available at the beginning of period t is equal to the amount leaving the warehouse during that period plus the amount available at the end of the period. To ensure consistent inventory levels, the last period in the horizon can be connected to the first one, which has the effect of forcing the inventory level at the end of the last period to be equal to the inventory level at the beginning of the first one. Demand constraints are imposed by Eqs. (19.4).

The second group of constraints comprises inequalities related to the capacity of the nodes and arcs in the network:

$$\sum_{k \in \mathscr{K}} \sum_{j \in \mathscr{D}^k} \sum_{m \in \mathscr{M}_{ij}^k} u_i^k x_{ij}^{kmt} - q_i y_i \leq 0 \quad i \in \mathscr{O}; t \in \mathscr{T} \tag{19.5}$$

$$\sum_{j \in \mathscr{D}^k} \sum_{m \in \mathscr{M}_{ij}^k} x_{ij}^{kmt} - q_i^k v_i^k \leq 0 \quad k \in \mathscr{K}; i \in \mathscr{O}^k; t \in \mathscr{T} \tag{19.6}$$

$$\sum_{m \in \mathscr{M}_{ij}^k} x_{ij}^{kmt} - q_{ij}^k w_{ij}^k \leq 0 \quad k \in \mathscr{K}; i \in \mathscr{O}^k; j \in \mathscr{D}^k; t \in \mathscr{T} \tag{19.7}$$

$$\sum_{k \in \mathscr{K}} u^{km} x_{ij}^{kmt} - q_{ij}^m z_{ij}^m \leq 0 \quad i \in \mathscr{O}; j \in \mathscr{D}; m \in \mathscr{M}_{ij}; t \in \mathscr{T}. \tag{19.8}$$

Constraints (19.5) impose aggregate capacity limits on suppliers, plants and warehouses, whereas limits for individual items are enforced through (19.6). Constraints (19.7) force node i to be selected if units of item k are transported from i to j. Finally, capacity constraints for individual transportation modes are represented by (19.8).

It should be noted that single-sourcing for item k at destination j can be imposed with the constraint

$$\sum_{i \in \mathscr{O}^k} w_{ij}^k \leq 1. \tag{19.9}$$

Furthermore, the following constraint can impose a single-assignment rule to ensure that all units of an item k come from the same origin:

$$\sum_{i \in \mathscr{O}^k} v_i^k \leq 1. \tag{19.10}$$

2.3 *Extensions*

The above formulation captures the essential aspects of the LNDP but it can also be generalized to address a variety of practical situations. For the sake of clarity, each extension is presented separately although the different extensions can obviously be combined.

2.3.1 Lower Bounds and Capacity Alternatives

Lower limits on acquisition, production, storage and transportation activities can be imposed by using constraints similar to (19.5)–(19.8) but reversing the inequality sign. In particular, lower bounds can be used to model quantity discounts or any other situation where a minimal volume is necessary for a given unit cost to be applicable. The combined use of lower and upper bounds can also serve to model situations where several capacity alternatives, with different operating costs, exist for the configuration of a node in the network. This also applies to different layouts or configurations of facilities.

2.3.2 Multi-Period Design Decisions

The above model assumes that all binary decisions are made at the beginning of the planning horizon and that the network structure thus remains the same throughout this horizon. However, facility location decisions are usually highly dependent on the value of some parameters such as the fixed costs of the facilities and the customer demand. If these parameters are expected to vary over time, it may be desirable to plan in advance for future adjustments in the number and location of facilities and in other related decisions. In this case, locating a set of facilities becomes a question not only of "where" but also of "when". This can be achieved by associating a time index to the binary design decisions and by adapting constraints (19.5)–(19.8) to reflect the fact that network configuration and facility capacity evolve over time. In particular, let y_{it} indicate whether the facility at node i is open and operating in period t. Simply adding the time index would allow for frequent opening and closing of facilities, which does not adequately reflect an implementable evolution of a real-world supply chain network. To depict the network evolution of a growing enterprise, constraints (19.11) ensure that facilities that have once been openend remain open throughout the planning horizon:

$$y_{i,t-1} - y_{it} \le 0 \qquad i \in \mathscr{I};\, t \in \mathscr{T} \setminus \{1\}. \tag{19.11}$$

When planning the optimal evolution of an existing supply chain network, the successive closing of existing facilities becomes important. In such phase-in/phase-out models the set of nodes $\mathscr{I}$ can be partitioned into two subsets: $\mathscr{I}^C$, the set of locations where existing facilities can be removed, and $\mathscr{I}^O$, the set of locations where new facilities can be installed. To ensure consistency in the resulting network configuration constraints (19.11) are replaced with constraints (19.12) and (19.13):

$$y_{i,t-1} - y_{it} \le 0 \qquad i \in \mathscr{I}^O;\, t \in \mathscr{T} \setminus \{1\} \tag{19.12}$$

$$y_{it} - y_{i,t-1} \le 0 \qquad i \in \mathscr{I}^C;\, t \in \mathscr{T} \setminus \{1\}. \tag{19.13}$$

The explicit consideration of opening and closing costs for facilities in this context was first introduced by Wesolowsky and Truscott (1975).
Multi-period models are sometimes called dynamic (location) models in the literature. Note that this is somewhat misleading because design decisions can only be made at specific moments, namely at the beginning of each indexed period. Some models go even further and, mostly in an attempt to reduce complexity, divide the set of periods $\mathscr{T}$ into strategic and tactical periods, whereby changes in the network configuration are restricted to the strategic periods. A multi-period modeling framework involves one extra dimension in the decision space: the timing. Hence, the resulting models tend to be large and harder to solve, even for instances of moderate size. Accordingly, one may ask whether it is worth considering this extra dimension instead of making static decisions even though costs, demands and other parameters may vary over time. An answer to this question can be given by the *value of the multi-period solution*, a concept first introduced by Alumur et al. (2012) in the context of a multi-period reverse logistics network design problem. The value of the multi-period solution compares the optimal value of the multi-period problem and the value of a solution found by solving a static counterpart. We refer the interested reader to Nickel and Saldanha-da-Gama (2015) for an in-depth discussion of multi-period facility location models and for details on the value of the multi-period solution.

2.3.3 Inventory Level Constraints

Lower and upper bounds on inventory levels in warehouses can be added to the model by introducing constraints on the I_w^{kt} variables. In particular, an upper bound $\hat{q}_w^{kt}$ on the amount of item k held in inventory in warehouse w at the end of period t can be imposed with the following constraint:

$$I_w^{kt} \le \hat{q}_w^{kt} v_w^k. \tag{19.14}$$

Similarly, an upper bound $\hat{q}_w^t$ on the total amount of inventory held in warehouse w at the end of period t can be imposed with the following constraint:

$$\sum_{k \in \mathcal{K}} u_w^k I_w^{kt} \leq \hat{q}_w^t y_w. \tag{19.15}$$

Because the LNDP is a strategic planning problem defined over a long planning horizon, the length of the time periods usually does not allow for a detailed representation of operational inventory decisions. Nevertheless, constraints can be imposed on expected safety stocks and cyclic (replenishment) inventory levels by relying on turnover ratios. Let ρ_w^{kt} denote the expected turnover ratio for item k at warehouse w in period t. A lower bound on the total inventory level at the end of period t can be imposed as follows:

$$I_w^{kt} \geq \sum_{d \in D^k} \sum_{m \in \mathcal{M}_{wd}^k} \frac{1}{\rho_w^{kt}} x_{wd}^{kmt}. \tag{19.16}$$

The actual inventory level can of course be larger than the lower bound when fluctuations in demand make it beneficial to accumulate inventory in some periods to be used in later ones. It is well known in inventory control that, because of pooling effects, the amount of safety stock needed increases less than linearly with the amount of demand served by a warehouse. To capture this non-linear relationship between demand volume and operational inventory levels, one may define several copies of the same warehouse with different turnover ratios for the given product. Better approximations of the concave relationship between throughput and inventory levels can be obtained by using a continuous piece-wise linear function specified as a set of base levels (the equivalent of a fixed cost) and unit rates of increase (the equivalent of a unit cost). Mathematically, this relationship can be imposed by using the constraint

$$I_w^{kt} \geq \alpha_w^{kt} + \sum_{d \in \mathcal{D}^k} \sum_{m \in \mathcal{M}_{wd}^k} \beta_w^{kt} x_{wd}^{kmt}, \tag{19.17}$$

where α_w^{kt} represents the base level and β_w^{kt} is the unit increase rate. This will lead to the type of approximation represented in Fig. 19.2.

It should be observed that by setting $\alpha_w^{kt} = 0$ for every segment, one obtains the first approximation (19.16) based only on turnover ratios. Furthermore, if a single segment is used, the value of β_w^{kt} should correspond to the inverse of the expected turnover ratio. If this value is estimated with respect to the maximum possible throughput, one will obtain the approximation illustrated in Fig. 19.3.

We refer to Martel (2005) for a detailed treatment of inventory representation in LNDPs.

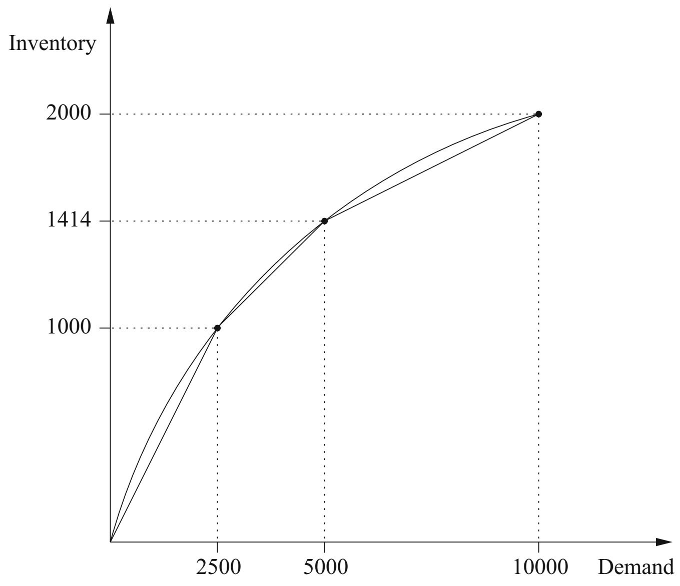
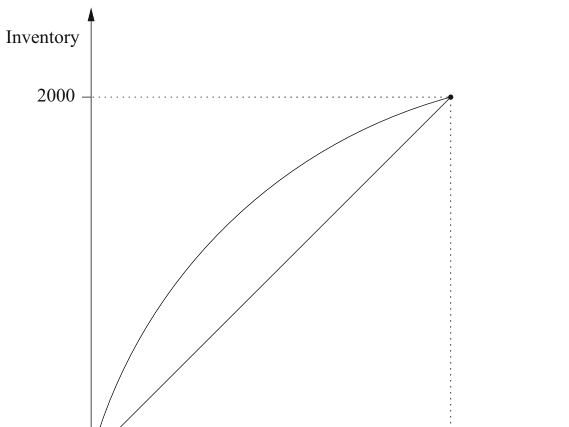


Fig. 19.2 Inventory levels with piece-wise linear segments

Fig. 19.3 Inventory levels with a single segment



2.3.4 Profit Maximization

The traditional focus in logistics network design is to minimize costs while satisfying an exogenous demand. In a value-creation paradigm, however, it may be more appropriate to consider a profit maximization objective function that captures both costs and revenues. Model (1)–(8) can be modified in different ways to account for profit maximization. A common approach is to consider the demand as a decision variable. This can be accomplished by associating a unit revenue r_c^f with product f and customer c and by replacing equality constraints (19.4) with two inequalities imposing minimum and maximum demand levels to be served in each market as follows:

$$\sum_{i \in \mathscr{O}^f} \sum_{m \in \mathscr{M}_{ic}^f} x_{ic}^{fmt} \geq \bar{d}_c^{ft} \quad f \in \mathscr{F}; c \in \mathscr{C}^f; t \in \mathscr{T} \tag{19.18}$$

$$\sum_{i \in \mathscr{O}^f} \sum_{m \in \mathscr{M}_{ic}^f} x_{ic}^{fmt} \leq \hat{d}_c^{ft} \quad f \in \mathscr{F}; c \in \mathscr{C}^f; t \in \mathscr{T}. \tag{19.19}$$

Here, $\bar{d}_c^{ft}$ and $\hat{d}_c^{ft}$ represent, respectively, the minimum amount of demand to be served and the maximum potential demand of customer c for product f in period t. Setting both parameters to the same value corresponds to imposing the equality constraints (19.4). The following term should be added to the objective function to measure total revenue:

$$\sum_{t \in \mathscr{T}} \sum_{c \in \mathscr{C}} \sum_{f \in \mathscr{F}} \sum_{i \in \mathscr{O}^f} \sum_{m \in \mathscr{M}_{ic}^f} r_c^f x_{ic}^{fmt}.$$

The latter expression assumes that unit revenue is constant, which makes the objective function linear and preserves model tractability. However, if demand is assumed to depend on price and price is itself treated as a decision variable, then the objective function becomes non-linear and makes the problem harder to solve. The above approach also assumes that demand is independent of the network design. In practice, sales are often affected not only by price but also by the design of the logistics network itself because it has an impact on various aspects of service such as response time, i.e., the time it takes to deliver a product to the customer. Figure 19.4 illustrates how, for a given price, the revenues (R^S) and costs (C^S) may depend on the response time S provided by the logistics network and, consequently, how the economic value added by the network can be expressed as the difference between revenue and cost. It also shows that beyond a given response time S^{max} revenues can decline abruptly.

This suggests another approach to address profit maximization: one can perform a sensitivity analysis based on the price and response time variations. For given values of price and response time, the demand can be estimated and the revenues calculated a priori. Then, the cost minimization model can be solved after removing from the model the variables that correspond to customer assignments that would violate the target response time S. Varying the price and response time allows one to approximate the two curves shown in Fig. 19.4 and to identify the value of S for which the difference P^S between R^S and C^S is maximized.

Finally, a more sophisticated approach to treat profit maximization is to model the set of potential market policies offered by a company through binary policy selection variables. A market policy specifies the price, desired response time and other

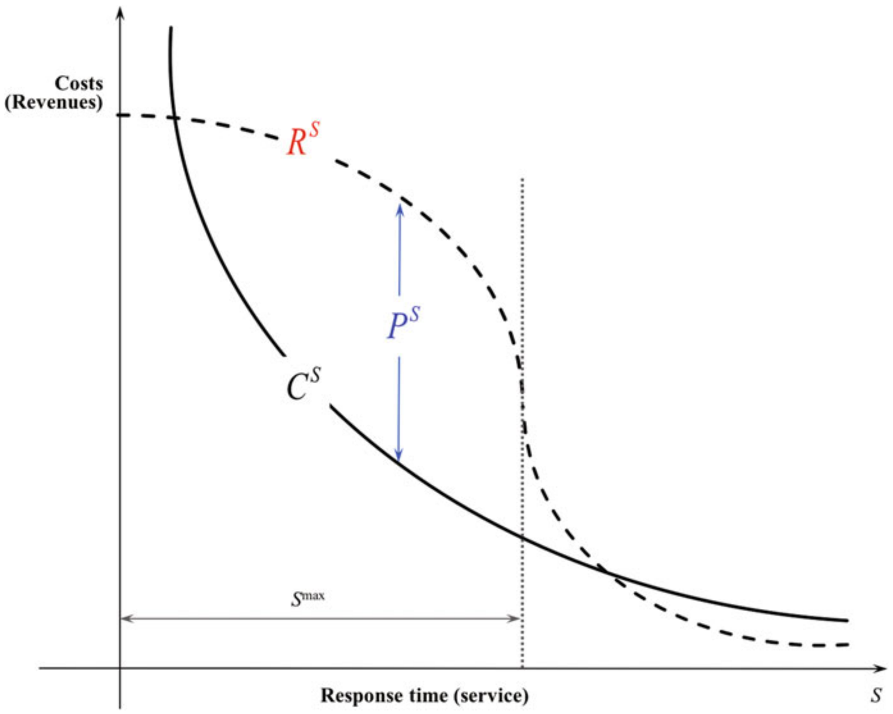


Fig. 19.4 Economic value added for a given logistics network design

attributes, and is characterised by a fixed implementation cost, service constraints and demand bounds. This approach is explained in more detail by Martel and Klibi (2016).

2.3.5 International Aspects

Some aspects related to international operations can easily be taken into consideration with this model. For example, exchange rates should be used to convert monetary values into a unique, common currency. In addition, tariffs and duties for products that cross a border can be added directly to the cost of the corresponding arcs. Local content rules can often be enforced in the form of lower bounds on a sum of arc flow variables. However, more complex questions such as transfer pricing and taxation require the introduction of additional variables and constraints. In particular, transfer pricing usually leads to non-linear formulations because of the need to determine both costs and flows simultaneously on some arcs of the networks. A detailed treatment of international aspects is beyond the scope of this chapter and we instead refer to Arntzen et al. (1995), Martel (2005) and to Martel and Klibi (2016).

3 Risk and Uncertainty

Because logistics network design decisions concern the long term, several of the input parameters are often subject to risk and uncertainty. Several modeling approaches can be considered to take this uncertainty into account. In this section, we explain how two of these approaches, stochastic programming and robust optimization, can be applied to the LNDP.

3.1 Stochastic Programming

Stochastic programming is a well-known optimization technique to address mathematical programs in which some of the data are random variables. It assumes that the probability distributions of the random parameters are known a priori. One can distinguish between two main types of models: (1) stochastic programs with recourse that explicitly model recourse decisions to hedge against uncertainty, and (2) chance-constrained programs that impose restrictions on the probability that a constraint is violated due to stochasticity. Here, we limit ourselves to the case of stochastic programming with recourse, which is the most common approach in the context of logistics network design.

Referring to formulation (1)–(8) of the LNDP, the aim is to incorporate notions of cost, demand and capacity uncertainty in the model. When all design decisions are made at once, the problem can be formulated as a two-stage stochastic program with recourse. When several design periods are considered, one obtains a multi-stage program, which is more difficult to solve. Under a two-stage setting, some decisions are made in the first stage before the uncertain information is known. In the second stage, the values of the random variables become known and the recourse actions are taken. In our context, the first stage could, for example, correspond to fixing all of the binary design variables. The second stage would consist in choosing the flows and resulting inventory levels in the network and also in choosing the short-term actions required to match supply and demand (i.e., use of additional capacity, subcontracting, or overtime). Uncertainty should thus be restricted to parameters that affect only the flow variables in connection with demand, unit costs and capacities. However, fixed costs associated with the binary variables should be deterministic. Structural information such as bills of materials should also be known with certainty. As a basic rule of thumb, one may consider that information that varies from period to period in the deterministic model can be considered random in the stochastic program, while information that is not period-dependent should be deterministic.

Let $\mathscr{H}$ denote the set of possible scenarios and, for each scenario $h \in \mathscr{H}$, denote by p_h the probability of scenario h being realized. Each scenario represents a realization of the vector of uncertain parameters. When continuous probability distributions are considered, the set $\mathscr{H}$ is implicitly infinite. However, a subset of scenarios can be generated by a sampling method. The main modification required to the demand, cost, and capacity parameters is the addition of a scenario index h. For every scenario h, let x_{ij}^{kmth} and I_w^{kth} denote the flows and inventory levels under scenario h. Using this notation, the generic LNDP model (19.1)–(19.8) can be transformed into the following two-stage stochastic program with recourse:

$$\text{Minimize} \sum_{i\in\mathscr{O}}\left[c_i y_i + \sum_{j\in\mathscr{D}}\sum_{m\in\mathscr{M}_{ij}} c_{ij}^m z_{ij}^m\right] + \sum_{k\in\mathscr{K}}\sum_{i\in\mathscr{O}^k}\left[c_i^k v_i^k + \sum_{j\in\mathscr{D}^k} c_{ij}^k w_{ij}^k\right] + \mathbb{E}_{\mathscr{H}}\left[Q\left(\mathbf{y},\mathbf{z},\mathbf{v},\mathbf{w},h\right)\right], \tag{19.20}$$

where $Q\left(\mathbf{y},\mathbf{z},\mathbf{v},\mathbf{w},h\right)$ is the optimal value of the second-stage program and $\mathbb{E}_{\mathscr{H}}[\cdot]$ denotes the expectation with respect to the scenario set $\mathscr{H}$. For a given scenario h and given values $\bar{\mathbf{y}}$, $\bar{\mathbf{z}}$, $\bar{\mathbf{v}}$, $\bar{\mathbf{w}}$ of the first-stage decisions, the second-stage problem can be expressed as follows:

$$\text{Minimize} \sum_{t\in\mathscr{T}}\sum_{k\in\mathscr{K}}\left[\sum_{i\in\mathscr{O}^k}\sum_{j\in\mathscr{D}^k}\sum_{m\in\mathscr{M}_{ij}^k} c_{ij}^{kmth} x_{ij}^{kmth} + \sum_{w\in\mathscr{W}^k} g_w^{kth} I_w^{kth}\right] \tag{19.21}$$

subject to

$$\sum_{i\in\mathscr{O}^\ell}\sum_{m\in\mathscr{M}_{ip}^\ell} x_{ip}^{\ell mth} - \sum_{k\in\mathscr{K}^\ell}\sum_{j\in\mathscr{D}^k}\sum_{m\in\mathscr{M}_{pj}^k} b^{k\ell} x_{pj}^{kmth} = 0 \quad \ell\in\mathscr{R}\cup\mathscr{A};\, p\in\mathscr{P}^\ell;\, t\in\mathscr{T} \tag{19.22}$$

$$\sum_{i\in\mathscr{O}^k}\sum_{m\in\mathscr{M}_{iw}^k} x_{iw}^{kmth} - \sum_{j\in\mathscr{D}^k}\sum_{m\in\mathscr{M}_{wj}^k} x_{wj}^{kmth} + I_w^{k,t-1,h} - I_w^{kth} = 0 \quad k\in\mathscr{K};\, w\in\mathscr{W}^k;\, t\in\mathscr{T} \tag{19.23}$$

$$\sum_{i\in\mathscr{O}^f}\sum_{m\in\mathscr{M}_{ic}^f} x_{ic}^{fmth} = d_c^{fth} \quad f\in\mathscr{F};\, c\in\mathscr{C}^f;\, t\in\mathscr{T} \tag{19.24}$$

$$\sum_{k\in\mathscr{K}}\sum_{j\in\mathscr{D}^k}\sum_{m\in\mathscr{M}_{ij}^k} u_i^k x_{ij}^{kmth} - q_i^h \bar{y}_i \le 0 \quad i\in\mathscr{O};\, t\in\mathscr{T} \tag{19.25}$$

$$\sum_{j\in\mathscr{D}^k}\sum_{m\in\mathscr{M}_{ij}^k} x_{ij}^{kmth} - q_i^{kh}\bar{v}_i^k \le 0 \quad k\in\mathscr{K};\, i\in\mathscr{O}^k;\, t\in\mathscr{T} \tag{19.26}$$

$$\sum_{m\in\mathscr{M}_{ij}^k} x_{ij}^{kmth} - q_{ij}^{kh}\bar{w}_{ij}^k \le 0 \quad k\in\mathscr{K};\, i\in\mathscr{O}^k;\, j\in\mathscr{D}^k;\, t\in\mathscr{T} \tag{19.27}$$

$$\sum_{k \in \mathcal{K}} u^{km} x_{ij}^{kmth} - q_{ij}^{mh} z_{ij}^{m} \leq 0 \quad i \in \mathcal{O};\ j \in \mathcal{D};\ m \in \mathcal{M}_{ij};\ t \in \mathcal{T}. \tag{19.28}$$

In the presence of uncertainty, it may not be possible to fully satisfy the customer demand under every scenario. To ensure that the second stage problem is always feasible, one may introduce in the demand constraints (19.24) recourse variables a_c^{fth} representing the amount of demand not satisfied for customer c and product f in period t under scenario h. These variables can be appended to the objective function with a recourse cost of e_c^{fth} per unit to represent the cost of the recourse needed when demand cannot be fully satisfied. Alternatively, one may introduce recourse variables in the capacity constraints if the demand should always be satisfied in full but possible recourse actions consist in acquiring extra capacity through subcontracting, overtime or any other means.

3.2 Robust Optimization

The above approach can be used to deal with general forms of uncertainty concerning demand, cost and capacity parameters. It assumes, however, that information is available on the likelihood of each scenario and that planners are risk-neutral. Under these assumptions, minimizing the expected cost is perfectly reasonable. In practice, however, decision-makers are often risk-averse and they may care more about the worst-case cost than the expected cost. In addition, rare events such as natural disasters or terrorist attacks do not have well-defined probabilities of occurrence. Even if they do, the probabilities are usually very small and will not have any real impact on the optimal solution to the problem although the corresponding events may have dramatic consequences.

One way to overcome these limitations of stochastic programming is to use robust optimization. In classical robust optimization, one is interested in finding a solution that minimizes the cost under the worst possible scenario. This leads to a min-max objective function, which is often seen as too pessimistic because it assumes that all uncertain parameters can take their worst possible value at the same time. An interesting alternative to worst-case optimization is the "budget-of-uncertainty" approach of Bertsimas and Sim (2004). This approach assumes that the number of uncertain parameters that can deviate from their nominal value or the sum of these deviations is bounded from above by a value known as the budget of uncertainty. This approach generally leads to more balanced solutions that are less sensitive to extreme scenarios. However, it still puts the focus on the worst-case at the expense of average performance.

A possibly more appropriate approach in practice is to combine the idea of average cost optimization with some notion of robust optimization in the same model. For example, risk aversion can be taken into account by adding to the objective function of the two-stage stochastic program an extra term measuring the worst-case cost with respect to the different possible scenarios. The problem then becomes parametric because one must define the relative weight of this extra term in the objective function.

Let $\mathbb{D}_{\mathscr{H}}\left[C(\mathbf{y},\mathbf{z},\mathbf{v},\mathbf{w},h)\right]$ denote a risk measure that the decision-makers want to consider in the selection process of robust solutions. Shapiro et al. (2009) identified a set of coherent risk measures that satisfy a number of convexity, monotonicity, translation equivalence, and positive homogeneity properties in stochastic programming. For instance the mean absolute upper semi-deviation from the mean is a common downside risk measure which can be written as follows:

$$\mathbb{D}_{\mathscr{H}}\left[C(\mathbf{y},\mathbf{z},\mathbf{v},\mathbf{w},h)\right] = \mathbb{E}_{\mathscr{H}}\left[(C(\mathbf{y},\mathbf{z},\mathbf{v},\mathbf{w},h) - \mathbb{E}_{\mathscr{H}}\left[C(\mathbf{y},\mathbf{z},\mathbf{v},\mathbf{w},h)\right])^{+}\right], \tag{19.29}$$

where $(x)^{+} = \max\{0, x\}$. Alternatively, a worst case measure is given by the maximum upper semi-deviation:

$$\mathbb{D}_{\mathscr{H}}\left[C(\mathbf{y},\mathbf{z},\mathbf{v},\mathbf{w},h)\right] = \max_{\mathscr{H}}\left[(C(\mathbf{y},\mathbf{z},\mathbf{v},\mathbf{w},h) - \mathbb{E}_{\mathscr{H}}\left[C(\mathbf{y},\mathbf{z},\mathbf{v},\mathbf{w},h)\right])^{+}\right]. \tag{19.30}$$

With either of these measures, one can transform the objective function (19.21) into the following weighted sum:

$$\text{Minimize } \mathbb{E}_{\mathscr{H}}\left[C(\mathbf{y},\mathbf{z},\mathbf{v},\mathbf{w},h)\right] + \omega\mathbb{D}_{\mathscr{H}}\left[C(\mathbf{y},\mathbf{z},\mathbf{v},\mathbf{w},h)\right], \tag{19.31}$$

where $\omega \geq 0$ is a scaling parameter.

Finally, one can also use the risk measures (19.29) or (19.30) to impose a constraint on risk inside a stochastic program. This is achieved by adding the constraint

$$\mathbb{D}_{\mathscr{H}}\left[C(\mathbf{y},\mathbf{z},\mathbf{v},\mathbf{w},h)\right] \leq \omega^{0}, \tag{19.32}$$

where ω^{0} corresponds to the upper bound on risk tolerance.

4 Reverse Logistics, Environmental Aspects and Sustainability

While traditional logistics network design has focused on forward logistics, i.e., the movement of goods from suppliers to end customers, there is a growing interest in both the scientific literature and the industry for the design of reverse logistics networks to manage upstream flows from end customers back to the plants and even to the suppliers. The logistics network for recovery and revalorization of used products can take several forms, depending on the industrial context. This could be a network composed by a second-hand market independent of the original equipment manufacturer (OEM) or an internal network based on demand for parts or used products by the OEM's manufacturing or remanufacturing activities. We note that when forward and reverse flows are coupled the resulting problem is referred to as a closed-loop logistics network design problem (Akçalı et al. 2009; Easwaran and Uster 2010).

The decisions related to the design of a reverse logistics network may involve the determination of the optimal locations and capacities of collection centers, inspection centers, remanufacturing facilities, and recycling plants in addition to the optimal shipment strategies between these facilities. There usually are various options available including repair, refurbishing, and recycling of products as well as alternatives such as inspection, disassembly, disposal, or selling to suppliers, to the secondary market or to external remanufacturing facilities. Different actors and facilities are also involved in reverse logistics networks, e.g., disposers, remanufacturers, and the secondary market. Moreover, unlike forward networks, which are mostly driven by economic considerations, there are further factors motivating the establishment of reverse logistics networks such as environmental laws and regulations (Mota et al. 2014). Finally, uncertainty is also prevalent because the supply of returned products is often highly unpredictable. Hence, reverse LNDPs are usually quite complex.

Items that are shipped in a reverse logistics network include used, repaired, or refurbished products, as well as components or raw materials of such products. The set of items $\mathcal{K}$ must thus account for different states (used, repaired, refurbished) of the same product. The transitions between the stages of products at nodes of the network as well as the reverse bills of materials need to be considered when modeling these problems. The most important type of constraints in reverse logistics network design models is the flow balance constraints. Flow balance needs to account for the total amount of products recovered at a location as well as the transition between different states of the product through various recovery options. For example, a used product may turn into a refurbished product at a remanufacturing facility. Another important issue to consider within the flow balance constraints of a reverse logistics network design model is the reverse bills of materials. A product may be decomposed into its components at a disassembly facility. Let $\delta^{\ell k}$ denote the amount of item k obtained by recovering one unit of item ℓ. One needs to define a new decision variable r_p^{kt} representing the amount of item k recovered at location p in period t. Assuming that recovery takes one period of time, the flow conservation constraint for item k at recovery plant p in period t can then be formulated as follows:

$$\sum_{i \in \mathcal{O}^k} \sum_{m \in \mathcal{M}_{ip}^k} x_{ip}^{kmt} - \sum_{j \in \mathcal{D}^k} \sum_{m \in \mathcal{M}_{pj}^k} x_{pj}^{kmt} + \sum_{\ell \in \mathcal{K}} \delta^{\ell k} r_p^{\ell,t-1} - r_p^{kt} = 0. \tag{19.33}$$

As noted above, the major driving forces in reverse logistics networks include not only economic factors, but also legislation and environmental consciousness.

Hence, there can be constraints associated with recovery targets. An example of such a constraint would be:

$$\sum_{p\in\mathscr{P}}\sum_{\tau=1}^{t} r_p^{k\tau} \geq RT^{kt} \quad k\in\mathscr{K}; t\in\mathscr{T}, \tag{19.34}$$

where RT^{kt} denotes the recovery target of item k up until the end of period t. This constraint forces the recovery target of each item to be met by the end of each period.

In terms of the objective function, it is common to consider profit maximization in reverse LNDPs rather than cost minimization. As noted in Alumur et al. (2012), a company could fully outsource its reverse logistics operations if the only motivation is to satisfy legislation or regulations. Moreover, there are usually multiple actors involved in the design and operation of a reverse logistics network in addition to those involved in a forward network. These multiple stakeholders include producers, distributors, third-party logistics providers, disposers, and municipalities. Multiple actors may obviously lead to multi-objective decision problems.

In practice, the estimation of greenhouse gas (GHG) emissions may be difficult because they cover upstream and downstream activities that are not under the control of a single company. For GHG, the common measure is the CO2-equivalent emissions (in kgCO2e/unit load) that could apply to inbound flows associated to suppliers and to outbound flows to include transportation emissions to customers. Accordingly, there is a growing preoccupation for the incorporation of environmental constraints in the design of forward and reverse logistics networks (Mota et al. 2014). Certain types of constraints can be imposed easily in the LNDP. For example, a limit on CO_2 emissions can be treated by imposing aggregate constraints on the sum of flows in the network. If e_i^k and e_{ij}^{km} denote the emissions produced by the flow (e.g., the production or storage) of one unit of item k at node i and the transportation of one unit of item k from node i to node j with mode m, then an upper bound E_t on total emissions in period t can be imposed with the following constraint:

$$\sum_{i\in\mathscr{O}}\sum_{j\in\mathscr{D}}\sum_{k\in\mathscr{K}}\sum_{m\in\mathscr{M}_{ij}^k}(e_i^k + e_{ij}^{km})x_{ij}^{kmt} \leq E_t. \tag{19.35}$$

In addition to environmental efficiency, there is a growing focus on social sustainability (Tang and Zhou 2012; Giusti et al. 2019). However, characterizing and measuring social well-being in logistics is still in its infancy. Employment is often cited as the main social indicator in connection with regional development (Mota et al. 2014). Other social objectives can take the form of an equity concern related to space, returns, or production factors and can be expressed in the LNDP as equity constraints in terms of lower and upper bounds.

5 Solution Methods

Because the LNDP is usually formulated as a mixed-integer program (MIP), it is often solved by general-purpose branch-and-bound or branch-and-cut solvers. However, the formulations are sometimes very large and solving them to near-optimality can be a challenge, even for state-of-the-art solvers. Hence, decomposition methods are often used to separate the problem into smaller and more tractable components. Several heuristic algorithms have also been developed to identify good solutions in reasonable time. This section provides an overview of the main solution methods.

5.1 Exact Algorithms

MIP formulations such as the one provided in Sect. 2.2 can be solved successfully by branch-and-cut for moderate size instances. However, like many network design problems, these formulations tend to have large integrality gaps caused by the presence of fixed costs associated with the binary design decisions. When facilities or arcs in the network are capacitated, linear programming (LP) solutions tend to be very fractional and a significant amount of branching is required to reach an optimal integer solution. The performance of branch-and-bound algorithms can be improved by strengthening the LP relaxations through the addition of valid inequalities, either directly in the formulation, or in the form of cuts in a branch-and-cut algorithm. General families of inequalities, such as cutset inequalities, can often be used directly or can be adapted to the special network structure considered. For example, the simple inequalities $v_i^k \leq y_i, \forall i \in \mathscr{O}$, have been shown to considerably strengthen the LP relaxation of formulation (1)–(8). Cordeau et al. (2006) provide several other families of valid inequalities. Even with the addition of valid inequalities, large-scale instances of the LNDP can remain formidably difficult to solve. Hence, several authors have turned to decomposition methods.

5.1.1 Lagrangian Relaxation

If one relaxes demand and flow conservations (2)–(4), the resulting subproblem becomes separable by origin node. This relaxation has been successfully exploited by some authors. For example, Pirkul and Jayaraman (1998) observed that by relaxing the demand constraints and the flow conservation constraints at the warehouses, their formulation decomposed into a set of independent continuous knapsack problems, one for each warehouse and each plant. In the same way, Hinojosa et al. (2008) relaxed the demand and flow conservations constraints connecting the distribution levels in a two-echelon warehouse location model with multiple commodities. The resulting subproblem decomposes by echelon, by facility and by time period. More recently, Pimentel et al. (2013) used a Lagrangian heuristic procedure to solve a stochastic LNDP, whereas Badri et al. (2013) used Lagrangian relaxation for a deterministic variant. In general, the weakness of these approaches is that they rarely provide feasible integer solutions. Hence, Lagrangian heuristics are usually necessary to produce good feasible solutions.

5.1.2 Benders Decomposition

The most popular decomposition approach for the LNDP is Benders decomposition. By keeping all binary design decisions in the master problem, one obtains a linear and continuous subproblem which usually takes the form of a capacitated multicommodity network flow problem. Although this problem is not separable by commodity, it is nevertheless much easier to solve than the original formulation. When the number of binary design decisions is small but the number of arcs in the network and the number of commodities is large, this decomposition can be very beneficial. The use of Benders decomposition to solve variants of the LNDP was investigated, among others, by Dogan and Goetschalckx (1999) and by Cordeau et al. (2006). Santoso et al. (2005) considered several acceleration techniques to solve a stochastic variant of the LNDP within a sample average approximation (SAA) framework. In particular, the use of a trust region for the master problem together with knapsack inequalities, cut strengthening and cut disaggregation was shown to improve performance. More recently, Mariel and Minner (2017) also used Benders decomposition in a heuristic way to solve a problem with a bilinear objective function arising in the context of supply chain design under NAFTA local content requirements. It is worth noting that in a scenario-based formulation of the LNPD, the decomposition scheme can decompose the subproblem by scenario and allow the use of parallelism.

5.2 Heuristic Algorithms

Several authors have observed that the LP relaxation solution sometimes contains many location variables that naturally take value 0 or 1. This has led to the idea of gradually rounding the LP solution into an integer one. For example, Thanh et al. (2010) solve a sequence of LPs by fixing some binary decisions at each step until all binary variables are integer or the remaining MIP can be solved to optimality. The idea of rounding the LP solution was also used by Melo et al. (2014), who proposed four rounding strategies followed by a local search algorithm to repair infeasibility or improve the resulting integer solution.

An interesting idea when designing heuristics for LNDPs is to explore the space of the integer design variables while relying on a general-purpose LP solver to set the values of the continuous variables to their optimal values. This idea was exploited, for example, by Melo et al. (2012) who use tabu search for a dynamic facility location problem. Their heuristic explores the space of the binary facility location variables while the remaining (continuous) variables are set by solving an LP. A rounding procedure is used to compute an initial solution and infeasible solutions that violate the budget constraints are allowed during the search. The neighborhood considered consists of solutions obtained by changing the status of a single facility at a time. The idea was also used by Carle et al. (2012) who described an agent-based metaheuristic for multi-period LNDPs with an explicit calculation of inventory levels. This metaheuristic combines tabu search procedures with iterated local search and mixed-integer programming to separately and iteratively optimize different components of the problem. Cordeau et al. (2008) devised an iterated local search heuristic for a special case of the problem with single assignment. Under single assignment constraints, the problem becomes purely combinatorial in nature and it can be stated directly with just the binary variables. Hence, the impact of opening or closing a facility or of changing the product assignment can be easily computed, which allows a fast exploration of a solution's neighborhood. Finally, an interesting approach is to take advantage of the strategic-tactical structure of the LNDP model to devise a hierarchical or nested heuristic approach. This allows decoupling the location and allocation decisions from the product flows and inventory decisions, and solving the corresponding parts of the problem with an appropriate exact or heuristic algorithm (see, for instance, You and Grossmann 2008 and Klibi et al. 2010a).

6 Bibliographical Notes

The design of logistics networks is rooted in discrete facility location (Daskin 2011; Laporte et al. 2016) and most early models were direct extensions of capacitated or uncapacitated facility location problems with fixed costs (Aikens 1985; ReVelle and Eiselt 2005). One of the first papers addressing the design of multicommodity distribution networks is that of Geoffrion and Graves (1974). Over the years, many models have been introduced with a focus on improving the realism and comprehensiveness of the problem setting. In particular, several models have been introduced to combine production and distribution decisions (Cohen and Lee 1989; Vidal and Goetschalckx 1997; Elhedhli and Goffin 2005).

An important area of research concerns the incorporation of international aspects in supply chain design. Arntzen et al. (1995) introduced a formulation capturing duties, duty drawback, local content rules and offset requirements. This was followed by Vidal and Goetschalckx (2001) and Goetschalckx et al. (2002) who took into account issues of transfer pricing. More recently, Mariel and Minner (2017) have modeled the problem of locating plants and planning production and distribution under the North American Free Trade Agreement local content rules.

The most comprehensive and realistic modeling framework described to date in the scientific literature is probably that of Martel (2005). The objective function aims to maximize after tax net revenue by taking into account the impact of delivery times on demand. Complex product structures with arbitrary bills of materials are considered along with facility layout and capacity options. A detailed representation of inventory levels is embedded in the formulation as well as a rather rich cost structure. Finally, many aspects of transfer prices, taxes, tariffs and duties are taken into account. The resulting formulation contains nonlinear terms in the objective function but it can be solved iteratively by using piecewise linear approximations of these terms that are updated each time the problem is solved.

Another important stream of literature concerns the dynamic location of facilities over a multiple-period planning horizon. Multi-period models were proposed, among others, by Martel (2005), Melo et al. (2006), Thanh et al. (2008), Hinojosa et al. (2008), and Jena et al. (2015). Melo et al. (2006) have introduced a general framework that not only supports facility relocation or shutdown but also capacity expansion and reduction under a budget constraint in each time period. This framework is mostly targeted at distribution network design as it does not consider supplier selection or the transformation of raw materials into finished products. Another multi-period model for supply chain design was introduced by Thanh et al. (2008). Facilities can be opened and closed during the planning horizon and modular capacities are considered for each facility. They also consider both seasonal and cyclic stocks. Finally, the work of Klibi and Martel (2013) provides a modeling framework for logistics network design under uncertainty. The problem is cast as a two-level organizational decision (strategic–operational) and is characterized by multiple design cycles and multiple planning periods.

Beside time aspects, the treatment of risk and uncertainty related to logistics networks is of inherent importance for realistic models (Dunke et al. 2018). Surprisingly the literature lacks a clear distinction between the notion of risk and uncertainty. An in-depth study of the notion of supply chain risk can be found in Heckmann et al. (2015). An approach for modelling risk and for the generation of scenarios in the LNDP can be found in Klibi and Martel (2012). Finally, in Heckmann and Nickel (2017) a more general discussion of common flaws in supply chain risk analysis is presented.

The application of robust optimization techniques to LNDPs is still a relatively new area of research. An interesting comparison and application of different formulations to a case study can be found in Govindan and Fattahi (2017). Robust optimization approaches were proposed to deal with simple location problems (Snyder and Daskin 2006). We refer to Klibi et al. (2010b) for a broader discussion of robustness issues in the design of logistics networks. With respect to uncertainty, Santoso et al. (2005) and Schutz et al. (2009) proposed to apply the sample average approximation (SAA) method coupled with Benders decomposition and dual decomposition, respectively, to solve stochastic LNDPs. For more details and background on stochastic programming in the context of logistics network design the reader is referred to Correia and Saldanha-da-Gama (2015), Fan et al. (2017), and to Govindan et al. (2017).

There is a quickly growing literature on green logistics network design and the incorporation of sustainability objectives and constraints. Two classes of problems can be distinguished: reverse logistics network and closed-loop logistics network design problems. The recent work of Mota et al. (2014) is linked to the former class with the establishment of reverse logistics networks including environmental laws and regulations. The work of Chaabane et al. (2012) considers life cycle assessment principles and emission trading schemes to design sustainable logistics network. The work of Akçalı et al. (2009) and Easwaran and Uster (2010) is oriented towards closed-loop network design problems. We also refer to Tang and Zhou (2012) for a discussion of social sustainability and to Garcia and You (2015) for the inclusion of sustainability issues in multi-objective optimization of LNDPs.

7 Conclusions and Perspectives

Logistics network design problems are challenging combinatorial optimization problems with widespread applications and a high potential for impact in terms of cost reduction and performance improvement for companies involved in the manufacturing and distribution of goods. The field has attracted a lot of attention in the operations research community and the models and algorithms currently available can solve instances of reasonable size with a sufficient degree of realism. Nevertheless, there still exist many opportunities for improvement.

One of the main challenges is the treatment of uncertainty. Although it is easy to formulate two-stage or multi-stage versions of the LNDP, solving these models remains extremely difficult. Because the problems involve a large number of uncertain parameters, one must consider large sets of scenarios to obtain a sufficient degree of precision in the representation of uncertainty. Decomposition methods such as Benders decomposition can be used to obtain some form of separability but convergence to an optimal solution is usually very slow.

Another area that could largely benefit from additional research is the integration of forward and reverse logistics network design decisions into so-called "closed-loop" supply chains. With environmental regulations becoming more stringent in most countries, there is a growing need to design forward networks that can also handle reverse flows in an effective way from the start. The resulting problems are again highly complex and difficult to solve.

Finally, we would like to point out that the growing importance of on-line commerce is slowly changing the way logistics networks are designed. The focus is now placed less on cost minimization through economies of scale and more on revenue maximization through improved quality of service to the end customers. The reorganization of distribution, in the retail sector in particular, gives rise to multi-channel distribution network structures and to a mix between fulfillment and storage strategies.

References

Aikens, C. (1985). Facility location models for distribution planning. *European Journal of Operational Research, 22*, 263–279.

Akçalı, E., Çetinkaya, S., & Uster, H. (2009). Network design for reverse and closed-loop supply chains: An annotated bibliography of models and solution approaches. *Networks, 53*(3), 231–248.

Alumur, S. A., Nickel, S., Saldanha-da-Gama, F., & Verter, V. (2012). Multi-period reverse logistics network design. *European Journal of Operational Research, 220*, 67–78.

Arntzen, B., Brown, G., Harrison, T., & Trafton, L. (1995). Global supply chain management at Digital Equipment Corporation. *Interfaces, 25*(1), 69–93.

Badri, H., Bashiri, M., & Hejazia, T. (2013). Integrated strategic and tactical planning in a supply chain network design with a heuristic solution method. *Computers & Operations Research, 40*(4), 1143–1154.

Bertsimas, D., & Sim, M. (2004). The price of robustness. *Operations Research, 52*(1), 35–53.

Carle, M. A., Martel, A., & Zufferey, N. (2012). The CAT metaheuristic for the solution of multi-period activity-based supply chain network design problems. *International Journal of Production Economics, 139*(2), 664–677.

Chaabane, A., Ramudhin, A., & Paquet, M. (2012). Design of sustainable supply chains under the emission trading scheme. *International Journal of Production Economics, 135*, 37–49.

Cohen, M., & Lee, H. (1989). Resource deployment analysis of global manufacturing and distribution networks. *Journal of Manufacturing and Operations Management, 2*, 81–104.

Cordeau, J. F., Laporte, G., & Pasin, F. (2008). An iterated local search heuristic for the logistics network design problem with single assignment. *International Journal of Production Economics, 113*(2), 626–640.

Cordeau, J. F., Pasin, F., & Solomon, M. (2006). An integrated model for logistics network design. *Annals of Operations Research, 144*, 59–82.

Correia, I., & Saldanha-da-Gama, F. (2015). Facility location und uncertainty. In G. Laporte, S. Nickel, & F. Saldanha-da-Gama (Eds.), *Location science* (pp. 177–203). Berlin: Springer.

Daskin, M. S. (2011). *Network and discrete location: Models, algorithms, and applications*. New York: Wiley.

Dogan, K., & Goetschalckx, M. (1999). A primal decomposition method for the integrated design of multi-period production-distribution systems. *IIE Transactions, 31*, 1027–1036.

Dunke, F., Heckmann, I., Nickel, S., & Saldanha-da-Gama, F. (2018). Time traps in supply chains: Is optimal still good enough? *European Journal of Operational Research, 264*, 813–829.

Easwaran, G., Uster, H. (2010). A closed-loop supply chain network design problem with integrated forward and reverse channel decisions. *IIE Transactions, 42*, 779–792.

Elhedhli, S., & Goffin, J. L. (2005). Efficient production-distribution system design. *Management Science, 51*(7), 1151–1164.

Fan, Y., Schwartz, F., Voß, S., & Woodruff, D. L. (2017). Stochastic programming for flexible global supply chain planning. *Flexible Services and Manufacturing Journal, 29*, 601–633.

Fleischmann, B., Ferber, S., & Henrich, P. (2006). Strategic planning of BMW's global production network. *Interfaces, 36*, 194–208.

Garcia, D. J., & You, F. (2015). Supply chain design and optimization: Challenges and opportunities. *Computers & Chemical Engineering, 81*, 153–170.

Geoffrion, A., & Graves, G. (1974). Multicommodity distribution system design by Benders decomposition. *Management Science, 20*, 822–844.

Giusti, R., Iorfida, C., Li, Y., Manerba, D., Musso, S., Perboli, G., et al. (2019). Sustainable and de-stressed international supply-chains through the synchro-net approach. *Sustainability, 11*(4), 1083.

Goetschalckx, M., Vidal, C., & Dogan, K. (2002). Modeling and design of global logistics systems: A review of integrated strategic and tactical models and design algorithms. *European Journal of Operational Research, 143*, 1–18.

Govindan, K., & Fattahi, M. (2017). Investigating risk and robustness measures for supply chain network design under demand uncertainty: A case study of glass supply chain. *International Journal of Production Economics, 183*, 680–699.

Govindan, K., Fattahi, M., & Keyvanshokooh, E. (2017). Supply chain network design under uncertainty: A comprehensive review and future research directions. *European Journal of Operational Research, 263*, 108–141.

Heckmann, I., Comes, T., & Nickel, S. (2015). A critical review on supply chain risk–definition, measure and modeling. *Omega, 52*,119–132.

Heckmann, I., & Nickel, S. (2017). Rethinking supply chain risk analysis – common flaws & main elements. *Supply Chain Forum, 18*, 84–95.

Hinojosa, Y., Kalcsics, J., Nickel, S., Puerto, J., & Velten, S. (2008). Dynamic supply chain design with inventory. *Computers & Operations Research, 35*(2), 373–391.

Jena, S. D., Cordeau, J. F., & Gendron, B. (2015). Dynamic facility location with generalized modular capacities. *Transportation Science, 49*(3), 484–499.

Klibi, W., Lasalle, F., Martel, A., & Ichoua, S. (2010a). The stochastic multiperiod location transportation problem. *Transportation Science, 44*(2), 221–237.

Klibi, W., & Martel, A. (2012). Scenario-based supply chain network risk modeling. *European Journal of Operational Research, 223*, 644–658.

Klibi, W., & Martel, A. (2013). The design of robust value-creating supply chain networks. *OR Spectrum, 35*(4), 867–903.

Klibi, W., Martel, A., & Guitouni, A. (2010b). The design of robust value-creating supply chain networks: a critical review. *European Journal of Operational Research, 203*(2), 283–293.

Laporte, G., Nickel, S., & Saldanha da Gama, F. (2016). *Location science*. New York: Springer.

Mariel, K., & Minner, S. (2017). Benders decomposition for a strategic network design problem under NAFTA local content requirements. *Omega, 68*, 62–75.

Martel, A. (2005). The design of production-distribution networks: A mathematical programming approach. In J. Geunes, & P. Pardalos (Eds.), *Supply chain optimization*. New York: Springer.

Martel, A., & Klibi, W. (2016). *Designing value-creating supply chain networks*. New York: Springer.

Melo, M., Nickel, S., & Saldanha-da-Gama, F. (2006). Dynamic multi-commodity capacitated facility location: a mathematical modeling framework for strategic supply chain planning. *Computers & Operations Research, 33*(1), 181–208.

Melo, M., Nickel, S., & Saldanha-da-Gama, F. (2009). Facility location and supply chain management – A review. *European Journal of Operational Research, 196*(2), 401–412.

Melo, M., Nickel, S., & Saldanha-da-Gama, F. (2012). A tabu search heuristic for redesigning a multi-echelon supply chain network over a planning horizon. *International Journal of Production Economics,136*(1), 218–230.

Melo, M. T., Nickel, S., & Saldanha-da-Gama, F. (2014). An efficient heuristic approach for a multi-period logistics network redesign problem. *TOP, 22*(1), 80–108.

Mota, B., Gomes, M., Carvalho, A., & Barbosa-Povoa, A. (2014). Towards supply chain sustainability: Economic, environmental and social design and planning. *Journal of Cleaner Production, 105*, 14–27.

Nickel, S., & Saldanha-da-Gama, F. (2015). Multi-period facility location. In G. Laporte, S. Nickel, & F. Saldanha-da-Gama (Eds.), *Location science* (pp. 289–310). Berlin: Springer.

Pimentel, B. S., Mateus, G. R., & Almeida, F. A. (2013). Stochastic capacity planning and dynamic network design. *International Journal of Production Economics, 145*, 139–149.

Pirkul, H., & Jayaraman, V. (1998). A multi-commodity, multi-plant, capacitated facility location problem: Formulation and efficient heuristic solution. *Computers & Operations Research, 25*, 869–878.

ReVelle, C. S., & Eiselt, H. A. (2005). Location analysis: A synthesis and survey. *European Journal of Operational Research, 165*, 1–19.

Santoso, T., Ahmed, S., Goetschalckx, M., & Shapiro, A. (2005). A stochastic programming approach for supply chain network design under uncertainty. *European Journal of Operational Research, 167*, 96–115.

Schutz, P., Tomasgard, A., & Ahmed, S. (2009). Supply chain design under uncertainty using sample average approximation and dual decomposition. *European Journal of Operational Research, 199*, 409–419.

Shapiro, A., Dentcheva, D., & Ruszczynski, A. (2009). *Lectures on stochastic programming*. New York, USA: SIAM and MPS, USA.

Snyder, L., & Daskin, M. (2006). Stochastic p-robust location problems. *IIE Transactions, 38*(11), 971–985.

Tang, C., & Zhou, S. (2012). Research advances in environmentally and socially sustainable operations. *European Journal of Operational Research, 223*, 585–594.

Thanh, P. N., Bostel, N., & Peton, O. (2008). A dynamic model for facility location in the design of complex supply chains. *International Journal of Production Economics, 113*(2), 678–693.

Thanh, P. N., Peton, O., & Bostel, N. (2010). A linear relaxation-based heuristic approach for logistics network design. *Computers & Industrial Engineering, 59*(4), 964–975.

Ulstein, N., Christiansen, M., Grønhaug, R., Magnussen, N., & Solomon, M. (2006). Elkem uses optimization in redesigning its supply chain. *Interfaces, 36*, 314–325.

Vidal, C., & Goetschalckx, M. (1997). Strategic production-distribution models: A critical review with emphasis on global supply chain models. *European Journal of Operational Research, 98*, 1–18.

Vidal, C., & Goetschalckx, M. (2001). A global supply chain model with transfer pricing and transportation cost allocation. *European Journal of Operational Research, 129*, 134–158.

Wesolowsky, G., & Truscott, W. (1975). The multi-period location-allocation problem with relocation of facilities. *Management Science, 22*, 57–66.

You, F., & Grossmann, I. E. (2008). Design of responsive supply chains under demand uncertainty. *Computers & Chemical Engineering, 32*, 3090–3111.

Chapter 20
Collaboration in Transport and Logistics Networks

Behzad Hezarkhani, Marco Slikker, and Tom Van Woensel

1 Introduction

Transport and logistics companies invested substantially to increase the efficiency of their individual operations. Research has also been fruitful in finding ways to optimize problems of planning routes, scheduling deliveries, designing networks, and deploying resources. It is generally well-understood that economy of scale in transportation and logistics plays a crucial role in increasing efficiency. Yet, efforts towards internal optimization cannot always increase the economies of scale for organizations beyond their operational scope. This is problematic as the logistics and transportation sector is fragmented and many operators of different sizes are present. It is no wonder then that the logistics sector suffers from low overall efficiency—for example, more than 20% for all truck movements in Europe is completely empty and the remainder is hardly ever full.

The success of new network design approaches, building on concepts, models and methodologies such as the Physical Internet, City Logistics, synchromodal networks, etc., is also to a large part depending upon the ability to successfully collaborate and agree on these cost-and-benefit sharing mechanisms. Collaboration is a way to open possibilities for achieving these important economies of scale needed for a successful implementation any Physical Internet or City Logistics solution. In fact, collaboration may positively affect many aspects. For example, by consolidating their loads, carriers can increase their service level and reduce their

B. Hezarkhani
Brunel Business School, Brunel University, London, UK
e-mail: Behzad.Hezarkhani@brunel.ac.uk

M. Slikker · T. Van Woensel (✉)
School of Industrial Engineering, Eindhoven University of Technology, Eindhoven, The Netherlands
e-mail: M.Slikker@tue.nl

T. G. Crainic et al. (eds.), *Network Design with Applications to Transportation and Logistics*, https://doi.org/10.1007/978-3-030-64018-7_20

total costs. Carriers could increase the utilization rate of their assets when combining their delivery demands. Finally, as a result of consolidated cargo and combined trips, the socio-environmental problems of transport and logistics can also be mitigated.

Despite the clear advantages of collaborative logistics, in practice, cooperation and collaboration among organizations are exceptions. Collaboration among carriers is often hampered by their competitive positions and by the risks of divulging information and losing customers. Shippers, on the other hand, may hesitate to collaborate as they might not have a clear understanding of collaborative mechanisms employed and whether or not they receive a fair share out of collaborative operations. Finally, designing a fair cost sharing scheme is a major impediment for collaboration.

This type of collaboration problems falls in the area of cooperative game theory, where coalitions and their respective cost sharing issues are researched. The intention of this chapter is not to give an exhaustive review of cost sharing problems, but to provide an overview of relevant approaches in dealing with cost sharing problems for collaboration in the setting of logistics network design problems.

By abstracting a cooperative situation into a cooperative game, consisting of a player set and a function that determines the cost of different groups of players, cooperative game theory studies solutions that satisfy collections of logically desirable properties expressed in relation to such an abstraction. The players in these situations require or provide transportation-based logistics services. The cost of groups of players is obtained via a network design optimization problem. The specific features of cooperative situations under study provide grounds for refining well-known solutions in cooperative game theory or develop new ones that are appropriate for special situations.

Important to note is that cooperative games are build on stylized situations. A situation is a description of the real-life problem to handle (e.g., network optimization or service network design). However, for these situations, we need to obtain the exact value of possible coalitions (e.g., players working together). From an Operations Research perspective, many of these underlying situations are combinatorial problems, leading to significant calculation times to obtain the relevant (optimal) values. That is why a large body of cooperative game theory literature is build around stylized models. Clearly, solutions in cooperative game context can prove to be unsatisfactory in more complex situations.

This chapter is build around three parts. In the first part, we discuss the most important components around cooperation within a transport and logistics network setting. In the second part, we discuss cost-sharing problems in some basic and stylized network design models. The simplicity of underlying situations in this category allows for adoption of well-known game theoretic solutions such as the core (Shapley 1955) and the Shapley value (Shapley 1953). The search for the core of cooperative games in network situations has motivated a large body of literature, and implementation of the Shapley value is suggested by a host of research in collaborative logistics. In the third part, we look at more operational problems in collaborative logistics and overview the cooperative truckload delivery situations where logistics providers jointly devise plans for their daily pickup/transport/deliver operations. We discuss desirable properties for allocations rules in these situations and introduce an appropriate one for these situations.

This chapter is organized as follows. Section 2 discusses the key concepts revolving around collaboration in transport and logistics networks. Section 3 provides some background and preliminaries on cooperative game theory and the relevant main concepts. In Section 4, we discuss the cost sharing problem in stylized cooperative network design problems, in particular, minimum cost spanning tree, facility location, and hub location. In Section 5, we turn our attention to designing logistics service networks and focus on cooperative truckload delivery situations. Section 6 concludes the chapter.

2 Key Collaboration Concepts in Transport and Logistics Networks

Transport and Logistics networks collaboration involves different aspects: Communication, Coordination, and Consolidation. Many different actors are involved in Transport and Logistics activities. One way of reducing costs is to consolidate activities, e.g. freight consolidation or capacity consolidation, as such reducing empty mileage or under-filled resources. But, these stakeholders hardly communicate with each other, let alone that there is a form of coordination.

Over the past years, more and more different types of collaboration emerged. **Vertical collaboration**, getting popular in the 90s, involves collaboration within the supply chain, i.e. connecting the upstream and downstream partners. This lead to concepts like Vendor Managed Inventory (VMI), factory gate pricing, Collaborative Planning Forecasting and Replenishment (CPFR), and Efficient Consumer Response (ECR). At this moment, these concepts were mainly focused on costs efficiency in the different key supply chain decision areas like inventory, transportation, forecasting, etc. Early 200, next to costs efficiency, companies also started to consider other drivers like sustainability and greenhouse gas emissions. Also in transport and logistics, continued observations on low vehicle utilisations, and a large number of empty running vehicles, lead to strong understanding that collaboration could be a solution towards costs reductions but also to significant reductions in the environmental pressure.

Next to vertical collaboration, **horizontal collaboration** started to gain momentum over the past 10 years. Here, collaboration in distribution and coordination among similar stakeholders, e.g. logistics service providers or shippers, is the focus. The rationale is that bundling of physical good flows into (urban) areas, results in fewer negative impacts (decongestion, less negative externalities in cities). Clearly, Transport and Logistics networks are large constructs of multiple many-to-many interconnected stakeholders, active in both horizontal and vertical relations.

Cruijssen et al. (2007) investigated the opportunities and obstacles carriers face in horizontal collaborations. They organized a spectrum of collaboration types from basically no collaboration (i.e. “arms length”) to a full integration, which is similar to a merger of companies. In between these two extremes, three different levels (denoted as Type I, II and III) are distinguished. **Type I** consists of partners who know and trust each other. They coordinate their activities and planning on a limited basis. The collaboration partnership may be short-term and a single division of each company may focus on one single activity. **Type II** collaboration maintains a longer collaborative relationship. The scope of collaboration for the participants is not only to coordinate, but also to integrate part of their business planning. The horizon is of a long though finite length and multiple divisions or functions of the companies are involved. **Type III** collaboration refers to those organizations which have a significant level of integration, and each company treat others as an extension of its own business unit. There is no end date for this kind of collaboration.

Other collaboration (Communication, Coordination, and Consolidation) concepts also arise in other Transport and Logistics networks fields. Again aiming to reduce vehicle movements and/or increase utilization, crowd logistics is a sharing economy concept. Unorganized individuals (the *crowd*) offer their services (e.g. movement or capacity) to the platform. In this setup, transportation is outsourced to the crowd or *crowdsourced*. Efficient use of different transportation modes, enabled by the use of standardized containers, presents a challenge. Synchro-modality as structured, efficient and synchronic combination of two or more transportation modes also brings interesting collaboration issues, as it also involves multiple stakeholders (i.e. modalities). In these concepts, issues around pricing, revenue and cost sharing are abundantly around.

These logistics processes can also be transformed to the *Physical Internet (PI)* paradigm. This PI acts as an autonomously managed network with nodes (locations where freight is collected, transferred or delivered) and flows (transport movements). For each request, a specific path from the origin to the destination through the network is determined, using standardized transport unit (e.g. containers). A number of prerequisites for successful Physical Internet implementations are real-time monitoring within dispatching systems, integrated in an information-sharing platform, high-level advanced predictions of the future supply of transport movements and advanced collaborative decision support systems, including pain-and-gain sharing mechanisms.

3 Cost Sharing: Preliminaries

Consider a situation wherein a set of players (partners) collaborate among themselves to improve upon their joint costs. The cost sharing problem in a situation entails finding ways to allocate the joint costs among the players. A solution to a cost sharing problem indicates appropriate ways to do the latter.

We distinguish between two alternative approaches to solve cost sharing problems. The first approach (α) defines a cooperative game associated with the situation and uses cooperative game theory to come up with allocations and/or cost-shares. A cooperative game among a set of players is defined by the joint costs of collaboration among the grand coalition as well as all sub-coalitions. The second approach (β) deals directly with the situation at hand and obtains cost-shares using the information contained in the situation. In this approach, the solution often relies on the underlying optimization problems.

Situations can be either more succinct or more expressive than their associated games. Cooperative games explicitly describe the costs of every sub-coalition, while these costs do not appear explicitly in the underlying situation. In this regard, a situation may present relevant justifications for a certain solution that cannot be devised just by focusing on costs of sub-coalitions. However, as the game theoretical solutions abstract away the details of underlying situations, they provide a generic framework to tackle cost sharing problems. In the remainder of this section, we introduce some 5 notions from cooperative game theory. Figure 20.1 illustrates the two approaches possible to cost sharing problems.

3.1 Cooperative Cost Games

A **cooperative game** is a pair (N, c) consisting of a player set $N = \{1, \ldots, n\}$ and a **characteristic cost function** c which assigns to every group of players $S \subseteq N$, hereafter a **coalition**, the cost $c(S) \in \mathbb{R}$. For the empty set we fix $c(\emptyset) = 0$.

The cooperative game (N, c) is **subadditive** if for every two disjoint coalitions S and T, i.e., $S, T \subseteq N$ with $S \cap T = \emptyset$, we have

$$c(S \cup T) \leq c(S) + c(T).$$

If a game is subadditive, then the cost of a combination of disjoint coalitions are always at most as much as the sum of their stand-alone costs so cooperation among players could be beneficial. We focus our attention in this chapter on subadditive games.

The cooperative game (N, c) is **concave** if for every S and T with $S \subset T \subset N$ and every $i \in N \setminus T$ we have

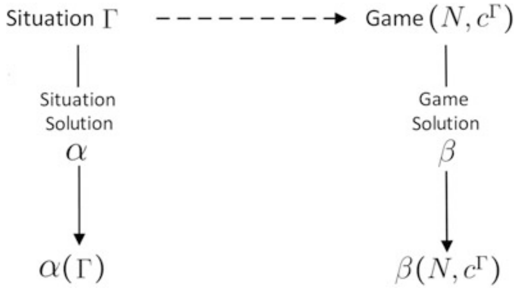

$$\begin{tikzcd}
\text{Situation } \Gamma \arrow[r, dashed] \arrow[d, "\text{Situation Solution } \alpha"] & \text{Game } (N, c^\Gamma) \arrow[d, "\text{Game Solution } \beta"] \\
\alpha(\Gamma) & \beta(N, c^\Gamma)
\end{tikzcd}$$

Fig. 20.1 A situation Γ, its associated game (N, c^Γ), and two approaches to cost sharing

$$c(S \cup \{i\}) - c(S) \geq c(T \cup \{i\}) - c(T).$$

Concavity of a game implies that the marginal cost of adding a new player to a larger coalition is non-increasing.

Example 1 Consider a cooperative game among three players, $N = \{1, 2, 3\}$. The costs for various coalitions are as follows. For $S \subseteq N$ we have: $c(S) = 10$ if $|S| = 1$, $c(S) = 19$ if $|S| = 2$, and $c(S) = 24$ if $|S| = 3$. Compared to the sum of their stand-alone costs, two-player coalitions save one and the grand coalition saves a total of 6. The game is sub-additive. It is also concave—for instance, the marginal cost of adding player 1 to player 2 is 9 units and the marginal cost of adding player 1 to the coalition of players 2 and 3 is 5 units.

The example above motivates an alternative approach in defining cooperative games. For every cooperative cost game (N, c) there exists a dual **cost-savings game** (N, v) where for every $S \subseteq N$: $v(S) = \sum_{i \in S} c(\{i\}) - c(S)$. The characteristic function in a savings game gives the amount of savings that can be made in coalitions compared to the stand-alone costs of the players involved.

Let $a_i \in \mathbb{R}$ be the cost-share of player $i \in N$. An **allocation** $a = (a_i)_{i \in N}$ is a vector of cost-shares for all players. A basic set of properties can be defined to reflect appropriate conditions that allocations should satisfy. Let (N, c) be an arbitrary but fixed game for the rest of this section.

An allocation a satisfies the **Efficiency property** if $\sum_{i \in N} a_i = c(N)$. With an efficient allocation, the entire cost of the grand coalition is shared among the players so that no excess or shortage occurs.

An allocation a satisfies the **Individual Rationality property** if for every $i \in N$ we have $a_i \leq c(\{i\})$. If an allocation fails to satisfy the individual rationality property, then some players would be better off not collaborating.

Two players $i, j \in N$ are substitutable if $c(S \cup \{i\}) = c(S \cup \{j\})$ for all $S \subseteq N \setminus \{i, j\}$. An allocation a satisfies the **Symmetry property** if for every pair of substitutable players $i, j \in N$ it holds that $a_i = a_j$. This property reflects a basic fairness feature, that is, for two players that are identical in contributions to costs, their cost-shares must be equal as well.

Example 2 In Example 1, both allocations $a = (8, 8, 8)$ and $(6, 9, 10)$ satisfy efficiency, and individually rationality. Only the former allocation satisfies the symmetry property.

3.2 Solutions for Cooperative Cost Games

Let $\mathscr{G}$ be the set of all cooperative cost games. Let $\mathscr{G}' \subseteq \mathscr{G}$ be a subset of all cooperative cost games. A (game) **solution** on $\mathscr{G}'$ is a set-valued function β that determines a set of allocations for every cooperative cost game in $\mathscr{G}'$. A solution

β on $\mathscr{G}'$ is called **single-valued** if $|\beta(N,c)| = 1$ for every $(N,c) \in \mathscr{G}'$. For any single-valued solution β on $\mathscr{G}'$ we refer to the function that assigns to any game $(N,c) \in \mathscr{G}'$ the unique element in $\beta(N,c)$ as an **allocation rule**.

We introduce some of the well-known solutions for cooperative games.

3.2.1 Core

The individual rationality property can be extended over all coalitions of players by requiring that the sum of cost-shares of players in every coalition be at most as much as the characteristic cost of that coalition. An allocation a is **stable** for the game $(N,c) \in \mathscr{G}$ if for every $S \subseteq N$ we have $\sum_{i \in S} a_i \leq c(S)$. The **core** of game $(N,c) \in \mathscr{G}$ is the set of all efficient and stable allocations. That is,

$$\mathscr{C}(N,c) = \left\{ a \in \mathbb{R}^n \middle| \sum_{i \in N} a_i = c(N) \text{ and } \sum_{i \in S} a_i \leq c(S), \forall S \subseteq N \right\}.$$

Given a game (N,c), consider the following linear program:

$$\begin{aligned} \max \quad & \sum_{i \in N} a_i \\ \text{s.t.} \quad & \sum_{i \in S} a_i \leq c(S) && \forall S \subseteq N \end{aligned}$$

The core of (N,c) is non-empty if and only if at optimality the objective function of the above program is $c(N)$, that is, an optimal solution to the above program a^* satisfies $\sum_{i \in N} a_i^* = c(N)$. If the latter holds, then every optimal solution to the program above is an allocation in the core and vice versa. Consider the dual to the program above:

$$\begin{aligned} \min \quad & \sum_{S \subseteq N} \delta_S c(S) \\ \text{s.t.} \quad & \sum_{S \subseteq N, S \ni i} \delta_S = 1 && \forall i \in N \end{aligned}$$

By the strong duality theorem, the core of the game (N,c) is non-empty if and only if the optimal value of the objective function in the dual formulation is also $c(N)$. Bondareva (1963) and Shapley (1967) provide a related condition for non-emptiness of the core of a game. A map $\kappa : 2^N \setminus \{\emptyset\} \to [0,1]$ is a **balanced map** if for all $i \in N$ we have

$$\sum_{S \subseteq N, S \ni i} \kappa(S) = 1.$$

The game (N, c) is a **balanced game** if for every balanced map κ it holds that

$$\sum_{S \in 2^N \setminus \{\emptyset\}} \kappa(S) c(S) \geq c(N).$$

Bondareva (1963) and Shapley (1967) show independently that the core of a game is non-empty if and only if it is a balanced game.

Example 3 Let $N = \{1, 2, 3\}$ and consider the game (N, c). An example of a balanced map in this case is $\kappa(S) = 0.5$ if $S \subset N$ and $|S| = 2$ and $\kappa(S) = 0$ for all other $S \subseteq N$. A necessary, but not sufficient, condition for the game to have a non-empty core is to have $0.5c(\{1, 2\}) + 0.5c(\{1, 3\}) + 0.5c(\{2, 3\}) \geq c(N)$, i.e., $c(\{1, 2\}) + c(\{1, 3\}) + c(\{2, 3\}) \geq 2c(N)$. Hence whenever the latter condition is violated the core of (N, c) would be empty.

The following example shows that the core of a game can be empty.

Example 4 Consider the game (N, c) with $N = \{1, 2, 3\}$. The costs for various coalitions of players are as follows. For $S \subseteq N$ we have: $c(S) = 11$ if $|S| = 1$, $c(S) = 17$ if $|S| = 2$, $c(S) = 28$ if $|S| = 3$. Note that $c(\{1, 2\}) + c(\{1, 3\}) + c(\{2, 3\}) = 17 + 17 + 17 = 51 < 56 = 2c(N)$. By the condition established in Example 3 we conclude that $\mathscr{C}(N, c) = \emptyset$.

3.2.2 Shapley Value

The Shapley value is a single-valued solution, i.e. for every game it results in a set with a single element (a singleton). To describe the allocation rule leading to this element, to which we refer as the Shapley value as well, let $\sigma : N \to N$ be a bijection of players in N. σ can represent the order in which players join in. Denote the set of all such permutations with $\Pi(N)$. For a given permutation σ, let $\sigma(i)$ be the position of player i in the order and $P_i^\sigma = \{j \in N | \sigma(j) \leq \sigma(i)\}$ be the set of players that come before i, including i itself, in σ. We define the marginal contribution of a player in an order as the cost that the player adds to the coalition of players joining before him. Given the game $(N, c) \in \mathscr{G}$, the marginal contribution of player i in σ is

$$m_i^\sigma(N, c) = c(P_i^\sigma) - c(P_i^\sigma \setminus \{i\})$$

Let $m^\sigma(N, c) = \left(m_i^\sigma(N, c)\right)_{i \in N}$ be the vector of marginal contributions of all players in σ. The Shapley value of a game (N, c) is defined as

$$\Phi(N,c) = \frac{1}{n!} \sum_{\sigma \in \Pi(N)} m^{\sigma}(N,c).$$

The Shapley value divides the total cost of the grand coalition according to the average marginal contributions of players in all different orders that they can join the cooperative game. Note that there are exactly $n!$ of such orders. An alternative formulation of the Shapley value is

$$\Phi(N,c) = \left(\sum_{S \subseteq N, i \in S} \frac{(|S|-1)!(|N|-|S|)!}{|N|!} [c(S) - c(S \setminus \{i\})] \right)_{i \in N}.$$

Example 5 In Examples 1 and 3 the corresponding Shapley values are (8, 8, 8) and (9, 9, 9) respectively.

Shapley (1971) shows that if (N,c) is a concave game then we have $\Phi(N,c) \in \mathscr{C}(N,c)$, i.e., the Shapley value is in the core. For ease of comparison we refer to the set containing $\Phi(N,c)$ as $SH(N,c)$, that is, $SH(N,c) = \{\Phi(N,c)\}$ for every $(N,c) \in \mathscr{G}$.

3.2.3 Least-Core

The intuitive appeal of the stability concept and the possibility of having empty cores motivates alternative solutions that address the stability-related issues. An allocation a for the game $(N,c) \in \mathscr{G}$ is ϵ-stable if $\sum_{i \in S} a_i - \epsilon \leq c(S)$ for all $S \subseteq N$. The set of all ϵ-stable allocations of the game associated with a situation comprises the *ϵ-core* (Shapley and Shubik 1966).

The **least-core** of a game (Maschler et al. 1979) is the intersection of all non-empty ϵ-cores of it. Accordingly, the least-core of a game $(N,c) \in \mathscr{G}$ is defined as:

$$\mathscr{LC}(N,c) = \left\{ a \in \mathbb{R}^N \,\middle|\, \sum_{i \in N} a_i = c(N) \text{ and } \sum_{i \in S} a_i - \epsilon^{\min} \leq c(S), \forall S \subset N \right\}.$$

where

$$\epsilon^{\min}(N,c) = \min \left\{ \epsilon \in \mathbb{R} \,\middle|\, \sum_{i \in N} a_i = c(N) \text{ and } \sum_{i \in S} a_i - \epsilon \leq c(S), \forall S \subset N \right\}.$$

Considering the definition of ϵ-core, it can be observed that when the core is not empty, then the least-core is a subset of the core. Also, for every game one can always find values of ϵ such that the corresponding ϵ-core is non-empty.

3.2.4 Nucleolus

The nucleolus (Schmeidler 1969) is another well-studied solution for cooperative games. Let $(N, c) \in \mathcal{G}$ be a given cooperative cost game. Define the *imputation set* of (N, c) as

$$I(N, c) = \left\{ a \in \mathbb{R}^N \,\middle|\, \sum_{i \in N} a_i = c(N) \text{ and } a_i \le c(\{i\}), \forall i \in N \right\}.$$

Observe that if (N, c) is a subadditive game, then the imputation set of the game is non-empty. Consider an allocation $a \in \mathbb{R}^N$. Define the *coalitional unhappiness* of every coalition $S \subseteq N$ as

$$\bar{\theta}_S(a) = \sum_{i \in S} a_i - c(S).$$

Let $\theta(a)$ contain the elements $(\bar{\theta}_S(a))_{S \subset N}$ in a non-increasing order. For two vectors $\theta, \theta' \in \mathbb{R}^m$, the *lexicographical order* $\theta \le_L \theta'$ implies that either $\theta = \theta'$, or there is $1 \le t \le m$ such that $\theta_i = \theta'_i$ for $1 \le j < t$ and $\theta_t < \theta'_t$. The **nucleolus** of the game (N, c), i.e. $\eta(N, c)$, is the set of imputations whose associated vectors of unhappiness are lexicographically minimal:

$$\eta(N, c) = \left\{ a \in I(N, c) \,\middle|\, \theta(a) \le_L \theta(a'), \forall a' \in I(N, c) \right\}.$$

The nucleolus of a game has the least maximum unhappiness over all coalitions in a lexicographical manner. For every subadditive cooperative game, the nucleolus is always non-empty, unique, and is contained in the least-core (Schmeidler 1969). We remark that the allocation rule leading to the unique element of the nucleolus is often times referred to as the nucleolus as well.

3.2.5 Comparing Solutions

We present some desirable properties for solutions and compare the aforementioned ones across these properties.

A solution β on $\mathcal{G}'$ satisfies the **non-emptiness property** if for every $(N, c) \in \mathcal{G}'$ it holds that $\beta(N, c) \neq \emptyset$. The non-emptiness of a solution assures that it can suggest ways for cost sharing in all games.

As we saw in Examples 1 and 2, the core can include many allocations or no allocation at all. The **least-unstability property** is the next best thing to maintain if stability is not achievable. A solution β on $\mathcal{G}'$ satisfies the **least-unstability property** if for every $(N, c) \in \mathcal{G}'$ and every $a \in \beta(N, c)$ we have $\sum_{i \in S} a_i - \epsilon^* \le c(S)$ for every $S \subset N$ where

Table 20.1 Comparing solutions on subadditive games; NE: non-emptiness, SV: single-value, LU: least-unstability, S: stability

Allocation rule		NE	SV	LU	S
Core	$\mathscr{C}$	×	×	✓	✓
Shapley value	SH	✓	✓	×	×
Least-core	$\mathscr{LC}$	✓	×	✓	×
Nucleolus	η	✓	✓	✓	×

$$\epsilon^*(N, c) = \min\left\{\epsilon \in \mathbb{R}_+ \,\middle|\, \sum_{i \in N} a_i = c(N) \text{ and } \sum_{i \in S} a_i - \epsilon \le c(S), \forall S \subset N\right\}.$$

If the latter holds while $\epsilon^* = 0$, we say that the solution is **stable**.

Table 20.1 compares core, Shapley value, least-core and nucleolus on the class of subadditive games along these properties. As can be seen from this table, there is no perfect solution that can satisfy all these properties. The core is the only solution that guarantees stability. However, the core can be empty. The Shapley value is a single-valued solution but it may fail to be stable—or least-unstable when the core is empty. The least-core and nucleolus are both least-unstable (they are stable if the core is not empty). Furthermore, the nucleolus is a single-valued solution, that is, it always obtains a unique allocation.

3.3 *Solutions for Situations*

As mentioned earlier, a collaborative situation is a succinct description of relevant information necessary to analyze the context. Formally, we denote a collaborative situation with Γ. The set of all situations with player set N is also denoted with $\mathscr{T}$. The joint cost of collaboration among all players in N in situation $\Gamma \in \mathscr{T}$ is $c^{\Gamma}(N)$.

Let $\mathscr{T}'$ be a subset of all situations. A (situation) **solution** on $\mathscr{T}'$ is a set-valued function α that determines a set of allocations for every situation $\Gamma \in \mathscr{T}'$. In line with solutions for cooperative cost games we can consider single-valued solutions and allocation rules for situations as well.

If a situation allows for explicit calculation of joint costs for all sub-coalitions, then one can construct a cooperative cost game associated with the situation. Given such a situation Γ the associated game is denoted with (N, c^{Γ}). In this case, a situation solution α on $\mathscr{T}'$ can be defined by drawing upon a game solution β on $\mathscr{G}'$, that is, $\alpha(\Gamma) = \beta(N, c^{\Gamma})$, if for every $\Gamma \in \mathscr{T}'$ we have $(N, c^{\Gamma}) \in \mathscr{G}'$. Accordingly, one can redefine the properties defined for game solutions in the previous sub-section to situation solutions by requiring the properties to hold in the associated games. The advantage of using situation solutions over game solutions is their ability to incorporate more details from situations that allows for formalizing properties which cannot otherwise be defined over associated games. We elaborate further on this issue in next sections.

4 Cost Sharing in Logistics Network Situations

In this section, we discuss cost sharing in some of the stylized logistics network design situations. We particularly focus on possibilities for having a non-empty core in the games associated with these situations.

4.1 *Minimum Cost Spanning Tree (mcst) Games*

The minimum cost spanning tree (*mcst*) problem is a well-studied problem in operations research. An *mcst* problem consists of a set of nodes including a special node called "source". The costs of establishing links among all nodes are known. Subsequently, a minimum cost spanning tree is a set of links between the nodes that connects all nodes to the source and has the lowest total cost of establishing links among all possibilities to do so.

The cooperative version of an *mcst* problem represents the situation where each node, except the source, corresponds to a player and the players collaborate to establish a network of paths to reach the source at the lowest total cost. In the context of logistics, players on nodes can represent a set of suppliers who want to establish transportation channels to a customer. The issue of sharing the cost of an *mcst* among the players is critical in such contexts.

Formally, let $N = \{1, \ldots, n\}$ be a set of players each corresponding to a node and denote the source node with 0. The set of all nodes is denoted with $N^+ = \{0, 1, \ldots, n\}$. The set of links that can be established in the network is denoted with $L^+ = \{\{i, j\} | i, j \in N^+, i \neq j\}$. The connection cost function $w : L^+ \to \mathbb{R}_+$ gives the cost that needs to be incurred in order to establish a link between any pair of nodes in the network. For convenience we refer to $w(\{i, j\})$ as w_{ij} for every $\{i, j\} \in L^+$. A minimum cost spanning tree (*mcst*) situation can be represented with the tuple:

$$\Gamma = (N^+, w).$$

For every coalition $S \subseteq N$, let E_S be a set of links constituting a minimum cost spanning tree for players in S using the nodes in S^+ only. The **cooperative *mcst* game** associated with situation Γ is the pair (N, c^Γ) where for every $S \subseteq N$ we have $c^\Gamma(S) = \sum_{ij \in E_S} w_{ij}$.

Example 6 Figure 20.2 illustrates a network with four nodes that corresponds to an *mcst* situation Γ with three players. The connection costs along all links are presented in the figure. The *mcst* for the grand coalition is indicated with bold lines. Observe that $c^\Gamma(N) = 27$, while $c^\Gamma(\{1\}) = 6$, $c^\Gamma(\{2\}) = 17$, $c^\Gamma(\{3\}) = 18$, $c^\Gamma(\{1\}) = 6$, $c^\Gamma(\{1, 2\}) = 23$, $c^\Gamma(\{1, 3\}) = 19$, and $c^\Gamma(\{2, 3\}) = 25$. Note that $c(\{1, 3\}) - c(\{3\}) = 1$, that is, the contribution of player 1 when he joins player 3 is 1. However, we have $c(\{1, 2, 3\}) - c(\{2, 3\}) = 2$. Therefore, player 1's contribution

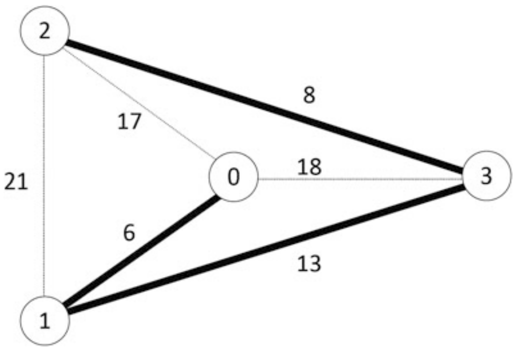


Fig. 20.2 An *mcst* example (Norde et al. 2004)

to costs increases when joining coalition of players 2 and 3. Thus, the game is not concave.

The fundamental result regarding cores of *mcst* games is as follows.

Theorem 1 *The core of an mcst game is non-empty.*

The first proof for non-emptiness of the core of an *mcst* game is given by Bird (1976). Tamir (1991) shows that the characteristic function of an *mcst* game can be represented with a mixed-integer linear program and that allocations in the core can be obtained via solutions to the dual of the integer relaxation of such program. Nevertheless, an interesting feature of *mcst* games is that one can obtain allocations in the core without solving a linear program and directly from the situation. This was shown by Bird (1976) using the Prim (1957) algorithm for solving an *mcst* problem.

The **Prim's algorithm** for finding an *mcst* over a given network starts by establishing a link between the source and the node such that the cost of this link is the lowest among all. It continues by establishing another link between a connected node and an unconnected node with the lowest connection cost. By repeating the last step the algorithm connects all nodes to the source. The solution for *mcst* situations that obtains by requiring newly connected players to pay their connection costs is called Bird's solution. The literature often referred to this solution as Bird's rule.

Bird's Solution Given situation $\Gamma = (N^+, w)$, let E_N^P be an *mcst* obtained from Prim's algorithm. For every player $i \in N$, find $j \in N^+$ such that i is directly connected to j in E_N^P on the path toward the source. Let $a_i^B(\Gamma) = w_{ij}$. Bird's solution $\alpha^B(\Gamma)$ is the set of all allocations that are obtained in this manner.

Let σ^* be an ordering of nodes as they are connected to the source using Prim's algorithm. The ordering is such that if $\sigma^*(i) < \sigma^*(j)$ for any $i, j \in N$, then i is on the path from j to the source. Then the allocation to player $i \in N$ obtained by Bird's solution with respect to σ^* is exactly his marginal contribution, that is

$$a_i^B(\Gamma) = c(P_i^{\sigma^*}) - c(P_i^{\sigma^*} \setminus \{i\}).$$

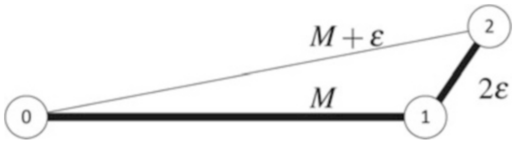


Fig. 20.3 The *mcst* situation in Example 7

Bird's solution always obtains allocations in the core, as implied by the theorem below.

Theorem 2 *For every mcst situation Γ we have $\alpha^B(\Gamma) \subseteq \mathscr{C}(N, c^\Gamma)$.*

It should be noted that Prim's algorithm does not necessarily produce unique *mcst*s thus the allocations obtained from Bird's solution need not be unique. We remark that convex combinations of allocations obtained via Bird's solution also generate allocations in the core (Curiel 1997).

Bird's solution provides a straightforward approach to obtain allocations in the core of these games. This solution directly builds upon the situation and thus one does not need to obtain the costs of all sub-coalitions or solve a linear program for finding core allocations. But is this solution always satisfactory? Consider the following example.

Example 7 Consider an *mcst* situation Γ with two players (see Fig. 20.3). Let M be a large number and ε a small number. Bird's solution obtains the unique allocation $a^B(\Gamma) = (M, 2\varepsilon)$ which is in the core. In this example, both players are at a long distance from the source although player 2 is slightly further away, i.e. $c^\Gamma(\{2\}) = c^\Gamma(\{1\}) + \varepsilon$. Still, Bird's solution requires player 1 to pay the entire cost of its connection while player 2 pays almost nothing. One would argue that this is not fair—especially when there are other allocations in the core. The core of the game is $\mathscr{C}(N, c^\Gamma) = \{(x, M+2\varepsilon-x) | \varepsilon \leq x \leq M\}$. Notice that in the allocation $(M, 2\varepsilon)$ players 1 and 2 are paying respectively the maximum and minimum amounts that they could pay in any core allocation.

The issue observed in Example 7 concerning Bird's solution is not coincidental. In fact, Bird's solution always gives extreme points in the cores of *mcst* games (Granot and Huberman 1981). Subsequently, Bird's solution always makes every group of players who are directly connected to the source collectively pay their stand-alone costs. The players that join such coalitions only pay their marginal cost of connection and thus enjoy the benefits of collaboration the most. A closer look at Bird's solution reveals some other shortcomings. We present an example.

Example 8 Consider again the situation Γ in Fig. 20.2. The highlighted *mcst* is indeed the one obtained by Prim's algorithm. We have $a^B(\Gamma) = (6, 8, 13)$ which is in the core of the game. It can be verified that in coalition $\{2, 3\}$ player 3 is allocated with 8 according to Bird's solution which is less than what that player pays in the grand coalition, i.e. 13.

As seen in Example 8, Bird's solution may result in some players being allocated with higher costs in larger coalitions. If this is the case, then such players might object to the inclusion of more players to the game despite the fact that the grand coalition can benefit from having more players (due to subaditivity). Accordingly, Bird's solution does not satisfy the population monotonicity property (Sprumont 1990).

An alternative approach for obtaining the allocations in the core of *mcst* games without recourse to the characteristic function is proposed by Norde et al. (2004) which is closely related to the Kruskal (1956) algorithm for obtaining *mcst*s. This solution is slightly less straightforward to obtain than Bird's solution. However, it has the additional advantage of producing allocations that ensure players in smaller coalitions would never be worse off by the addition of new players to the coalition (and thus satisfies the population monotonicity property).

There are several extensions of *mcst* games in the literature. We discuss two of such extensions briefly in this section.

Extension 1 Recall that in the original *mcst* game the cost of sub-coalitions are defined with regard to the *mcst*s that connect their members to the source drawing upon the nodes in their corresponding network only. That is, members of $S \subset N$ cannot use nodes involving players not in S for connecting to the source. The first extension of *mcst* games relaxes this assumption, that is, coalitions of players can construct their connection to the source using the nodes corresponding to other players. For instance, suppose the players in a coalition correspond to factories in different cities who would like to construct a network of pipelines to a supplier of water. Then the factories can indeed construct the network through the cities where other factories are located at. Given the *mcst* situation Γ, in the associated **monotone *mcst* game** (Granot and Huberman 1981), $(N, \bar{c}^{\Gamma})$, for every $S \subseteq N$ we have

$$\bar{c}^{\Gamma}(S) = \min_{S \subseteq T \subseteq N} c^{\Gamma}(T).$$

Theorem 3 *The core of a monotone mcst game is non-empty. Bird's solution gives extreme points of the cores of monotone mcst games.*

Extension 2 In the previous extension, we allowed sub-coalitions to use outsiders' nodes to construct their path to the source. Still, the grand coalition of players constituted all available nodes except the source node. Another extension of *mcst* games allows for additional nodes in the network, i.e. nodes that correspond to no players in N. Let

$$\Gamma = (V, N, w)$$

be a situation with a set of nodes V that includes the source node, the set of players $N \subset V$, and the connection cost function w defined over pairs of nodes in V. In the **Steiner Tree game** (Megiddo 1978; Sharkey 1995) associated with Γ, $(N, \bar{\bar{c}}^{\Gamma})$, the

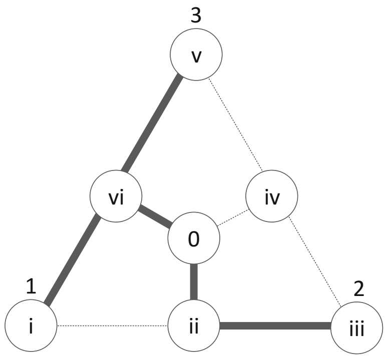


Fig. 20.4 An extended *mcst* example (Sharkey 1995)

cost of each coalition $S \subseteq N$ is the cost of *mcst* that connects players in S to the source while using any nodes in V. The following example shows that the core of these games can be empty.

Example 9 An extended *mcst* situation is depicted in Fig. 20.4. The node set includes six locations in addition to the source. There are three players in the grand coalition. The cost of connection on all links are 1. Observe that $\bar{\bar{c}}^{\Gamma}(S) = 2$ whenever $|S| = 1$, $\bar{\bar{c}}^{\Gamma}(S) = 3$ whenever $|S| = 2$, and $\bar{\bar{c}}^{\Gamma}(N) = 5$. Note that $\bar{\bar{c}}^{\Gamma}(\{1, 2\}) + \bar{\bar{c}}^{\Gamma}(\{1, 3\}) + \bar{\bar{c}}^{\Gamma}(\{2, 3\}) = 9 < 2\bar{\bar{c}}^{\Gamma}(N) = 10$. By the condition established in Example 3 we conclude that the core of the game is empty.

4.2 Facility Location Games

In facility location games, players collaborate to jointly open facilities as well as to establish connections to their locations. The basic facility location situation can be formulated as follows. Let V be a set of nodes. The player set is a subset of the nodes, that is $N \subseteq V$. A flow function $f : N \to \mathbb{R}_+$ gives the requirement of demand for each player. Let $E \subseteq \{\{i, j\}|i, j \in V\}$ be the link set representing feasible connections between the nodes. A connection cost function $w : E \to \mathbb{R}_+$ gives the cost of providing one unit of service across each link. We let $w_{ii} = 0$ for all $i \in N$. The function $t : V \to \mathbb{R}_+$ gives the total investment needed to establish facilities at different nodes (fixed costs). A facility location situation is thus

$$\Gamma = (V, N, f, E, w, t).$$

Given the facility location situation Γ, the associated cooperative cost games is the pair (N, c^{Γ}) where for every $S \subseteq N$ we have:

$$c^{\Gamma}(S) = \min \sum_{i\in S, k\in V:\{i,k\}\in E} f_i w_{ik} x_{ik} + \sum_{k\in V} t_k y_k \tag{20.1}$$

$$\text{s.t.} \sum_{k\in V:\{i,k\}\in E} x_{ik} = 1 \qquad \forall i \in S \tag{20.2}$$

$$y_k - x_{ik} \geq 0 \qquad \forall i \in S, \forall k \in V : \{i,k\} \in E \tag{20.3}$$

$$x_{ik}, y_k \in \{0,1\} \qquad \forall i \in S, \forall k \in V : \{i,k\} \in E \tag{20.4}$$

The program above minimizes the total cost of flow as well as opening facilities. The optimal solution satisfies the following constraints. First, all players in a coalition must be connected to a facility. Second, a facility should be established if there is a link to a player. Finally, integrality constraints ensure the feasibility of solution. The dual program associated with the relaxation of program (20.1)–(20.4) for N is

$$\bar{c}^{\Gamma}(N) = \max \sum_{i\in N} a_i \tag{20.5}$$

$$\text{s.t.} \quad a_i - \mu_{ik} \leq f_i w_{ik} \qquad \forall k \in V, \forall i \in N : \{i,k\} \in E \tag{20.6}$$

$$\sum_{i\in N:\{i,k\}\in E} \mu_{ik} \leq t_k \qquad \forall k \in V \tag{20.7}$$

$$\mu_{ik} \geq 0 \qquad \forall i \in N, \forall k \in V : \{i,k\} \in E \tag{20.8}$$

The solutions of the primal and the dual programs can be used to provide insights regarding non-emptiness of the core. Kolen (1983), Chardaire (1998), and Goemans and Skutella (2004) show that the dual program above is exactly the same as the program for obtaining the core of the game. Therefore, non-emptiness of the core can be guaranteed when the optimal objective function of the dual equals that of the original (un-relaxed) program. In other words, the core is non-empty if the duality gap is zero.

Theorem 4 *Let $\Gamma = (V, N, f, E, w, t)$ be a facility location situation. The core of the associated game (N, c^{Γ}) is non-empty if and only if $c(N) = \bar{c}(N)$, that is, there is no integrality gap between the primal and dual (of the relaxation) programs for the grand coalition.*

As implied by the result above, an integrality gap renders the core of a facility location game empty. The example below shows that facility location games can have empty cores.

Example 10 The network of a facility location situation is depicted in Fig. 20.5. All nodes are situated on a circle and require a unit flow. The distance between every

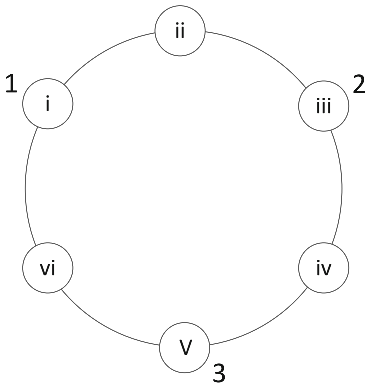


Fig. 20.5 A facility location situation on a circle (Goemans and Skutella 2004)

pair of adjacent nodes, which constitute the link set, is one and the cost of flow equals the distance. There are three players, $N = \{1, 2, 3\}$, located at nodes i, iii, and v respectively. The cost of opening a facility on nodes ii, iv, and vi is two and for the other nodes the cost is a large number. In the associated cooperative game individual players each need one facility adjacent to them thus $c^{\Gamma}(S) = 2 + 1 = 3$ for $|S| = 1$. In two-player coalitions one facility can serve both players so $c^{\Gamma}(S) = 2+1+1 = 4$ for $|S| = 2$. Finally, in the grand coalition the best option is to open two facilities thus $c^{\Gamma}(N) = 2+2+1+1+1 = 7$. Note that $c^{\Gamma}(\{1, 2\})+c^{\Gamma}(\{1, 3\})+c^{\Gamma}(\{2, 3\}) = 12 < 14 = 2c^{\Gamma}(N)$. By the condition established in Example 3 we conclude that the core of the game is empty.

There are several special situations where the zero duality gap between the primal and dual programs, and subsequently non-emptiness of the core, can be proven to always hold. For instance, suppose that the underlying graph of the situation (V, E) is a tree—i.e. there is exactly one path between any two nodes—and that the costs of connection between any pair of nodes correspond to the metric distance between those nodes on the corresponding planar graph. In this case, the original program can be re-written in the following way. For each player $i \in N$, let $0 = r_{i1} \leq r_{i2} \leq \ldots \leq r_{i|V|}$ be the ordered sequence of distances between player i's node and all other nodes. Also let $r_{iV+1} = M$ where M is a sufficiently large number. Define the variables z_{ij} such that $z_{ij} = 1$ if player i is not connected to an open facility which is situated at the distance less than or equal to r_{ij}, and $z_{ij} = 0$ otherwise. Also, define u^j_{ik} such that $u^j_{ik} = 1$ if $c_{ik} \leq r_{ij}$ and $u^k_{ij} = 0$ otherwise. Then we have (see Kolen 1983):

$$c^{\Gamma}(S) = \min \sum_{i\in S, j\in V} f_i(r_{ij+1} - r_{ij})z_{ij} + \sum_{k\in V} t_k y_k \tag{20.9}$$

$$\text{s.t.} \quad \sum_{k\in V:\{i,k\}\in E} u^j_{ik} y_k + z_{ij} \geq 1 \qquad \forall i \in S, \forall j \in V \tag{20.10}$$

$$z_{ij}, y_k \in \{0, 1\} \qquad \forall i \in S, \forall k \in V \tag{20.11}$$

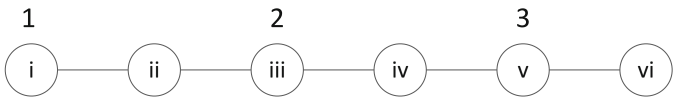


Fig. 20.6 A facility location situation on a line

Constraint (20.10) ensures that whenever there is no open facility within the range r_{ij} from i, then $z_{ij} = 1$. Kolen (1983) show that the constraint coefficient matrix in the program above has a special feature which guarantees a zero duality gap. Thus, in this class of situations the core is always non-empty.

Example 11 A facility location situation Γ' is depicted in Fig. 20.6. The only difference between this situation and the one in Example 10 is that the nodes are now situated on a line. In the cooperative game associated with this situation individual players each need one facility adjacent to them thus $c^{\Gamma'}(S) = 2 + 1 = 3$ for $|S| = 1$. In two-player coalitions $\{1, 2\}$ and $\{2, 3\}$ one facility can serve both players so $c^{\Gamma'}(S) = 2 + 1 + 1 = 4$ for $S = \{\{1, 2\}, \{1, 3\}\}$. However, for coalition $\{1, 3\}$ we have $c^{\Gamma'}(\{1, 3\}) = 2 + 2 + 1 + 1 = 6$. Finally, in the grand coalition the best option is again to open two facilities thus $c^{\Gamma'}(N) = 2 + 2 + 1 + 1 + 1 = 7$. Notice that allocation $a = (2, 2, 3)$ is in the core.

4.3 Hub Location Games

Another class of collaborative situations related to logistical problems pertains to finding the locations of logistical hubs, i.e., points of consolidation in a network, which allows for more efficient dispatching of vehicles. The basic hub location situation encompass hub-spoke structures where the transport costs in between hubs are cheaper due to the use of more efficient means of movement. Accordingly, in these collaborative situations players jointly establish hubs and connections to reduce the cost of their aggregated network.

Let V be a set of nodes in a network and let the player set N be situated amongst the nodes, i.e., $N \subseteq V$. Each player is positioned on a node and has transportation requirements from his node to other nodes. Let the requirement function $f : N \times V \to \mathbb{R}_+$ represent the latter. Define the link cost function $w : V \times V \to \mathbb{R}_+$ and hub cost function $t : V \to \mathbb{R}_+$. The cost of direct movements between nodes are sufficiently high so that it is beneficial that flows of goods between two nodes always pass through hubs. The link costs satisfy the triangular inequalities which ensure that transports between any two nodes does not need more than two hubs involved. Finally, let the coefficient $\lambda \in [0, 1]$ be the discount factor for movements between hubs. This means, if there are two hubs established at nodes i and j, then the unit cost of transportation from i to j drops from w_{ij} to λw_{ij}. A hub location situation is a tuple

$$\Gamma = (V, N, f, w, t, \lambda).$$

The players collectively decide to open hubs at some nodes in order to satisfy the flow requirements with the minimum cost. Skorin-Kapov (1998) gives a formulation of the optimal cost for the grand coalition as

$$c(N) = \min \sum_{i \in N; j,k,m \in V} f_{ij}(w_{ik} + \lambda w_{km} + w_{mj})x_{ijkm} + \sum_{k \in V} t_k y_{kk} \tag{20.12}$$

$$\text{s.t.} \sum_{k \in V} y_{ik} = 1 \qquad \forall i \in N \tag{20.13}$$

$$y_{kk} - y_{ik} \geq 0 \qquad \forall i, k \in V \tag{20.14}$$

$$\sum_{m \in V} x_{ijkm} = y_{ik} \qquad \forall i \in N, \forall j, k \in V \tag{20.15}$$

$$\sum_{k \in V} x_{ijkm} = y_{jm} \qquad \forall i \in N, \forall j, m \in V \tag{20.16}$$

$$x_{ijkm} \geq 0 \qquad \forall i \in N, \forall k, j, m \in V \tag{20.17}$$

$$y_{ik} \in \{0, 1\} \qquad \forall i, k \in V \tag{20.18}$$

The program above minimizes the total cost of movements between nodes and through hubs, plus the cost of establishing the hubs. Each player must be connected to a hub. The variable x_{ijkm} represents the fraction of flow from i's node to j that passes through hubs k and m. The zero-one variable $y = (y_{ij})_{i,j \in V}$ also indicates the connections between nodes and hubs with y_{ii} indicating the establishment of a hub at node i. The constraint (20.15) (respectively constraint (20.16)) indicates that the entire flow from a player i's node to destination j will be routed via link ik (link mj) if and only if i is allocated to hub k (j is allocated to hub m) independently of the destination (source). Let (x^*, y^*) be an optimal solution to the above problem.

There are different possibilities for defining cooperative games associated with hub locations situations based on how the cost of sub-coalitions are defined. In the basic hub location game (Skorin-Kapov 1998) the cost of sub-coalitions are calculated on the network which is optimal for the grand coalition. Let $y_k^{*S} = 1$ whenever there is $i \in S$ such that $x^*_{ijkm} > 0$ or $x^*_{ijmk} > 0$, that is, $y_k^{*S} = 1$ if a member of S uses the hub k. Then, the basic hub-location game associated with situation Γ is (N, c^Γ) where $c(N)$ is defined above and for $S \subset N$ we have

$$c(S) = \sum_{\substack{i \in S \\ j,k,m \in V}} f_{ij}(w_{ik} + \lambda w_{km} + w_{mj})x^*_{ijkm} + \sum_{k \in V} t_k y^{*S}_{kk} \tag{20.19}$$

Therefore, the cost of a sub-coalition S is the total transportation cost of movements on the optimal network plus the cost of establishing the hubs that S uses on the optimal network.

Theorem 5 *Let $\Gamma = (V, N, f, w, t, \lambda)$ be a hub location situation. The core of the basic hub location game associated with Γ is non-empty.*

4.4 Delivery Consolidation Games

With the increasing attention to reducing the negative side effects of transportation such as congestion and pollution, Urban Consolidation Centers (UCC) became an important new logistical initiative. Through a UCC, logistics providers can combine their LTL cargo and collaboratively dispatch FTL trucks to urban areas. However, the cost of joint dispatches must be shared among the users. In this section we overview a cost sharing game associated with UCCs introduced in Hezarkhani et al. (2019). The carriers (players) have deliveries that are destined for the same area. Instead of individually driving to their destinations, the players can arrive at the consolidation center and bundle their cargo into full-truck loads. The deliveries are time-sensitive and the amounts of savings that the carriers obtain are dependent on their dispatch times.

The network V consists of only two nodes: a consolidation center and a common destination and the players in N can drive the distance between the two nodes either individually or jointly. We call r_i the arrival time of delivery i to UCC and assume that deliveries have non-identical arrival times and that N is arranged by increasing order of arrival times, i.e., $r_1 < r_2 < \ldots < r_n$. Let $p_i \geq 0$ be the waiting penalty rate for player i, that is the cost that he incurs when his cargo sits in the consolidation center for a unit of time. Thus, the cost to player i if dispatched from the consolidation center at time $d_i \geq r_i$ is $p_i(d_i - r_i)$. The cost of dispatching a truck from the consolidation center to the common destination is $W \geq 0$. We assume players have small yet time sensitive cargo and the capacity of a truck is not a restriction. Accordingly, a Dispatch Consolidation (DC) situation can be defined by the tuple $\Gamma = (V, N, \boldsymbol{r}, \boldsymbol{p}, W)$.

The consolidation center decides a collection of dispatches, representing consolidated subsets of players, and their associated dispatch times. The objective of the UCC is to minimize the sum of waiting and dispatching costs for all players. One can verify that the optimal time for the dispatch of a fixed group of players in $T \subseteq N$, is the arrival time of the last player in T. Denote the first and last arriving delivery in T with $b(T)$ and $e(T)$, respectively. Since the players are ordered by their arrival times, $b(T)$ and $e(T)$ also represent respectively the smallest and largest elements in T. The cost function f for a group of players $T \subseteq N$ is

$$f_T = \sum_{i \in T} \left[p_i \left(r_{e(T)} - r_i \right) \right] + W.$$

We can construct the optimization problem as a set packing formulation and define the associated dispatch consolidation (DC) game by letting $c(S)$ to be

$$\begin{aligned} c^{\Gamma}(S) = \min & \sum_{T \subseteq S} x_T f_T \\ s.t. \quad & \sum_{T \subseteq S : S \ni i} x_T \geq 1 && \forall i \in S \\ & x_T \in \{0, 1\} && \forall T \subseteq S \end{aligned}$$

DC games are special instances of the class of set packing games (Deng et al. 1999). The general characterization of the conditions for non-emptiness of the cores of set packing games gives us the following result; The core of a DC game is non-empty if and only if the integer relaxation of the program above for N does not affect optimality.

Using the results of Barany et al. (1986) regarding zero duality gap of set packing problems on trees via their sub-trees, Hezarkhani et al. (2019) show that integer relaxation of the program above in DC games does not affect optimality. Therefore, the core of any DC games is non-empty.

The extension of DC games to incorporate restrictive capacities of the trucks is also considered by Hezarkhani et al. (2019). With restrictive capacities, DC games might have an empty core. In this case, Hezarkhani et al. (2019) introduce the notion of component-wise core as an alternative notion of stability and prove that DC games with restrictive capacities have non-empty component-wise cores.

5 Cost Sharing in Cooperative Truck-Load Delivery Situations

The logistical situations studied in the previous section were all concerned with establishing the physical network which is comprised of links, facilities, and hubs. The corresponding decisions are at the strategic level and as such necessitate a long term cooperation time line. However, there are other opportunities for cooperative logistics which deal with day-to-day activities of participating players and target operational decisions. In these *service logistics* situations, the nature of cost sharing problems can be different. In this section, we discuss the Cooperative Truck-Load Delivery (CTLD) situations, introduced by Hezarkhani et al. (2016), that arise in service network design and explain how an appropriate allocation rule for these situations can be devised.

CTLD situations are comprised of a number of logistics providers and their individual resources—e.g. depots, trucks, drivers, equipment, etc. Players have delivery requirements. A delivery requirement is simplified as an order for picking up cargo at some location and transporting it to another location. In practice, delivery requirements can be more complex and involve time windows, special equipment and personnel, and other constraints. The delivery requirements must be fulfilled by vehicles in feasible trips. In this context, the players seek to collaboratively design their service network at the tactical as well as operational levels.

Formally, let V be a set of nodes corresponding to spatial locations, and $w : V \times V \to \mathbb{R}^+$ be a distance function which satisfies the triangular inequalities. We assume hereafter that cost and distance are equivalent. A set of delivery requirements $\{d^1, \ldots, d^m\}$ is given. A *delivery requirement* d^k corresponds to an arc (i^k, j^k), consisting of the corresponding pickup location $i^k \in V$ and delivery location $j^k \in V, i^k \neq j^k$. The *fulfillment* of the delivery requirement d^k corresponds to a single traverse of the arc (i^k, j^k) for requirement k. A non-empty set of depots $\{o^1, \ldots, o^h\} \subseteq V$ is available. The depots station vehicles that fulfill the delivery requirements. Delivery requirements must be fulfilled in trips. A trip is a sequence of deliveries that start and ends at a particular depot. Thus a trip l can be defined as a tuple (o^l, D^l, σ^l) where o^l is the origin/destination, D^l is a subset of deliveries that are fulfilled in l, and σ^l is an ordering of deliveries in D^l which represents the sequence of fulfillments in trip l. Let $\mathscr{L}$ be the set of all such trips. Let $L \subseteq \mathscr{L}$ be the feasible trip set. The feasibility of a trip can depend on the number and type of deliveries it fulfills, specific depots and equipment that must be employed, and other details.

The cost of a feasible trip l, w^l, is the sum of costs of the arcs traversed in trip l. The *full kilometers cost* of a trip is independent of both the choice of the trip's depot and the sequence of fulfillments:

$$w_F^l = \sum_{k:d^k \in D^l} w\left(i^k, j^k\right).$$

The second part of a trip's cost, i.e. *empty kilometers cost*, is the cost associated with the distance travelled from/to the depot and among different fulfillments:

$$w_E^l = w\left(o^l, i^{\sigma_1^l}\right) + \sum_{k=1}^{|D^l|-1} w\left(j^{\sigma_k^l}, i^{\sigma_{k+1}^l}\right) + w\left(j^{\sigma_{|D^l|}^l}, o^l\right).$$

where the shorthand notation σ_k^l represents the index of the delivery requirement that is fulfilled after all the $k-1$ deliveries preceding it in σ^l are fulfilled. By $|D^l|$ we denote the number of deliveries in D^l. The *cost of trip l* is defined by $w^l = w_F^l + w_E^l$.

A fulfillment plan P from O to D is a collection of feasible trips in $L(O, D)$ that fulfills all deliveries in D exactly once. The deliveries fulfilled in the trips of the plan P partition the corresponding set of delivery requirements, i.e. $\bigcup_{l \in P} D^l = D$ and $D^l \cap D^k = \emptyset$ for all $k, l \in P$ with $l \neq k$. The cost of the fulfillment plan P is the total cost of its trips, i.e. $w(P) = \sum_{l \in P} w^l$. Accordingly, $w(P)$ is decomposable into full and empty movements:

$$w(P) = w_F(P) + w_E(P),$$

where $w_F(P) = \sum_{l \in P} w_F^l$ and $w_E(P) = \sum_{l \in P} w_E^l$ are the total costs of full and empty kilometers of P respectively. Let $\mathscr{P}(O, D)$ be the set of feasible plans from O to D. The cost of optimal plan from O to D is

$$w^*(O, D) = \min_{P \in \mathscr{P}(O,D)} w(P).$$

Consider a non-empty set $N = \{1, \ldots, n\}$ of players. Each player $i \in N$ possesses a set of delivery requirements $D_i = \{d_i^1, \ldots, d_i^{m_i}\}$ and a non-empty set of depots $O_i = \{o_i^1, \ldots, o_i^{h_i}\}$ such that $\cup_{i \in N} D_i = \{d^1, \ldots, d^m\}$ and $\cup_{i \in N} O_i = \{o^1, \ldots, o^h\}$. Let $O_S = \cup_{i \in S} O_i$ and $D_S = \cup_{i \in S} D_i$ denote the combined set of depots and delivery requirements of players in coalition $S \subseteq N$. The set $L(O_S, D_S)$ contains all feasible trips that coalition $S \subseteq N$ can use to fulfill its combined delivery requirements. Combining all this, a CTLD situation is a tuple:

$$\Gamma = (N, V, w, (D_i)_{i \in N}, (O_i)_{i \in N}, L).$$

Let $\mathscr{T}'$ be the set of all CTLD situations. By joint planning of fulfillments, a coalition in a CTLD situation could reduce the cost of its empty kilometers. The cost saving generated by a coalition can be due to utilization of a larger pool of depots for constructing trips or combining fulfillments together more efficiently in trips, or both. It can be verified that shrinking the set of delivery requirements cannot increase the minimum cost of delivery, and augmenting the set of depots cannot increase the minimum cost of delivery. Also, there is a subadditive effect with regard to the minimum costs of fulfillment that results from aggregated planning of delivery requirements (see Hezarkhani et al. 2016).

We refer to the cost games associated with CTLD situations as the CTLD games. The characteristic function in CTLD game (N, c^Γ) associated with situation Γ assigns to coalition $S \subseteq N$ the cost

$$c^\Gamma(S) = w^*(O_S, D_S).$$

Although there are special CTLD situations where the core is always non-empty (see Özener and Ergun 2008 and Hezarkhani et al. 2014), in general, CTLD games can have empty cores, as shown in the example below.

Example 12 Consider the CTLD situation Γ depicted in Fig. 20.7. There are three players $N = \{1, 2, 3\}$ each having a depot and a delivery requirement. The distance between the pickup and delivery locations for all delivery requirements is two and the distance from the depots to any pickup/delivery point is one. The set of feasible

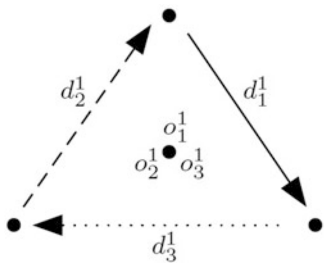


Fig. 20.7 A CTLD situation where players have different competitive positions

trips includes all trips which fulfill no more than two delivery requirements, i.e. $L = \{l \in \mathscr{L} | |D^l| \leq 2\}$ (only two deliveries can be fulfilled sequentially during a day). For $S \subseteq N$ we have $c^\Gamma(S) = 4$ if $|S| = 1$, $c^\Gamma(S) = 6$ if $|S| = 2$, and $c^\Gamma(N) = 10$. Applying the condition in Example 3, we obtain that the core of this game is empty.

5.1 *Desirable Properties for CTLD Solutions*

In order to find solutions for CTLD situations, i.e., solutions defined over the set of all CTLD situations $\mathscr{T}'$, we define a set of properties that could be considered as desirable in these situations.

The notion of stability is a critical concept in many cooperative situations, including CTLD situations. Given the possibility of having empty cores, we seek for the best possible outcomes in terms of instability of allocations. Thus the first desirable property for CTLD solutions is that of least-unstability. A solution α on $\mathscr{T}'$ satisfies the least-unstability property if for every $\Gamma \in \mathscr{T}'$ and every $a \in \alpha(\Gamma)$ we have $\sum_{i \in S} a_i - \epsilon^* \leq c^\Gamma(S)$ for every $S \subset N$ where ϵ^* is defined in the same way as in Sect. 3.2.5 for the associated game (N, c^Γ).

The highly competitive nature of logistics markets as well as the limited number of potential participants necessitate solutions that are capable of incorporating the notion of competitiveness among the logistics providers. The two properties discussed in the remainder of this section are specific to CTLD situations and address issues concerning the competitive positions of the players and the scope beyond which the network of deliveries of a player should be ignored by the solution. We start by introducing two special classes of delivery requirements in CTLD situations. Let $\Gamma \in \mathscr{T}'$ be a CTLD situation with player set N. $D \subseteq D_i$ is a separable delivery set (SDS) of player i if

$$w^*(O_i, D) + w^*(O_N, D_N \setminus D) = w^*(O_N, D_N). \tag{20.20}$$

Let $SDS_i(\Gamma)$ be the set of separable delivery sets of i. The stand-alone cost of fulfilling a separable delivery set of a player is additive to the cost of fulfilling

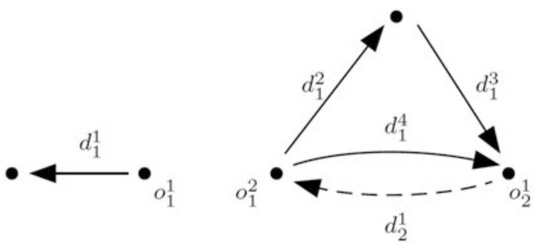


Fig. 20.8 Separable and irrelevant deliveries

the remaining deliveries in the grand coalition. Therefore, a player can individually fulfill a separable delivery set of itself without disrupting the optimality of delivery plans in the grand coalition. Let $\Gamma \in \mathscr{T}'$ be a CTLD situation with player set N. $D \subseteq D_i$ is an **irrelevant delivery set** (IDS) of i if for all $D' \subseteq D$, all $S \subseteq N$ with $i \in S$, and all $D'' \subseteq D_S \setminus D$ it holds that

$$w^*(O_i, D') + w^*(O_S, D'') = w^*(O_S, D' \cup D''). \tag{20.21}$$

Let $IDS_i(\Gamma)$ be the set of irrelevant delivery sets of i. The cost of fulfilling any subsets of irrelevant deliveries of a player is additive to any subset of the set of remaining deliveries in any coalition that includes that player, so the player can fulfill such deliveries separately in any possible combination with other deliveries. The following example elaborates on the notion of separable and irrelevant deliveries.

Example 13 Figure 20.8 depicts a CTLD situation Γ with two players $N = \{1, 2\}$. It is easy to see that player 1 can individually fulfill the delivery requirement $\{d_1^1\}$. Also, player 1 can take out either $\{d_1^2, d_1^3\}$ or $\{d_1^4\}$ (but not both sets!) from the grand coalition's delivery requirements and fulfill them separately such that the total cost of fulfillment does not increase. Thus, we have $SDS_1(\Gamma) = \big\{\{d_1^1\}, \{d_1^2, d_1^3\}, \{d_1^4\}, \{d_1^1, d_1^2, d_1^3\}, \{d_1^1, d_1^4\}\big\}$ and $IDS_1 = \big\{\{d_1^1\}\big\}$.

Given $D_i' \subseteq D_i$, let $\Gamma \setminus D_i'$ be a CTLD situation that coincides with Γ except for the delivery set of i which is replaced by $D_i \setminus D_i'$. Define the *independence of irrelevant deliveries* property as the insensitivity of a solution to the exclusion of irrelevant deliveries of the players. A solution for CTLD situations α satisfies the **independence of irrelevant deliveries property** if for every $\Gamma \in \mathscr{T}'$, every $a \in \alpha(\Gamma)$, and every $a' \in \alpha(\Gamma \setminus D)$ it holds for every $i \in N$ and every $D \in IDS_i(\Gamma)$ that $a_i = a_i' + w^*(O_i, D)$ and $a_j = a_j'$ for every $j \in N \setminus \{i\}$.

The last property addresses the competitive aspect of solutions in CTLD situations.

We define the *average cost of fulfillment* from O to $D \neq \emptyset$ as

$$z(O, D) = \frac{w^*(O, D)}{w_F(D)} \tag{20.22}$$

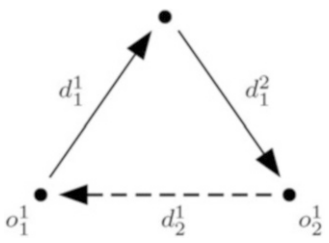


Fig. 20.9 A CTLD situation where players have different competitive positions

where $w_F(D)$ is the cost of full kilometers needed to be traversed to fulfill D. The average cost of fulfillment $z(O, D)$ represents the average distance (cost) that need to be traveled (incurred) in D in order to fulfill a unit distance of delivery requirement.

The average cost of fulfillment provides a basis for calculating unit delivery prices in logistics markets. However, it can also be utilized as a measure of comparison among the players. This idea is motivated by the observation that a lower average cost of fulfillment of a logistics player compared to that of another logistics player allows the former to charge a lower unit price for its delivery services while remaining profitable. Therefore, if for two players i and j it holds that $z(O_i, D_i) < z(O_j, D_j)$, it can be stated that prior to cooperation, i is in a better competitive position than j. The definition of average cost of fulfillment can be naturally extended to incorporate the savings allocated to the players after the cooperation. Given an allocation a and player $i \in N$, $D_i \neq \emptyset$, define the average cost of fulfillment of a player i under a as

$$z_i^a(O_i, D_i) = \frac{a_i}{w_F(D_i)} \tag{20.23}$$

We are now ready to present a competitiveness property defined over a restricted set of CTLD situations. Let $\hat{\mathscr{T}}'$ be the set of all CTLD situations $\Gamma \in \mathscr{T}'$ with player set N such that $SDS_i(\Gamma) = \{\emptyset\}$ for all $i \in N$. A CTLD solution satisfies the **restricted competitiveness property** if for every situation Γ with player set $N = \{1, 2\}$ and any $a \in \alpha(\Gamma)$ it holds that

$$z_1^a(O_1, D_1)z(O_2, D_2) = z_2^a(O_2, D_2)z(O_1, D_1). \tag{20.24}$$

Example 14 Figure 20.9 represents a CTLD situation with two players $N = \{1, 2\}$. Assuming that the distance between any two locations is 1, we get $z(O_1, D_1) = 1.5$ and $z(O_2, D_2) = 2$. The cooperation in this case results in $c^\Gamma(N) = 3$, i.e., 2 units of saving compared to individual fulfillments $c^\Gamma(\{1\}) = 3$ and $c^\Gamma(\{2\}) = 2$. Observe that the allocation $a = (1.8, 1.2)$ preserves the competitive positions of players 1 and 2 before and after the cooperation, resulting in $z_1^a(O_1, D_1) = 0.9$ and $z_2^a(O_2, D_2) = 1.2$.

5.2 A Solution for CTLD Situations

The proposed CTLD solution is constructed in two steps. In the first step, we introduce a proportional allocation, a^P, which incorporates the notions of competitiveness and scope defined in the previous section. In the second step, we use the latter proportional allocation to construct a least-unstable solution, α^P, for CTLD situations.

Let $\Gamma \in \mathscr{T}'$ be a CTLD situation with player set N. $D \subseteq D_i$ is a **minimal essential delivery set** (MEDS) of player i if

$$w^*(O_i, D_i \setminus D) + w^*(O_N, D_{N\setminus i} \cup D) = w^*(O_N, D_N). \tag{20.25}$$

and for every $D' \subset D$, $D \neq \emptyset$:

$$w^*(O_i, D_i \setminus D') + w^*(O_N, D_{N\setminus i} \cup D') > w^*(O_N, D_N) \tag{20.26}$$

and $w^*(O_i, D) \leq w^*(O_i, D')$ for any $D^{'}$ that satisfy the above two conditions.

Fix Γ, let $D_i^m \in MEDS_i(\Gamma)$, and define

$$a_i^p(\Gamma)=w^*(O_i, D_i)-\frac{w^*(O_i, D_i^m)}{\sum_{j\in N} w^*(O_j, D_j^m)}\left(\sum_{j\in N} w^*(O_j, D_j)-w^*(O_N, D_N)\right) \tag{20.27}$$

The allocation a^P obtains a unique efficient allocation that divides the savings obtained in the grand coalition of CTLD situation Γ among players with non-empty essential delivery sets proportional to the stand-alone cost of their minimal essential deliveries. The above formulation assumes that the essential delivery set of all players are non-empty. See Hezarkhani et al. (2016) for the treatment of the other case. The allocation a^P completely preserves the competitive positions of the players with regard to their minimal essential delivery sets. This means that for every pair of players $i, j \in N$ with non-empty essential delivery sets we have

$$\frac{z_i^{a^P(\Gamma)}(O_i, D_i^m)}{z_i(O_i, D_i^m)} = \frac{z_j^{a^P(\Gamma)}(O_j, D_j^m)}{z_j(O_j, D_j^m)}.$$

The allocation a^P, however, does not necessarily obtain a least-unstable allocation. In order to achieve this, we present our CTLD solution α^P:

$$\alpha^P(\Gamma) = \arg\min_{a \in \mathbb{R}^N} \sum_{i \in N} (a_i^P(\Gamma) - a_i)^2 \tag{20.28}$$

$$s.t. \quad \sum_{i \in S} a_i - \epsilon^* \leq w^*(O_S, D_S) \qquad \forall S \subset N \tag{20.29}$$

$$\sum_{i \in N} a_i = w^*(O_N, D_N) \tag{20.30}$$

where ϵ^* is defined in Sect. 3.2.3. Given the situation Γ, $\alpha^P(\Gamma)$ gives the set of all ϵ^*-stable allocations that have the shortest distance from the proportional allocation $a^P(\Gamma)$. The following result is proven by Hezarkhani et al. (2016).

Theorem 6 α^P *satisfies the nonemptiness, uniqueness, least-unstability, independence of irrelevant deliveries, and restricted competitiveness properties for all CTLD situations.*

6 Bibliographical Notes

We split this section in two parts: literature on collaborations and literature on the relevant game theorical background.

6.1 Collaborations

Quak and Tavasszy (2011) report that among more than 100 initiatives in urban logistics collaborations, more than half of them fail during implementation. There are several underlying reasons for this (Vanovermeire et al. 2014), e.g. collaboration among carriers is often hampered by their competitive positions and by the risks of divulging information and losing customers. Shippers, on the other hand, may hesitate to collaborate as they might not have a clear understanding of collaborative mechanisms employed and whether or not they receive a fair share out of collaborative operations. In a survey based on a large number of logistics service providers (LSPs) in Belgium, Cruijssen et al. (2007) observe that despite the obvious benefits of cooperation, designing a fair cost sharing scheme is a major impediment for collaboration among LSPs. For more information on the fill rates of vehicles refer to (Eurostat 2018).

Good examples of such cost sharing reviews already exist in the literature (see e.g. Deng and Fang 2008; Marinakis et al. 2008). Although the literature often associates the definition of the core to Gillies (1959), it was Shapley who first defined the core in its current form (Zhao 2018).

In their review paper, Gansterer and Hartl (2018) distinguish between centralized versus decentralized planning in cooperation. Having perfect information with regards to all requests (central planning) leads to profit sharing approaches, usually based on game-theoretical principles. In decentralized planning, imperfect information to no request information is assumed. Most research is circulating around horizontal collaboration and cost sharing concepts. Early research on horizontal collaboration considering independent freight carriers is discussed in Kopfer and Pankratz (1998), researching a groupage system, and coining the term Collaborative Transport Planning (CTP). One fair allocation of the savings can be done via the Shapley value introduced by Shapley (1953) that uniquely distributes the savings among the participants.

Cruijssen, and Salomon (2004) showed that order sharing potentially leads to remarkable savings up to 15%. In a follow up paper, Cruijssen et al. (2007) investigated the opportunities and obstacles carriers face in horizontal collaborations. Topics such as a fair allocation of the savings, carrier differentiation, trust and the extent of cooperation are important drivers for success or failure (see also Pomponi et al. (2015)).

Krajewska and Kopfer (2006) introduced an exchange mechanism build around three phases: preprocessing, exchange mechanism, and profit sharing. These cooperation mechanisms are applied to the pickup and delivery problem with time windows (PDPTW) in Krajewska and Kopfer (2006) and Krajewska et al. (2008). This problem is extended with transshipment points for the collaborating carriers by Vornhusen et al. (2014). Wang et al. (2017) investigated the capacitated VRP. Cuervo et al. (2016) did simulations on the effects of partner characteristics. Larger order portfolios lead to larger gains through collaborative coalitions.

Berger and Bierwirth (2010) focused on the exchange mechanism in cooperation for the traveling salesman problem with pickup and delivery. The auctioning of request bundles is an NP–hard combinatorial auctioning problem (CAP). Wang and Kopfer (2014) showed potential cost savings of on average 18.2% up to 64.8%. Wang et al. (2017) applied a route—based bidding mechanism to the PDPTW. Li et al. (2015) formulated a single request exchange approach. Jacob and Buer (2018) investigated the effects of non-truthful bidding and showed that is individually rational but not collectively rational, resulting into a variant of the famous prisoner's dilemma.

Gansterer and Hartl (2016) investigated several request evaluation strategies building on Berger and Bierwirth (2010). Using heuristics, they solve larger instances for the TSP with precedence constraints. Gansterer and Hartl (2018) showed that attractive subsets of predefined bundles can be effectively identified, reducing the computation complexity. More recently, Gansterer et al. (2020) showed the advantage to bundle requests rather than individual requests. Karels et al. (2020) investigate an auction mechanism to facilitate collaboration amongst carriers while maintaining autonomy for the individual carriers, based on a traditional vehicle routing problem.

6.2 Game Theoretical Concepts

Lloyd Shapley introduced two of the most well-known game theoretic solutions, i.e., the core (Shapley 1955), and the Shapley value (Shapley 1953). Although the literature often associates the definition of the core to Gillies (1959), it was Shapley who first defined the core in its current form (Zhao 2018). The search for the core of cooperative games in network situations has motivated a large body of literature (e.g. Borm et al. 2001; Curiel 2008), and implementation of the Shapley value has been suggested by a host of research in collaborative logistics (e.g. Krajewska et al. 2008).

The Nucleolus was first developed by Schmeidler (1969). The unhappiness function used in the definition of the nucleolus can be defined in other ways as well. See Tijs and Driessen (1986) for a review of alternative definitions. Alternative approaches for proving non-emptiness of the cores of *mcst* games have been proposed in the literature, (e.g., Bird 1976, Granot 1986, Granot and Huberman 1981, and Tamir 1991). Although the basic *mcst* situation presented here deals with undirected graphs, similar results also hold for the more general situations with directed graphs. The proof in Tamir (1991) is for directed situations. The proof of Theorem 2 is given in Granot and Huberman (1981). Other solutions for *mcst* situations have been discussed, among others, by Aarts and Driessen (1993) and Bogomolnaia and Moulin (2010) via the concept of the irreducible core, which gives subsets of core allocations. It is worth mentioning that the Shapley value in *mcst* games is also studied in Kar (2002) who provides an axiomatization of this allocation rule for the class of *mcst* games. Interested readers can refer to Granot and Huberman (1981) for the proof of Theorem 3.

Further extensions of the facility location game are studied in the literature, see for instance Mallozzi (2011) and Xu and Du (2006).

The proof of Theorem 5 is given by Skorin-Kapov (1998) where he also considers other variations of hub location games. Further extensions of hub locations games are discussed in Matsubayashi et al. (2005) and Skorin-Kapov (2001).

6.3 Other Classes of Stylized Situations Related to Cooperative Network Design Problems

There are several other classes of stylized situations related to cooperative network design problems for which the cost sharing problems have been studied in the literature. In traveling salesman situations, the goal is to construct cycles with minimum total cost from a source through a set of given nodes representing the players. Accordingly, in traveling salesman games players in a coalition cooperate to establish such cycles among themselves and the source. The main difference between the traveling salesman and *mcst* situations is that of cycles versus trees in constructing solutions respectively. It has been proven that all traveling salesman games with five or less players have non-empty cores (Potters et al. 1992; Tamir 1989; Kuipers 1993). However, for games with six players and above the core can be empty (Tamir 1989; Faigle et al. 1998). In vehicle routing situations, the players would have demands with specific sizes that must be satisfied with vehicles with limited capacity via tours from an origin node. As the class of vehicle routing situations contains the traveling salesman situations as a special instance, the negative results regarding the emptiness of the cores of associated games holds as well. However, Göthe-Lundgren et al. (1996) casts the vehicle routing situations as set partitioning problems and show that non-emptiness of the core can be guaranteed whenever the duality gap for the corresponding linear relaxation is zero. Interested readers are also referred to Chinese-Postman Games (Hamers et al. 1999; Platz and Hamers 2013), Delivery Scheduling Games (Hezarkhani 2016), and Delivery Consolidation Games (Hezarkhani et al. 2019).

7 Conclusions and Perspectives

In this chapter, we looked into the role of cooperation within Transport and Logistics networks. The success concepts like the Physical Internet, urban hubs, or crowd-sourcing, depends heavily on managing the pain-and-gain sharing mechanisms. Clearly, having multiple stakeholders involved in the transportation processes, leads to important cooperation issues. The drivers for cooperation are mainly related to resource utilization optimizations, leading to e.g. less empty mileage or increase truckloads. Game theory helps us to model, understand and optimize these collaborations from a cost sharing perspective. Cooperative game theory provides a set of tools and techniques to address such problems.

Most discussed Transport and Logistics applications (including the network design models) involve very complex situations, as their underlying models are not easy to solve to optimality in a tractable way. This poses a serious problem in adoption of available solutions originating from cooperative game theory. Hence, finding appropriate cost shares is challenging for the Operations Research-based network design models, and we have to revert to the more basic and stylized network design models.

In these highly stylized situations, it might be possible to directly use well-known solutions. Accordingly, one might be able to devise solutions that obtain appropriate cost shares, e.g. allocations in the core, directly from the underlying optimization problems. Specifically, in a collaborative network design situation, there might be a straightforward connection between the optimization program and the appropriate cost shares.

However, classical approaches in cooperative game theory alone are not able to satisfactorily solve cost-sharing problems in the more complex network design situations. On the one hand, the core of games associated with these simulations might be empty—even in relatively simple situations. On the other hand, inherent difficulties in solving the underlying optimization problem can render these solutions too complex (or time consuming).

Despite the theoretical appeal of basic problems discussed in the previous sections, collaborative situations in practice are often complicated by many factors and constraints. Solutions might need to satisfy properties that are specific to a collaborative situations and cannot be captured by standard game-theoretic solutions.

All this motivates research on situation-specific solutions for more advanced network design models. In developing reasonable solutions for these situations, one can formulate practical requirements in terms of desirable properties. We argue that on exactly on this interface of cooperative game theory and network design models, investigating the desirable properties of these solutions and their formal definition ex-ante is needed to obtain more meaningful results, rather than using standard game theoretical solutions.

References

Aarts, H., & Driessen, T. (1993). The irreducible core of a minimum cost spanning tree game. *Mathematical Methods of Operations Research, 38*(2), 163–174.

Barany, I., Edmonds, J., & Wolsey, L. A. (1986). Packing and covering a tree by subtrees. *Combinatorica, 6*(3), 221–233.

Berger, S., & Bierwirth, C. (2010). Solutions to the request reassignment problem in collaborative carrier networks. *Transportation Research Part E: Logistics and Transportation Review, 46*, 627–638.

Bird, C. G. (1976). On cost allocation for a spanning tree: a game theoretic approach. *Networks, 6*(4), 335–350.

Bogomolnaia, A., & Moulin, H. (2010). Sharing a minimal cost spanning tree: Beyond the folk solution. *Games and Economic Behavior, 69*(2), 238–248.

Bondareva, O. N. (1963). Some applications of linear programming methods to the theory of cooperative games. *Problemy kibernetiki, 10*, 119–139 (In Russian).

Borm, P., Hamers, H., & Hendrickx, R. (2001). Operations research games: A survey. Discussion Paper 45, Tilburg University, Center for Economic Research.

Chardaire, P. (1998). *Facility location optimization and cooperative games.* (PhD thesis, University of East Anglia, 1998).

Cruijssen, F., & Salomon, M. (2004). Empirical study: Order sharing between transportation companies may result in cost reductions between 5 to 15 percent. *CentER Discussion Paper*, (2004-80).

Cruijssen, F., Cools, M., & Dullaert, W. (2007). Horizontal cooperation in logistics: Opportunities and impediments. *Transportation Research Part E, 43*(2), 129–142.

Cuervo, D. P., Vanovermeire, C., & Sörensen, K. (2016). Determining collaborative profits in coalitions formed by two partners with varying characteristics. *Transportation Research Part C: Emerging Technologies, 70*, 171–184.

Curiel, I. (1997). Minimum cost spanning tree games. In *Cooperative game theory and applications* (pp. 129–148). Berlin: Springer.

Curiel, I. (2008). Cooperative combinatorial games. In *Pareto optimality, game theory and equilibria* (pp. 131–157). Berlin: Springer.

Deng, X., & Fang, Q. (2008). Algorithmic cooperative game theory. *Pareto Optimality, Game Theory And Equilibria, 17*, 159–185.

Deng, X., Ibaraki, T., & Nagamochi, H. (1999). Algorithmic aspects of the core of combinatorial optimization games. *Mathematics of Operations Research, 24*(3), 751–766.

Eurostat. (2018). Annual road freight transport vehicle movements, loaded and empty, by reporting country. http://epp.eurostat.ec.europa.eu/portal/page/portal/transport/data/database

Faigle, U., Fekete, S. P., Hochstättler, W., & Kern, W. (1998). On approximately fair cost allocation in Euclidean TSP games. *OR Spectrum, 20*(1), 29–37.

Gansterer, M., & Hartl, R. F. (2016). Request evaluation strategies for carriers in auction-based collaborations. *OR Spectrum, 38*, 3–23.

Gansterer, M., & Hartl, R. (2018). Centralized bundle generation in auction-based collaborative transportation *OR Spectrum, 40*, 613–635.

Gansterer, M., Hartl, R. F., & Sörensen, K. (2020). Pushing frontiers in auction-based transport collaborations. *Omega, 94*, 102042.

Gillies, D. B. (1959). Solutions to general non-zero-sum games. *Contributions to the Theory of Games, 4*, 47–85.

Goemans, M. X., & Skutella, M. (2004). Cooperative facility location games. *Journal of Algorithms, 50*(2), 194–214.

Göthe-Lundgren, M., Jörnsten, K., Värbrand, P. (1996). On the nucleolus of the basic vehicle routing game. *Mathematical Programming, 72*(1), 83–100.

Granot, D. (1986). A generalized linear production model: A unifying model. *Mathematical Programming, 34*(2), 212–222.

Granot, D., & Huberman, G. (1981). Minimum cost spanning tree games. *Mathematical Programming, 21*(1), 1–18.

Hamers, H., Borm, P., van de Leensel, R., & Tijs, S. (1999). Cost allocation in the Chinese postman problem. *European Journal of Operational Research, 118*(1), 153–163.

Hezarkhani, B. (2016). Pairwise mergers in bipartite matching games with an application in collaborative logistics. *Operations Research Letters, 44*(6), 818–822.

Hezarkhani, B., Slikker, M., & Van Woensel, T. (2014). On characterization of the core of lane covering games via dual solutions. *Operations Research Letters, 42*(8), 505–508.

Hezarkhani, B., Slikker, M., & Van Woensel, T. (2016). A competitive solution for cooperative truckload delivery. *OR Spectrum, 38*(1), 51–80.

Hezarkhani, B., Slikker, M., & Van Woensel, T. (2019). Gain-sharing in urban consolidation centers. *European Journal of Operational Research, 279*(2), 380–392.

Jacob, J., & Buer, T. (2018). Impact of non-truthful bidding on transport coalition profits. *Operations Research Proceedings 2016, 84*, 203–208.

Kar, A. (2002). Axiomatization of the Shapley value on minimum cost spanning tree games. *Games and Economic Behavior, 38*(2), 265–277.

Karels, V. C., Veelenturf, L. P., & Van Woensel, T. (2020). An auction for collaborative vehicle routing: Models and algorithms. *EURO Journal on Transportation and Logistics, 9*(2), 100009.

Kolen, A. (1983). Solving covering problems and the uncapacitated plant location problem on trees. *European Journal of Operational Research, 12*(3), 266–278.

Kopfer, H., & Pankratz, G. (1998). Das Groupage-Problem kooperierender Verkehrsträger. In *Operations Research Proceedings* (pp. 453–462). Berlin: Springer.

Krajewska, M.A., & Kopfer, H. (2006). Collaborating freight forwarding enterprises. *OR Spectrum, 28*(3), 301–317 (2006)

Krajewska, M. A., Kopfer, H., Laporte, G., Ropke, S., & Zaccour, G. (2008). Horizontal cooperation among freight carriers: request allocation and profit sharing. *Journal of Operational Research Society, 59*(11), 1483–1491.

Kruskal, J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society, 7*(1), 48–50.

Kuipers, J. (1993). A note on the 5-person traveling salesman game. *Mathematical Methods of Operations Research, 38*(2), 131–139.

Li, J., Rong, G., & Feng, Y. (2015). Request selection and exchange approach for carrier collaboration based on auction of a single request. *Transportation Research Part E: Logistics and Transportation Review, 84*, 23–39.

Mallozzi, L. (2011). Cooperative games in facility location situations with regional fixed costs. *Optimization Letters, 5*(1), 173–181.

Marinakis, Y., Migdalas, A., & Pardalos, P. M. (2008). Cost allocation in combinatorial optimization games. In *Pareto optimality, game theory and equilibria* (pp. 217–247). Berlin: Springer

Maschler, M., Peleg, B., & Shapley, L. S. (1979). Geometric properties of the kernel, nucleolus, and related solution concepts. *Mathematics of Operations Research, 4*(4), 303–338.

Matsubayashi, N., Umezawa, M., Masuda, Y., & Nishino, H. (2005). A cost allocation problem arising in hub–spoke network systems. *European Journal of Operational Research, 160*(3), 821–838.

Megiddo, N. (1978). Cost allocation for Steiner trees. *Networks, 8*(1), 1–6.

Norde, H., Moretti, S., & Tijs, S. (2004). Minimum cost spanning tree games and population monotonic allocation schemes. *European Journal of Operational Research, 154*(1), 84–97. ISSN 0377-2217.

Özener, O. Ö., Ergun, Ö. (2008). Allocating costs in a collaborative transportation procurement network. *Transportation Science, 42*(2), 146–165.

Platz, T., & Hamers, H. (2013). On games arising from multi-depot Chinese postman problems. Technical report, Tilburg University, Center for Economic Research.

Pomponi, F., Fratocchi, L., Tafuri, S. R. (2015). Trust development and horizontal collaboration in logistics: A theory based evolutionary framework. *Supply Chain Management: An International Journal*, 20(1):83–97.

Potters, J. A. M., Curiel, I. J., & Tijs, S. H. (1992). Traveling salesman games. *Mathematical Programming, 53*(1), 199–211.

Prim, R. C. (1957). Shortest connection networks and some generalizations. *Bell Labs Technical Journal, 36*(6), 1389–1401.

Quak, H., Tavasszy, L. (2011). Customized solutions for sustainable city logistics: The viability of urban freight consolidation centres. In *Transitions towards sustainable mobility* (pp. 213–233). Berlin: Springer.

Schmeidler, D. (1969). The nucleolus of a characteristic function game. *SIAM Journal on Applied Mathematics, 17*(6), 1163–1170.

Shapley, L. (1953). *A value for n-person games*, volume II, chapter Contributions to the Theory of Games (pp. 307–317). Princeton, NJ: Princeton University Press.

Shapley, L. (1971). Cores of convex games. *International Journal of Game Theory, 1*(1), 11–26.

Shapley, L., & Shubik, M. (1966). Quasi-cores in a monetary economy with nonconvex preferences. *Econometrica: Journal of the Econometric Society, 34*(4), 805–827.

Shapley, L. S. (1955). Markets as cooperative games. Technical report, Rand Corporation Paper P-629.

Shapley, L. S. (1967). On balanced sets and cores. *Naval Research Logistics, 14*(4), 453–460.

Sharkey, W. W. (1995). *Handbooks in Operations Research and Management Science*, (Vol. 8, pp. 713–765), chapter Network models in economics. Network routing. Amsterdam: Elsevier.

Skorin-Kapov, D. (1998). Hub network games. *Networks, 31*(4), 293–302.

Skorin-Kapov, D. (2001). On cost allocation in hub-like networks. *Annals of Operations Research, 106*(1), 63–78.

Sprumont, Y. (1990). Population monotonic allocation schemes for cooperative games with transferable utility. *Games and Economic Behavior, 2*(4), 378–394.

A. Tamir. (1989). On the core of a traveling salesman cost allocation game. *Operations Research Letters, 8*(1), 31–34.

Tamir, A. (1991). On the core of network synthesis games. *Mathematical Programming, 50*(1), 123–135.

Tijs, S. H., & Driessen, T. S. H. (1986). Game theory and cost allocation problems. *Management Science, 32*(8), 1015–1028.

Vanovermeire, C., Sörensen, K., Van Breedam, A., Vannieuwenhuyse, B., & Verstrepen, S. (2014). Horizontal logistics collaboration: Decreasing costs through flexibility and an adequate cost allocation strategy. *International Journal of Logistics Research and Applications, 17*(4), 339–355.

Vornhusen, B., Wang, X., & Kopfer, H. (2014). Vehicle Routing under Consideration of Transhipment in Horizontal Coalitions of Freight Carriers. *Procedia CIRP, 19*(1), 117–122.

Wang, X., & Kopfer, H. (2014). Collaborative transportation planning of less-than-truckload freight. *OR Spectrum, 36*, 357–380.

Wang, Y., Ma, X., Li, Z., Liu, Y., Xu, M., & Wang, Y. (2017). Profit distribution in collaborative multiple centers vehicle routing problem. *Journal of Cleaner Production, 144*, 203–219.

Xu, D., & Du, D. (2006). The k-level facility location game. *Operations Research Letters, 34*(4), 421–426.

Zhao, J. (2018). Three little-known and yet still significant contributions of Lloyd Shapley. *Games and Economic Behavior, 108*, 592–599.

Index