CIRRELT

Centre interuniversitaire de recherche
sur les réseaux d'entreprise, la logistique et le transport

Interuniversity Research Centre
on Enterprise Networks, Logistics and Transportation

Thèse de doctorat

Lagrangian-Based Methods for Single and Multi-Layer Multicommodity Capacitated Network Design

Mohammad Rahim Akhavan Kazemzadeh

February 2019

CIRRELT-2019-07

Bureaux de Montréal :
Université de Montréal
Pavillon André-Aisenstadt
C.P. 6128, succursale Centre-ville
Montréal (Québec)
Canada H3C 3J7
Téléphone : 514 343-7575
Télécopie : 514 343-7121

Bureaux de Québec :
Université Laval
Pavillon Palasis-Prince
2325, de la Terrasse, bureau 2642
Québec (Québec)
Canada G1V 0A6
Téléphone : 418 656-2073
Télécopie : 418 656-2624

www.cirrelt.ca

Université de Montréal

Lagrangian-Based Methods for Single and Multi-layer Multicommodity Capacitated Network Design

par
Mohammad Rahim Akhavan Kazemzadeh

Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Thèse présentée à la Faculté des études supérieures
en vue de l'obtention du grade de Philosophiæ Doctor (Ph.D.)
en computer science

December, 2018

RÉSUMÉ

Le problème de conception de réseau avec coûts fixes et capacités (MCFND) et le problème de conception de réseau multicouches (MLND) sont parmi les problèmes de conception de réseau les plus importants. Dans le problème MCFND monocouche, plusieurs produits doivent être acheminés entre des paires origine-destination différentes d'un réseau potentiel donné. Des liaisons doivent être ouvertes pour acheminer les produits, chaque liaison ayant une capacité donnée. Le problème est de trouver la conception du réseau à coût minimum de sorte que les demandes soient satisfaites et que les capacités soient respectées. Dans le problème MLND, il existe plusieurs réseaux potentiels, chacun correspondant à une couche donnée. Dans chaque couche, les demandes pour un ensemble de produits doivent être satisfaites. Pour ouvrir un lien dans une couche particulière, une chaîne de liens de support dans une autre couche doit être ouverte. Nous abordons le problème de conception de réseau multiproduits multicouches à flot unique avec coûts fixes et capacités (MSMCFND), où les produits doivent être acheminés uniquement dans l'une des couches.

Les algorithmes basés sur la relaxation lagrangienne sont l'une des méthodes de résolution les plus efficaces pour résoudre les problèmes de conception de réseau. Nous présentons de nouvelles relaxations à base de nœuds, où le sous-problème résultant se décompose par nœud. Nous montrons que la décomposition lagrangienne améliore significativement les limites des relaxations traditionnelles.

Les problèmes de conception du réseau ont été étudiés dans la littérature. Cependant, ces dernières années, des applications intéressantes des problèmes MLND sont apparues, qui ne sont pas couvertes dans ces études. Nous présentons un examen des problèmes de MLND et proposons une formulation générale pour le MLND. Nous proposons également une formulation générale et une méthodologie de relaxation lagrangienne efficace pour le problème MMCFND. La méthode est compétitive avec un logiciel commercial de programmation en nombres entiers, et donne généralement de meilleurs résultats.

Mots cles : Conception de réseau, relaxation lagrangienne, méthode de sous-gradient, méthode de faisceaux, programmation en nombres entiers.

ABSTRACT

The multicommodity capacitated fixed-charge network design problem (MCFND) and the multilayer network design problem (MLND) are among the most important network design problems. In the single-layer MCFND problem, several commodities have to be routed between different origin-destination pairs of a given potential network. Appropriate capacitated links have to be opened to route the commodities. The problem is to find the minimum cost design and routing such that the demands are satisfied and the capacities are respected. In the MLND, there are several potential networks, each at a given layer. In each network, the flow requirements for a set of commodities must be satisfied. However, the selection of the links is interdependent. To open a link in a particular layer, a chain of supporting links in another layer has to be opened. We address the multilayer single flow-type multicommodity capacitated fixed-charge network design problem (MSMCFND), where commodities are routed only in one of the layers.

Lagrangian-based algorithms are one of the most effective solution methods to solve network design problems. The traditional Lagrangian relaxations for the MCFND problem are the flow and knapsack relaxations, where the resulting Lagrangian subproblems decompose by commodity and by arc, respectively. We present new node-based relaxations, where the resulting subproblem decomposes by node. We show that the Lagrangian dual bound improves significantly upon the bounds of the traditional relaxations. We also propose a Lagrangian-based algorithm to obtain upper bounds.

Network design problems have been the object of extensive literature reviews. However, in recent years, interesting applications of multilayer problems have appeared that are not covered in these surveys. We present a review of multilayer problems and propose a general formulation for the MLND. We also propose a general formulation and an efficient Lagrangian-based solution methodology for the MMCFND problem. The method is competitive with (and often significantly better than) a state-of-the-art mixed-integer programming solver on a large set of randomly generated instances.

Keywords: Network Design, Multilayer Network Design, Lagrangian Relaxation, Subgradient Method, Bundle Method, Mixed-Integer Programming.

CONTENTS

RÉSUMÉ . . . . . **ii**

ABSTRACT . . . . . **iii**

CONTENTS . . . . . **iv**

LIST OF TABLES . . . . . **viii**

LIST OF FIGURES . . . . . **ix**

LIST OF ABBREVIATIONS . . . . . **x**

DEDICATION . . . . . **xi**

ACKNOWLEDGMENTS . . . . . **xii**

CHAPTER 1: INTRODUCTION **1**

CHAPTER 2: LITERATURE REVIEW **6**

2.1 An Overview of Network Design Problems 6

2.1.1 Directed or Undirected 6

2.1.2 Multicommodity or Single-Commodity 7

2.1.3 Capacitated or Uncapacitated 7

2.1.4 Flow and Design Cost 8

2.1.5 Bifurcated or Non-Bifurcated Flow 8

2.1.6 Side Constraints: Topological and Budget Constraints 8

2.1.7 Time-Dependency 9

2.1.8 Deterministic or Stochastic 9

2.1.9 Single-Layer or Multilayer 9

2.2 An Overview of Lagrangian relaxation for Mixed-Integer Programming 10

2.2.1 Subgradient Method 11

2.2.2 Bundle Method 11
2.3 Multicommodity Capacitated Fixed-Charge Network Design Problem: Formulations and Solution Methods 12
2.3.1 Problem Statement 12
2.3.2 Formulations 13
2.3.3 Solution Methods 15
2.3.4 Lagrangian-Based Algorithms 19
2.4 Discussion and Summary 22

CHAPTER 3: NODE-BASED LAGRANGIAN RELAXATIONS FOR MULTICOMMODITY CAPACITATED FIXED-CHARGE NETWORK DESIGN 25
3.1 Introduction 25
3.2 Literature Review 27
3.3 Multicommodity Capacitated Fixed-charge Network Design Formulation 29
3.4 Node-Based Lagrangian Relaxations 32
3.4.1 First Reformulation-Lagrangian Relaxation 33
3.4.2 Second Reformulation-Lagrangian Relaxation 36
3.4.3 Third Reformulation-Lagrangian Relaxation 42
3.5 Solving Lagrangian Dual 46
3.5.1 Subgradient Method 47
3.5.2 Bundle Method 47
3.6 Primal Heuristic 48
3.7 The Proposed Algorithm 50
3.7.1 Initialization 51
3.7.2 Lagrangian Phase 51
3.7.3 Post-Optimization Phase 55
3.7.4 Summary 55
3.8 Experimental Results 56
3.8.1 Parameter Setting 58

3.8.2 Lower Bound Analysis 59
3.8.3 Upper Bound Analysis 61
3.9 Conclusion 68

CHAPTER 4: A SURVEY OF MULTILAYER NETWORK DESIGN . . 71
4.1 Introduction 71
4.2 Multilayer Network Design, Definition and Formulation 74
4.2.1 Definition 74
4.2.2 Formulation 77
4.3 Multilayer Network Design Taxonomy 82
4.3.1 Multilayer Network Design Classification 82
4.3.2 Overview of the Literature 83
4.4 Two-Layer Network Design Problems with Design Connectivity 84
4.4.1 Service Network Design with Resource Management 85
4.4.2 Telecommunications Applications 88
4.5 Two-Layer Network Design Problems with Flow Connectivity 92
4.5.1 Integrated Crew Scheduling and Aircraft Routing: Literature Review 92
4.5.2 MLND Formulation for Integrated Crew Scheduling and Aircraft Routing 94
4.6 Three-Layer Network Design Problems with One-to-Many Flow Connectivity 95
4.6.1 Integrated Crew Pairing and Assignment: Literature Review . . 96
4.6.2 MLND Formulation for Integrated Crew Pairing and Assignment 96
4.7 Three-Layer Network Design Problems with One-to-One Flow-Design Connectivity 98
4.7.1 Integrated Rail Freight Service Network Design: Literature Review 98
4.7.2 MLND Formulation for Integrated Rail Freight Service Network Design 99

4.8 Solution Approaches for Multilayer Network Design Problems 101
4.8.1 Exact Solution Methods 101
4.8.2 Heuristic Solution Methods 106
4.9 Conclusions 108

CHAPTER 5: A LAGRANGIAN-BASED MATHEURISTIC FOR MULTILAYER SINGLE FLOW-TYPE MULTICOMMODITY CAPACITATED FIXED-CHARGE NETWORK DESIGN . . 110
5.1 Introduction 110
5.2 Problem Statement and Formulation 113
5.2.1 Problem Definition 113
5.2.2 Formulation 115
5.2.3 Preprocessing Procedure 117
5.3 Bounding Procedures 118
5.3.1 Computing Lower Bounds through Lagrangian Relaxation . . . 118
5.3.2 Computing Upper Bounds through Slope Scaling 120
5.4 The Lagrangian-Based Matheuristic 123
5.4.1 Initialization Phase 124
5.4.2 Lagrangian Phase 124
5.4.3 Post-Optimization Phase 127
5.4.4 Summary 128
5.5 Experimental Results 128
5.5.1 Instances 129
5.5.2 Parameter Setting 130
5.5.3 Upper Bound Analysis 132
5.5.4 Lower Bound Analysis 135
5.6 Conclusions 136

CHAPTER 6: CONCLUSION 141

BIBLIOGRAPHY 143

LIST OF TABLES

3.I	C and C+ Instances of the MCFND	58
3.II	Values of parameters tested for relaxations and Bundle method	59
3.III	Values of parameters tested for the primal heuristic	60
3.IV	Values of parameters tested for the primal heuristic	60
3.V	Lagrangian lower bounds comparison with strong LP lower bound	62
3.VI	Lagrangian lower bounds comparison with best known upper bound	63
3.VII	Upper bound comparison of different Lagrangian Heuristics	66
3.VIII	Proposed heuristic VS state-of-the-art heuristics in literature	67
3.IX	Comparing normalized computational time of proposed heuristic to state-of-the-art heuristics in literature	69
5.I	Instances	130
5.II	Values of parameters tested for subgradient method	131
5.III	Values of parameters tested for the primal heuristic	132
5.IV	CPLEX and the proposed algorithm comparison for MLTS1 (2-layer instances)	133
5.V	CPLEX and the proposed algorithm comparison for MLTS2 (3-layer instances)	138
5.VI	Performance analysis of the proposed algorithm in terms of $\|K\|$, $\|L\|$, $\|A\|$ and $\|N\|$.	138
5.VII	Lower bound analysis: comparing proposed Lagrangian to CPLEX root node and strong LP for MLTS1.	139
5.VIII	Lower bound analysis: comparing proposed Lagrangian to CPLEX root node and strong LP for MLTS2.	140

LIST OF FIGURES

3.1 Small network example. 32
3.2 Capacitated facility location subproblem of the location relaxation 36
3.3 Average time of Lagrangian relaxation bounding procedures and Strong LP relaxation 64
3.4 Elapsed time analysis of Lagrangian relaxation bounding procedures 64
3.5 Average number of iterations of Lagrangian relaxation bounding procedures 64
3.6 Time analysis of Lagrangian-based heuristics 68

4.1 Multilayer network design example illustrating design connectivity constraints: arc 1 in layer l' is supported by the path made of arcs 3 and 4 in layer l, and arc 2 in layer l' is supported by the path made of arcs 5, 6, and 4 77
4.2 Multilayer network design example showing flow connectivity constraints contradict flow conservation constraints 81
4.3 Classification dimensions of multilayer network design problems . 83
4.4 Classification of existing multilayer network design problems in the literature 84
4.5 Resource cycles and their supported services in the service network design problem with resource management 86
4.6 A two-layer telecommunications network 89

LIST OF ABBREVIATIONS

DW	Dantzig-Wolfe
IP	Integer Programming
LP	Linear Programming
MCFND	Multicommodity Capacitated Fixed-Charge Network Design
MIP	Mixed-Integer Programming
MSMCFND	Multilayer Single Flow-Type Multicommodity Capacitated Fixed-Charge Network Design
MMCNF	Multilayer Multicommodity Capacitated Network Flow
MLND	Multilayer Network Design
NDO	Non-Differentiable Optimization

To my family.

ACKNOWLEDGMENTS

First and foremost, I would like to express my most sincere gratitude to my supervisors, Professors Bernard Gendron and Teodor Gabriel Crainic for their advice and strong support not only on research but also in my career path. I was so honored and fortunate to work with them in my Ph.D. program.

I would like to thank the academic and technical staff of the Department of Computer Science and Operations Research of the Université de Montréal and the Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT) particularly Serge Bisaillon for his technical help and advice. I would like to thank Professors Antonio Frangioni, Enrico Gorgone, and Tolga Bektas for their insights, advice, and collaboration on the node-based Lagrangian relaxations.

This list of acknowledgments would be incomplete without my most sincere appreciation to my family. I have to thank my mother and father who always believed in me and supported me to reach all the goals I have for my life. I want to express my sincere appreciation to my wife, Faezeh, who is always helping me especially in studying and living abroad. I cannot imagine any success in my life without her support. My best wishes go to my daughter and son, Sana and Mahdi, who were born during my Ph.D. program and boosts emotionally everything.

Finally, last but by no means least, I gratefully acknowledges the support of the Université de Montréal, through its end of doctoral studies scholarship and the excellence scholarship of the Department of Computer Science and Operations Research, as well as the support of CIRRELT through its doctoral excellence scholarship. I also gratefully acknowledge the support of Fonds de recherche du Québec through their international internship grant which helps me to go to Italy, University of Pisa, and improve my research.

CHAPTER 1

INTRODUCTION

Network design is a well-known class of problems in combinatorial optimization. The *multicommodity capacitated fixed-charge network design problem* (MCFND), with interesting applications in transportation, telecommunications, logistics and production planning [64, 70], as well as the *multilayer network design problem* (MLND) that recently presented interesting applications in the fields of transportation [17, 31, 88] and telecommunications [33, 61], are among the most important network design problems.

In the single-layer MCFND, several commodities, such as goods, data or people, have to be routed between different origin-destination pairs of a given potential network. The flow of each commodity is permitted to split on different paths from the origin to destination. A predefined maximum flow (capacity) can be routed on each link of the network. A network has to be designed by selecting appropriate links to route the commodities. A *fixed design cost* has to be paid to open a link. In addition, a variable *flow cost* is imposed to route each unit of commodity demand on each link. The problem is to find the minimum cost design and routing such that the demands and the capacities are satisfied.

In multilayer network design, however, unlike a typical network design problem, there are several networks, each at a given layer. Each network has its nodes, potential arcs with (or without) limited capacities, and, possibly, commodities. The commodities, if any, need to be routed from their origins to their destinations in each layer to satisfy the demands. To route the commodities, appropriate arcs have to be selected (or opened) by paying a fixed cost. A particular layer might not have any commodity, but still has to be designed to support the routing of other layers. At least one layer has commodities to route. There are two types of coupling constraints between the layers, flow connectivity and design connectivity requirements. A common type of design connectivity requirements arises when each link in a layer can be selected only if some arcs (typically forming a path or a cycle) are opened in another layer. The flows in a layer

might also be related to the flows of another layer, corresponding to flow connectivity requirements. For example, the amount of flow on each arc in a particular layer might be computed based on the flow on several arcs in another layer. In general, the objective is to find a minimum cost design and routing for all layers, while satisfying typical network design constraints in each layer, as well as coupling constraints between layers.

Among different versions of the MLND problems, in this thesis, we address a version of the problem called the *multilayer single flow-type multicommodity capacitated fixed-charge network design problem* (MSMCFND), where commodities are routed only in layer $l = 1$, given a set of layers $L = \{1, 2, .., |L|\}$. The amount of flow on each link in a particular layer $l \geq 2$ is equal to the amount of flow on its corresponding links in layer $l - 1$ and consequently to the amount of flow on its corresponding links in layer $l = 1$. The application that motivates our work on the MMCFND is described in Zhu et al. [88], which proposes a three-layer network to model a problem in rail freight transportation planning where a double consolidation policy is performed. First, cars are grouped into so-called blocks (blocking process). Then, they are grouped into services to make up trains (train make-up process). The next step is to select the services and define their frequencies. To propose a formulation determining blocking, train make-up, and service selection decisions simultaneously, Zhu et al. [88] considered a three-layer network including: 1) a car layer that consists of links on which cars are moved in each terminal; 2) a block layer that includes block links from the origins to the destinations of the blocks; and 3) a service layer that includes service links to support block movements. In the car layer, the flows of commodities are moved via the car links and the projected block links from the block layer. To design a block link, a chain of supporting services has to be opened in the service layer. The flow of each service is equal to the summation of the flows on all its supported blocks. The problem is to find a minimum cost blocking and service design, and flow routing of the cars, while considering flow capacity of the blocks and the services, blocking capacity of each terminal, and design connectivity of the block and the service layers.

Lagrangian-based algorithms are one of the most effective solution methods to solve network design problems, in particular the single-layer MCFND (see [27] and [53] for

example). The single-layer MCFND is typically formulated as a Mixed-Integer Programming (MIP) model including two main sets of constraints: 1) *flow conservation constraints* ensuring that the demands of the commodities are satisfied, and 2) *flow capacity constraints* ensuring that the flow on each arc does not exceed its capacity. The usual Lagrangian relaxations for the formulation are the so-called shortest path and knapsack relaxations, which are obtained, respectively, by relaxing capacity constraints and flow conservation equations. For the first one, the resulting Lagrangian subproblem decomposes into a collection of shortest path subproblems, one for each commodity, while the second one allows solving the Lagrangian subproblem as a series of continuous knapsack subproblems, one for each arc. Therefore, the shortest path and knapsack relaxations are also known as *commodity-based* and *arc-based* relaxations, respectively. The nodes of a network are the other entities that can be considered as decomposition components. The first goal of this thesis is to present new relaxations where the Lagrangian subproblem decomposes by node to propose a new Lagrangian-based matheuristic to solve the problem.

The applications of network design models and their solution techniques have been surveyed in [23, 64, 71]. However, in recent years, interesting applications of multilayer network design have appeared, that are not covered in these surveys. To the best of our knowledge, the only survey paper on multilayer networks is [59] which studies the general concept of multilayer networks, but does not cover the multilayer network design problem. In addition, several research works, particularly in transportation (crew scheduling [17] and railway engineering [88], for example), are missing in this survey. Therefore, the second objective of this thesis is to fill this gap. We propose a new classification of multilayer network design problems, and we synthesize its applications in transportation and telecommunications, as well as the methods used to solve these problems. Our synthesis emphasizes the multilayer structure of the problems such as the number of layers, and the coupling constraints between layers, rather than general network design characteristics. This focus allows to understand the multilayer network design features better and to identify new research avenues. We propose a general formulation to facilitate the exposition of multilayer network design problem while covering

many applications.

Motivated by the success of Lagrangian-based heuristics for the single-layer MCFND, we propose a Lagrangian-based matheuristic for the MSMCFND. Although there are effective Lagrangian-based heuristics for the MCFND (see [53], [82], [27], and [60]), these methods cannot be adapted in a straightforward way to the MMCFND. Indeed, the challenge in multilayer network design is how to handle the significant additional complexity incurred by the coupling constraints between layers. On the one hand, these additional constraints complicate the task of developing Lagrangian relaxations that keep a balance between the quality of the lower bound and the computational efficiency of solving the Lagrangian subproblem. On the other hand, the derivation of effective feasible solutions becomes significantly more difficult in the presence of these constraints. We propose a Lagrangian-based matheuristic solution method based on a *slope scaling* scheme. The idea of slope scaling is to iteratively solve a linear multicommodity flow formulation, and to use the flow distribution to adjust the linear approximation at the next iteration. When the slope scaling method stalls, a perturbation move changes the initial linear approximation to start a new slope scaling procedure; for more details on slope scaling, see [27] and [56]. The Lagrangian relaxation approach provides not only the lower bounds to evaluate the obtained feasible solutions, but also guides the slope scaling search by providing initial solutions and by defining the perturbation moves.

The remainder of this thesis is organized as follows. Chapter 2 provides: 1) a literature review on different versions of network design problems as well as the formulations and solution methods for the MCFND, and 2) an overview on the existing Lagrangian relaxation methods for the MCFND. In Chapter 3, we present new node-based Lagrangian relaxations, where the resulting Lagrangian subproblem decomposes by node. We show that the Lagrangian dual bound improves upon the so-called strong linear programming (LP) bound, known to be equal to the Lagrangian dual bounds of the shortest path and knapsack relaxations. We also propose a Lagrangian-based matheuristic algorithm to obtain upper bounds. The results show that the proposed algorithm is competitive with the state-of-the-art heuristics in the literature on a large set of benchmark instances.

In Chapter 4, we propose a general formulation for the multilayer network design

problem, which can cover a wide range of applications. We also present a state-of-the-art review and synthesis of multilayer network design problems and discuss their solution approaches in transportation and telecommunications. In Chapter 5, we propose a general formulation for the MSMCFND. We also propose an efficient solution methodology based on Lagrangian relaxation to solve the large-scale MSMCFND instances. The proposed algorithm is enhanced by intensification, diversification, and post-optimization procedures using long-term and short-term memories. The results show that the resulting algorithm is competitive with (and often significantly better than) a state-of-the-art MIP solver on a large set of randomly generated instances, not only with respect to the obtained upper bounds, but also in terms of optimality gaps. Finally, Chapter 6 summarizes the thesis and proposes potential future research directions.

Chapters 3 to 5 are the preliminary versions of research articles co-authored with my thesis supervisors, and Chapter 3, with three other researchers. The article of Chapter 3, co-authored with Tolga Bektaş, Teodor Gabriel Crainic, Antonio Frangioni, Bernard Gendron, and Enrico Gorgone, is under final algorithmic implementations. The articles of Chapters 4 and 5, co-authored with Teodor Gabriel Crainic and Bernard Gendron, have been submitted to, respectively, *European Journal of Operational Research* and *Computers and Operations Research*. My role in these three chapters include the development, implementation, and testing the algorithms and formulations, as well as writing the papers.

CHAPTER 2

LITERATURE REVIEW

In this chapter, we first provide an overview of the main variants of network design problems. Then, we present different formulations of the MCFND and the solution methods proposed in the literature, with a focus on Lagrangian relaxation methods. Finally, we discuss research avenues to conclude the chapter.

2.1 An Overview of Network Design Problems

Network design has a wide range of applications for planning and operations in transportation, telecommunications, logistics, and distribution systems. In the context of transportation systems, such applications are, for example, to decide the construction or improvement of infrastructures and facilities, and the selection of transportation services, their frequencies, and schedules.

In network design problems, we are given a graph including nodes and links and several demands between pairs of nodes. The common feature of all network design problems is the decision to be made about selecting a subset of links to route the demands. A fixed cost, associated with some or all of the links, has to be paid for using or selecting a link. A variable cost, that depends on the volume of traffic, might also be associated with each link. The objective is to select a subset of links in the network to satisfy the demands and minimize the total cost. Network design problems can be categorized into different classes. Many survey papers with classification schemes exist in the literature [23, 64, 71]. In the following subsections, we summarize some classes of network design problems.

2.1.1 Directed or Undirected

In network design problems, the graph is either directed or undirected. In directed graphs, a link can be used only in a particular direction, while in an undirected graph,

one can use a link in both directions. An undirected graph can be transformed into a directed graph using pairs of links in both directions.

2.1.2 Multicommodity or Single-Commodity

A commodity is a product or a data type that has to be moved between origins and destinations. The origins provide supplies of the commodity, while the destinations require demands of the commodity. When there are several commodities with different origins and destinations, the resulting problem is called *multicommodity network design*. In this case, the resources and the capacities of the links are shared between the commodities. There are applications where only a single commodity has to be moved, known as *single-commodity network design*. There are also some special applications where the commodities have to be routed from several origins to a single destination. For example, in a raw material distribution application, several origins serve the demand of a single destination. In addition, one can see the same commodity type with different origins and destinations as different commodities, each with several origins and a single destination, resulting in a multicommodity network design problem.

2.1.3 Capacitated or Uncapacitated

Capacity is a property of the network that can be defined for each link (or node). It limits the maximum flow of the commodities that can be routed through the link (or node). In undirected graphs, the link capacity limits the total flow that can be routed in both directions. If the link (or node) is selected to be part of the network design, then the corresponding capacity is available. Otherwise, the link is closed, and there is no capacity to route the flow of the commodities. Sometimes, the problem is to determine the number of facilities, each with a given capacity, installed on the links (or nodes), as in the network loading [65, 71].

A node capacity can be converted into a link capacity by splitting each node into two nodes and defining a link between them with a capacity equal to the node capacity. Therefore, there is no loss of generality in considering only link capacities.

In a *capacitated* network design problem, each link has a limited flow capacity equal to the resource or facility capacity installed on the link. When the capacity of each link is at least equal to the total demands of all commodities, then the problem is called *uncapacitated* network design problem.

2.1.4 Flow and Design Cost

Typically, there are two types of costs in network design problems. The flow variable cost is the per unit cost of flow for each commodity on each link. This cost depends on the type of commodity and the link. The design cost is the cost of installing or adding facilities to a link. When the problem is only to decide opening and closing the links (not installing a number of facilities on the links), then the cost is called fixed cost or *fixed charge*.

2.1.5 Bifurcated or Non-Bifurcated Flow

Each commodity's flow can be either bifurcated or non-bifurcated. The former means that each commodity's flow can be split and routed on multiple paths, i.e., flow-splitting or bifurcated routing is allowed, while the latter means that the flow of each commodity has to be routed through a single path from its origin to its destination.

2.1.6 Side Constraints: Topological and Budget Constraints

There are several side constraints that can be added to a network design problem among which *topological* and *budget constraints* are the most important ones. Topological restrictions are imposed upon the configuration of the network. The examples are: 1) forcing the final design to be a spanning tree, and 2) limiting the resources shared by several links; see [23] for more details. There are several types of budget constraints, which restrict the design or routing costs to a limited budget. The examples are: 1) limiting the maximum cost of building a network to a total budget, and 2) forcing the maximum cost of routing each commodity to a limited available budget.

2.1.7 Time-Dependency

Sometimes the form of the network, the resource availability, and the flow of the commodities are time-dependent resulting in *time-dependent network design*. One way to model a given time-dependent network design problem is to attach the time dimension to the network by building a *time-space* network in which the physical nodes (terminals or stations in transportation planning, for example) are duplicated on the time horizon to represent the time dependency. A node in such a network represents a terminal at a specific time, and each link represents a transfer from a terminal at a specific time to either the same or a different terminal at another time in the future (see [88] for more details).

2.1.8 Deterministic or Stochastic

Most network design problems in the literature are deterministic: they assume that there is perfect knowledge of the data. However, in practice, the data is uncertain. For instance, the demands and the costs might change in the future depending on different situations. Stochastic programming is a way to deal with uncertainty when probability distributions are known. For example, when demands are uncertain, a two-stage stochastic programming model can be considered, where the first stage focuses on the link selection decisions, while the second stage focuses on the routing decisions, which are directly influenced by the uncertain demands. Several scenarios can be generated to represent the uncertainty of the second stage, and the model seeks to minimize the cost of the design at the first stage plus the expected routing cost over all scenarios at the second stage (see [30] for more details).

2.1.9 Single-Layer or Multilayer

As mentioned before, the single-layer network design problem is to find the minimum cost design and routing on a given potential network such that the demands of commodities are satisfied. In multilayer network design, there are several networks, each at a given layer, where any link in that layer corresponds to a chain of links (a path

or a cycle) of another layer. To open (or design) a link in a given layer, the corresponding path or cycle in the other layer has to be opened. The flows in a layer might also be related to the flows of another layer.

2.2 An Overview of Lagrangian relaxation for Mixed-Integer Programming

Consider the following general *Mixed-Integer Programming* (MIP) prolem:

$$(\mathscr{P}_{MIP}) \quad Z_{MIP} = \min \quad cx \tag{2.1}$$

$$A^1 x \leq b^1 \tag{2.2}$$

$$A^2 x \leq b^2 \tag{2.3}$$

$$x \geq 0 \quad \text{and integer} \tag{2.4}$$

where A^1 and A^2 are, respectively, $m^1 \times n$ and $m^2 \times n$ matrices, and b^1 and b^2 are, respectively, vectors of dimension m^1 and m^2. Then, the following Lagrangian relaxation of $\mathscr{P}_{MIP}$ is obtained by relaxing constraints (2.2) and by letting $\alpha \geq 0, \alpha \in R^n$, be the corresponding Lagrange multipliers:

$$\begin{aligned} (\mathscr{P}_{LR}(\alpha)) \quad Z_{LR}(\alpha) &= \min cx + \alpha(A^1 x - b^1) \\ s.t. \quad & (2.3), (2.4) \end{aligned} \tag{2.5}$$

Given a particular vector of multipliers α, $\mathscr{P}_{LR}(\alpha)$ is the Lagrangian subproblem. The Lagrangian dual, $Z_{LD} = \max_\alpha Z_{LR}(\alpha)$, should then be solved to find the best possible lower bound. Note that $Z_{LD} \geq Z_{LP}$ where Z_{LP} is the LP relaxation bound obtained by dropping integrality constraints. However, if the Lagrangian subproblem has the integrality property [46], i.e., if its LP relaxation always has an integral optimal solution, then the Lagrangian dual provides the same optimal value as that of the LP relaxation lower bound ($Z_{LD} = Z_{LP}$).

The Lagrangian function $Z_{LR}(\alpha)$ is concave, but unfortunately non-differentiable.

However, it is easy to compute a subgradient direction for any choice of the Lagrange multipliers. There are several methods to solve the Lagrangian dual problem among which the *subgradient* and *bundle* methods are the most popular and efficient ones. We describe these methods in the following subsections.

2.2.1 Subgradient Method

A subgradient method includes moving from a current point α_i to a new point α_{i+1} using a step size s_i and a direction d_i in the following simple formula:

$$\alpha_{i+1} = \alpha_i + s_i d_i \tag{2.6}$$

In a simple subgradient method, d_i is considered to be equal to the current subgradient g_i computed as the violation of the relaxed constraints $(A^1x - b^1)$ when the variables (x) are fixed to the values found by solving the Lagrangian subproblem in the current point α_i. A simple approach to compute the step size is $s_i = \mu_i \left(Z^e_{LD} - Z(\alpha_i)\right)/g_i d_i$ where Z^e_{LD} and μ_i are an estimation for the value of Z_{LD} and a scaling parameter, respectively. For the details on the different strategies to compute the step size and the scaling parameter, see [41].

2.2.2 Bundle Method

The idea of the bundle method is to compute the direction using a bundle B of the subgradients found so far. The direction d_i with respect to the current point α_i is found by solving the following quadratic problem:

$$\min_{\theta} \left\{ \frac{1}{2} t \| \sum_{b \in B} g_b \theta_b \|^2 + \sum_{b \in B} E_b \theta_b; \quad s.t. \quad \sum_{b \in B} \theta_b = 1, \theta \geq 0 \right\} \tag{2.7}$$

where for each bundle member $b \in B$, g_b and $t > 0$ are the corresponding subgradient vector and the so-called trust region parameter, respectively. $E_b = Z_{LR}(\alpha_b) + g_b(\alpha_i - \alpha_b) - Z_{LR}(\alpha_i)$ is the current linearization error. The ascent direction is then computed using the convex combination of the subgradients defined by θ. Valuable information

can be extracted from the solution θ to produce integer feasible solutions. The quadratic problem in bundle methods is the Augmented Lagrangian of the standard Dantzig-Wolfe (DW) approach, with a linear Lagrangian term related to the current point (stability center) and a quadratic term related to the stability parameter ($t > 0$). Let ε^{Lin} be the relative precision required, then a general stopping condition is when the bundle finds an epsilon-subgradient g such that $t^*||g|| \leq \varepsilon^{Lin}|Z^e_{LD}|$, where t^* is a parameter which is an estimate of the longest step that can be performed and $||g||$ is a norm-like function. Different estimates can be used for Z^e_{LD}, although using the best lower bound found so far is pretty common. A detailed explanation of the bundle method can be found in [39] and [26].

2.3 Multicommodity Capacitated Fixed-Charge Network Design Problem: Formulations and Solution Methods

In this thesis, we focus on the deterministic MCFND with bifurcated flows and directed links. The problem is assumed to have no side constraint. We also consider a multilayer version of the problem. In the following subsections, we state the MCFND and present its arc-based and path-based formulations.

2.3.1 Problem Statement

In the MCFND, there is a directed network with a set of nodes and a set of links. Several commodities have to be transferred between different pairs of nodes. Each link is characterized by 1) a fixed cost that has to be paid for using the link; 2) per unit flow variable costs incurred when routing flow of the commodities on the link; and 3) a capacity that limits the amount of flow of all commodities on the link. A network should be designed by selecting or opening links to route the commodities, while considering the capacities. The obtained network has to be the minimum cost network in terms of fixed and variable costs.

In a transportation network, for example, the nodes, links and commodities, depending on the specific application, may correspond to cities, roads or tracks, and goods,

respectively. In a telecommunications network, these elements might correspond to, for example, switch centers, transmission facilities, and data traffic, respectively.

2.3.2 Formulations

The MCFND is normally formulated in an arc-based or a path-based form [23] where the decision variables are associated with the arcs or the paths of the network, respectively. The following subsections present these formulations.

2.3.2.1 Arc-Based Formulation

The MCFND is defined on a directed graph $G = (N, A)$, where N is the set of nodes and A is the set of arcs. For each node $i \in N$, we define the sets of forward and backward neighbours, N_i^+ and N_i^-, respectively. Each commodity $k \in K$ corresponds to an origin-destination pair such that d^k units of flow must travel between the origin $O(k)$ and the destination $D(k)$. The objective function to be minimized includes a cost $c_{ij}^k \geq 0$ for routing one unit of commodity $k \in K$ through arc $(i, j) \in A$ and a fixed cost $f_{ij} \geq 0$ for using arc $(i, j) \in A$, thus providing a capacity $u_{ij} > 0$ on the arc. A classical model for the MCFND introduces two sets of variables: x_{ij}^k is the flow of commodity $k \in K$ on arc $(i, j) \in A$, while y_{ij} is 1, if arc $(i, j) \in A$ is used, and 0, otherwise. The model is written as follows:

$$\min \sum_{(i,j)\in A} \sum_{k\in K} c_{ij}^k x_{ij}^k + \sum_{(i,j)\in A} f_{ij} y_{ij} \tag{2.8}$$

$$\sum_{j\in N_i^+} x_{ij}^k - \sum_{j\in N_i^-} x_{ji}^k = b_i^k \qquad \forall i \in N, \forall k \in K \tag{2.9}$$

$$\sum_{k\in K} x_{ij}^k \leq u_{ij} y_{ij} \qquad \forall (i,j) \in A \tag{2.10}$$

$$x_{ij}^k \leq s_{ij}^k y_{ij} \qquad \forall (i,j) \in A, \forall k \in K \tag{2.11}$$

$$x_{ij}^k \geq 0 \qquad \forall (i,j) \in A, \forall k \in K \tag{2.12}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{2.13}$$

The objective function (2.8) is to minimize the total routing and design costs. Constraints (2.9) are the usual *flow conservation* equations ensuring that the demands are routed from the origins to the destinations, where:

$$b_i^k = \begin{cases} +d^k, & \text{if } i = O(k), \\ -d^k, & \text{if } i = D(k), \\ 0, & \text{otherwise,} \end{cases} \quad \forall i \in N, \forall k \in K.$$

Capacity constraints (2.10) ensure that the sum of the flows on each arc $(i, j) \in A$ does not exceed its capacity u_{ij}. These are also known as *linking constraints* because they ensure that no flow is allowed on arc $(i, j) \in A$, unless arc (i, j) is open and its fixed cost is paid. Constraints (2.11) are *strong linking constraints*, where $s_{ij}^k = \min(d^k, u_{ij})$. Although constraints (2.11) are redundant (valid) for the MIP model, adding these inequalities significantly improve the lower bounds for the linear programming (LP) relaxation [42]. Therefore, the LP relaxation with the strong inequalities is called *strong LP relaxation*, while the LP relaxation without the strong linking constraints is the *weak LP relaxation*. Constraints (2.12) and (2.12) define the domain of the decision variables. Continuous flow variables mean that the flow of each commodity is bifurcated.

2.3.2.2 Path-Based Formulation

Another formulation of the MCFND is the path-based model. Let P^k be the set of paths for commodity $k \in K$, and $P = \cup_{k \in K} P^k$. Let $\delta_{ij}^{pk} = 1$, if arc $(i, j) \in A$ belongs to path $p \in P^k$ (0, otherwise). We introduce a flow variable h_p^k which is the flow of commodity $k \in K$ on path $p \in P^k$. Then, the path-based formulation takes the following form:

$$min \quad \sum_{p \in P} \sum_{k \in K} s_p^k h_p^k + \sum_{(i,j) \in A} f_{ij} y_{ij} \tag{2.14}$$

s.t. (2.13) and

$$\sum_{p \in P^k} h_p^k = d^k \qquad \forall k \in K \qquad (2.15)$$

$$\sum_{k \in K} \sum_{p \in P^k} h_p^k \delta_{ij}^{pk} \leq u_{ij} y_{ij} \qquad \forall (i,j) \in A \qquad (2.16)$$

$$\sum_{p \in P^k} h_p^k \delta_{ij}^{pk} \leq s_{ij}^k y_{ij} \qquad \forall (i,j) \in A, \forall k \in K \qquad (2.17)$$

$$h_p^k \geq 0 \qquad \forall k \in K, \forall p \in P^k \qquad (2.18)$$

Constraints (2.15) ensure the demands are satisfied. Constraints (2.16) and (2.17) are the capacity and strong linking constraints, respectively. $s_p^k = \sum_{(i,j) \in A} c_{ij}^k \delta_{ij}^{pk}$ is the flow cost of commodity $k \in K$ on path $p \in P^k$. The arc-based flow variables are related to the path-based flow variables with the formula $x_{ij}^k = \sum_{p \in P^k} h_p^k \delta_{ij}^{pk}, \forall (i,j) \in A, \forall k \in K$.

2.3.3 Solution Methods

Cutting-plane algorithms, relaxations, heuristics, meta and matheuristics, and hybrid solution methods (combining heuristics and exact techniques) are the main solution approaches for the MCFND. In the following subsections, we review and explain these solution methods, except the relaxations. We provide a comprehensive review of the relaxation methods, particularly Lagrangian relaxation methods in Section 2.3.4.

2.3.3.1 Cutting-Plane Methods

Well-known valid inequalities for the MCFND are: the strong linking constraints (2.11), *cutset inequalities*, *cover inequalities*, *minimum cardinality inequalities*, *flow cover inequalities*, and *flow pack inequalities* [14]. For each subset of nodes $S \subset N$, a *cutset* $(S, \bar{S})$ is a subset of arcs with tail in S and head in $\bar{S} = N \setminus S$. The cutset inequalities then state that the capacity of a cutset must be sufficient to carry the demands of the commodities originating from the partition to its complementary set.

Cover and minimum cardinality inequalities are 0-1 knapsack structured inequalities. A subset of a cutset $(S, \bar{S})$ is said to be a *cover* C if the total capacity of the arcs in $(S, \bar{S}) \setminus C$ does not cover the demand of the cutset. The cover inequalities then ensures

that at least one arc from the cover C must be opened in order to meet the demand. Minimum cardinality inequalities derived by sorting the capacities of the arcs in a cutset in non-increasing order. These inequalities guarantee the least number of arcs in a cutset that must be used in every solution. To obtain the flow cover and flow pack inequalities, the flow variables are inserted into cutset inequalities; see [14] for more details.

In LP-relaxation-based methods, such as LP-based branch-and-bound, strong inequalities are able to improve the lower bounds significantly [42]. These inequalities, and the other inequalities mentioned above, can be included a priori to the model. However, this makes the LP relaxation too large and degenerate [44]. Therefore, an option is to generate these inequalities in a cutting-plane algorithm where they are added iteratively to the model to improve the lower bound. This approach has been applied in [14].

Chouman et al. [14] applied the strong, cover, minimum cardinality, flow cover, and flow pack inequalities in a cutting-plane algorithm. In the problem they address, the flow variable costs are independent of the commodities. Therefore, the commodities with the same origin can be aggregated into a single commodity. Without commodity-dependent costs, this transformation leads to the same optimal solution as the original commodity representation with one origin and one destination per commodity. They use 196 benchmark instances with up to 100 nodes, 700 arcs, and 400 commodities to test the algorithm. The results show that the strong linking inequalities are effective for the disaggregated representation of the commodities, while the other types of cuts are effective for the instances with many nodes and few commodities. Adding the cover inequalities along with the strong linking inequalities improve the lower bound, but for the instances with a large number of commodities (hundreds of commodities) the improvement is small and such instances are still very difficult to solve. To improve this approach and to solve larger instances, Gendron and Larose [43] added a column generation approach into a branch-and-cut procedure to obtain a branch-and-price-and-cut procedure. In this procedure, at each node of the branch-and-bound tree, the LP relaxation is solved by adding in a dynamic way not only the strong linking inequalities (row generation), but also the flow variables (column generation). The proposed method was

tested on the same set of 196 benchmark instances with up to 100 nodes, 700 arcs, and 400 commodities. This procedure can solve larger instances in terms of the number of commodities more efficiently than the cutting-plane method without column generation.

Cutting-plane algorithms based on the Benders decomposition method [9], which generate Benders cuts at each iteration, are presented in [19]. In addition, Costa et al. [20] compared the Benders, cutset, and metric inequalities for the network design problem. The metric inequalities define every possible capacity vector for which there is a feasible multicommodity flow by projecting out the flow variables from the formulation. They performed computational tests on a set of benchmark instances that contains up to 100 nodes, 700 links, and 400 commodities. The computational results show that the time for the Benders decomposition algorithm to reach a first feasible solution is improved by strengthening the Benders feasibility cuts to metric inequalities.

2.3.3.2 Heuristics

Network design problems are NP-hard, therefore solving them to optimality with an exact solution method is difficult for large-scale instances. A heuristic is a solution method that can obtain good or near-optimal solutions in a reasonable time. Unlike an exact solution method, a heuristic does not guarantee the optimality, but it is generally more time efficient than an exact method.

Kim and Pardalos [56] proposed a slope scaling algorithm to solve a fixed-charge transportation problem. Crainic et al. [27] adapted the approach to the MCFND, and improve its performance by adding a long-term memory. The information collected during the slope scaling procedure is used to perform intensification/diversification phases, perturb the linear approximation, and start a new procedure. A *capacity scaling* approach has been used in Katayama et al. [55]. Like slope scaling, capacity scaling is an iterative solution method, but it depends on changing the arc capacities using the flow volumes on the arcs.

A *tabu search* [49] solution method was proposed in Crainic et al. [25]. Tabu search is a metaheuristic solution method, which iteratively improves the current solution by moving to a neighborhood solution. It prevents revisiting previously searched solutions

by putting the visited solutions' attributes into a tabu list. The solutions containing these attributes can not be visited for a specific number of iterations. It also moves to a worse solution if no better solution is found. The neighborhood structure proposed in Crainic et al. [25] is based on simplex pivots. The proposed solution method works well for the large-scale instances. Ghamlouche et al. [47] proposed a cycle-based tabu search. The idea is based on searching the solution space by rerouting commodities around a cycle. The algorithm defines a neighborhood by introducing a cycle, and then reroute the commodities around the cycle, which opens and closes several arcs. The results show the powerful performance of the cycle-based neighborhood. This solution approach was then strengthened by Ghamlouche et al. [48] in a *path relinking* procedure. The feasible solutions from the tabu search procedure produce a reference set. This reference set is then used to produce a new solution by linking to a reference solution. The experiments show better performance of this solution method in comparison with the tabu search procedure. Kim et al. [57] introduced short-term memories to the algorithm of Crainic et al. [27] and integrated the search into a tabu search procedure.

Recently, an *evolutionary algorithm* has been applied to the MCFND in Paraskevopoulos et al. [77]. Evolutionary algorithms are a class of population-based metaheuristics where the algorithm tries to improve a population of solutions. The proposed algorithm evolves a population of solutions in which an iterated local search is interlinked as a post-improvement method. Besides, a new cycle-based neighborhood structure is proposed which can partially reroute multiple commodities. Computational results on a set of benchmark instances show that the proposed evolutionary algorithm can reach high-quality solutions in reasonable computational times.

2.3.3.3 Matheuristics

Matheuristics combine exact mathematical programming solution methods and heuristics to take advantage of both approaches. In Hewitt et al. [52], an innovative solution method is proposed for the bifurcated and non-bifurcated MCFND based on the integration of the arc-based and path-based formulations. It finds feasible solutions using a local search in the arc-based formulation and applies an LP relaxation in the path-based

formulation to find lower bounds. The information that is obtained during the solution of the path-based formulation is used in the arc-based formulation to find a better feasible solution. The proposed algorithm is compared with CPLEX and with the cycle-based tabu search proposed in Ghamlouche et al. [47]. The authors use instances with up to 500 nodes, 3,000 arcs, and 200 commodities. In comparison with CPLEX, the proposed solution approach is effective. For nearly all instances in their test set, the solution is better than the solution of the cycle-based tabu search algorithm, and this solution is found much faster. Rodríguez-Martín and José Salazar-González [80] applied *local branching* cuts to partition the solution, and used MIP solvers to solve each obtained local domain. This idea has also been applied by Fischetti et al. [37] to solve the multi-level network design problem.

Chouman and Crainic [13] proposed a hybrid approach based on a metaheuristic (tabu search) and a mathematical programming method. The mathematical programming algorithm works on a restricted formulation obtained through a partial variable fixing. The cycle-based tabu search tries to improve the obtained feasible solution by exploring the design variables neighborhood. The mathematical programming algorithm finds feasible solutions, while the tabu search improves these solutions by exploring a local area.

2.3.4 Lagrangian-Based Algorithms

In this section, we first explain the usual Lagrangian relaxations of the MCFND, and then we review the Lagrangian-based solution methods proposed for the MCFND. The existing Lagrangian methods can be categorized into: 1) Lagrangian-based heuristics, and 2) Lagrangian-based branch-and-bound methods. We explain these categories in the following subsections.

2.3.4.1 Lagrangian Relaxations for MCFND

The usual Lagrangian relaxations for the formulation are the so-called *knapsack* and *flow relaxations*, which are obtained, respectively, by relaxing flow conservation equa-

tions (2.9) and capacity constraints (2.10) as well as strong linking constraints (2.11). The first one allows solving the Lagrangian subproblem as a collection of continuous knapsack subproblems, one for each arc; while for the second one, the resulting Lagrangian subproblem decomposes into a collection of shortest path subproblems, one for each commodity. Therefore, the knapsack and flow relaxations are also known as *commodity-based* and *arc-based* relaxations, respectively.

2.3.4.2 Lagrangian-Based Heuristics

Gendron and Crainic [42] proposed and studied the classical Lagrangian relaxations including the flow and knapsack relaxations. They show that both Lagrangian relaxations yield the same theoretical lower bound. The LP relaxation also generates the same bound because the Lagrangian subproblems of both Lagrangian relaxations have the integrality property. The authors concluded that although the knapsack relaxation has nice practical and theoretical properties, more sophisticated procedures than the classical subgradient method need to be developed to find better bounds. The flow relaxation, however, obtains the best performances with respect to the quality of the lower bounds. They also proposed a Lagrangian-based primal heuristic to assess the quality of the generated bounds. At each iteration of the subgradient procedure, a subset of arcs is obtained by applying specific rules on the solution of the Lagrangian subproblem. A restricted flow problem is then solved where only the subset of arcs exists in order to find upper bounds. The best obtained upper bound is then used to guide the subgradient procedure.

Crainic et al. [27] presented a Lagrangian-based heuristic using a slope scaling approach that iteratively solves a linear approximation of the original formulation. In the proposed heuristic, the initial costs are set to heuristic values. When the slope scaling stalls, a perturbation method adjusts the costs to trigger another slope scaling procedure. Once the number of performed slope scaling procedures reaches a predefined maximum number, a bundle-based process is launched to find a lower bound and new Lagrange multipliers. The Lagrange multipliers are then used to reinitialize the linearized costs and start another slope scaling procedure followed by a predefined number of perturbations.

Crainic et al. [26] compared bundle and subgradient methods applied to optimize the Lagrangian dual of the two classical Lagrangian relaxations. The authors conclude that although bundle methods are more difficult to implement, they outperform subgradient approaches as they converge in much fewer iterations and are more robust relative to the parameter values, problem specifications, and different relaxation types. In comparison with the standard simplex approach, also, the proposed bundle method can find better lower bounds in a fraction of the time the simplex method needs. Further improvements on bundle methods with an application to the MCFND problem can be found in [40].

2.3.4.3 Lagrangian-Based Branch-and-Bound Methods

Holmberg and Yuan [53] proposed a Lagrangian-based branch-and-bound approach using the knapsack relaxation. They choose a subgradient method to solve the Lagrangian dual where, similar to [42], a restricted multicommodity flow problem is solved in the primal heuristic. The subset of arcs in the restricted flow problem is determined by either the branching information or the design solution of the Lagrangian subproblem. The best obtained upper bounds are used not only to guide the subgradient method, but also to cut branches in the branch-and-bound tree. To test the proposed solution method, they used 65 instances with up to 150 nodes, 1000 arcs, and 282 commodities. The results are compared with CPLEX. They show that the proposed solution methods can solve 9 instances to optimality and can find feasible solutions for all other instances, while CPLEX can solve 8 instances to optimality, is not able to find even a feasible solution for 13 instances, and cannot initialize the solution process for 9 instances.

Sellmann et al. [82] proposed a Lagrangian-based branch-and-cut approach using so-called minimum cardinality cuts and a number of variable fixing strategies. The two classical Lagrangian relaxations, flow and knapsack, are tested and compared. The results show that the knapsack relaxation is more effective with respect to the convergence of the subgradient algorithm. Although they used a simple subgradient method for the Lagrangian dual optimization, the results of the proposed exact branch-and-bound are competitive and encouraging in comparison with a state-of-the-art MIP solver. They also proposed a Lagrangian heuristic with different variable fixing approaches. The re-

sults of the proposed heuristic approach also outperform the other proposed heuristics in the literature. They used 48 benchmark instances with up to 24 nodes, 440 arcs, and 160 commodities. The results show that the first relaxation is preferable because of its performance when embedded into a subgradient method. The results also show that using cardinality cuts improves the total computational time.

Kliewer and Timajev [60] proposed a Lagrangian-based branch-and-cut algorithm that uses cover inequalities and *local cuts*. Local cuts are inspired from the classical reduced cost variable fixing procedure in a branch-and-bound tree. They show how to add the inequalities to the Lagrangian subproblem without destroying their known structure, knapsack or flow. The results show that the proposed algorithm outperforms many other methodologies in the literature. To test the proposed algorithm, different benchmark data instances with up to 30 nodes, 700 arcs, and 400 commodities are used. The results show that the proposed solution method outperforms CPLEX. The results also show that adding cuts can further improve the performance of the proposed solution method.

2.4 Discussion and Summary

In this thesis, among the variants of the network design problem, we focus on the two following important problems: 1) the multicommodity capacitated fixed-charge network design problem, MCFND, with interesting applications in transportation, telecommunications, logistics and production planning [64, 70], and 2) the multilayer network design problem, MLND, that recently showed interesting applications in the fields of transportation [17, 31, 88] and telecommunications [33, 61].

There are several solution methods that can be applied to these problems. Heuristics can obtain near-optimal solutions, especially for large instances, but the main drawback is how to assess the quality of obtained solutions. The lower bounds obtained by the relaxations and valid inequalities can help to determine the quality of the solutions obtained by heuristics.

Lagrangian-based algorithms are one of the most effective solution methods to solve

network design problems, in particular the MCFND (see [53] and [27] for example). The heuristic side of the algorithms produces the upper bound, while the relaxation side provides the lower bound to assess the quality of the solutions. The usual Lagrangian relaxations for the MCFND are the so-called flow and knapsack relaxations, which are obtained, respectively, by relaxing capacity constraints and flow conservation equations. For the first one, the commodity-based relaxation, the resulting Lagrangian subproblem decomposes into a collection of flow subproblems, one for each commodity; while the second one, the arc-based relaxation, allows solving the Lagrangian subproblem as a collection of continuous knapsack subproblems, one for each arc. The nodes of a network are the other entities that can be considered as decomposition components. Therefore, the first goal of this thesis is to present new relaxations decomposing the Lagrangian subproblem by node and to propose a new Lagrangian-based matheuristic to solve the problem.

There are many surveys on the MCFND in the literature [23, 64, 71]. In this chapter, we also reviewed the formulations and the solution methods proposed in the literature for the MCFND. For multilayer networks, however, to the best of our knowledge, the only survey is Kivelä et al. [59], which does not cover the multilayer network design problem and its applications, specifically in transportation. Therefore, the second objective of this thesis is to fill this gap by 1) proposing a new classification of multilayer network design problems, as well as a general formulation to facilitate the exposition of multilayer network design problems; and 2) synthesizing its applications in transportation and telecommunications, as well as the methods used to solve these problems.

Inspired by the existing effective Lagrangian-based heuristics for the single-layer MCFND, the third objective of the thesis is to propose a Lagrangian-based matheuristics for the MSMCFND. The challenge in multilayer network design problems, is how to handle the significant additional complexity incurred by the coupling constraints between the layers which complicate: 1) the task of developing Lagrangian relaxations that keep a balance between the quality of the lower bound and the computational efficiency of solving the Lagrangian subproblem; and 2) the derivation of effective feasible solutions. We propose a Lagrangian relaxation approach method, which not only provides

the lower bounds, but also guides a heuristic search to find the high quality upper bounds.

CHAPTER 3

NODE-BASED LAGRANGIAN RELAXATIONS FOR MULTICOMMODITY CAPACITATED FIXED-CHARGE NETWORK DESIGN

The usual Lagrangian relaxations for multicommodity capacitated fixed-charge network design are the so-called flow (or commodity-based) and knapsack (or arc-based) relaxations, where the resulting Lagrangian subproblems decompose by commodity and by arc, respectively. We present new node-based Lagrangian relaxations, where the resulting Lagrangian subproblem decomposes by node. We show that the Lagrangian dual bound improves upon the so-called strong linear programming bound, known to be equal to the Lagrangian dual bounds of the flow and knapsack relaxations. We also propose a Lagrangian-based matheuristic algorithm to obtain upper bounds. The results show that the proposed algorithm is competitive with the state-of-the-art heuristics in the literature on a large set of benchmark instances.

3.1 Introduction

In the multicommodity capacitated fixed-charge network design (MCFND) problem, several commodities such as goods, data or people, have to be routed between different origin-destination pairs of a given potential network. A predefined maximum flow (capacity) can be routed on each arc of the network. A network has to be designed by selecting appropriate links to route the commodities. A *fixed design cost* has to be paid to open an arc. In addition, a variable *flow cost* is imposed to route each unit of commodity demand on each arc. The problem is to find the minimum cost design and routing such that the demands are satisfied and the capacities are respected.

The problem is typically formulated as a Mixed-Integer Programming (MIP) model that includes two main sets of constraints: 1) *flow conservation equations* ensuring that the demands of the commodities are satisfied, and 2) *capacity constraints* ensuring that the flow on each arc does not exceed its capacity. The usual Lagrangian relaxations

for this formulation are the so-called flow (also known as shortest path) and knapsack relaxations, which are obtained, respectively, by relaxing capacity constraints and flow conservation equations. For the first one, the resulting Lagrangian subproblem decomposes into a collection of shortest path subproblems, one for each commodity; while the second one allows solving the Lagrangian subproblem as a collection of continuous knapsack subproblems, one for each arc. Therefore, the flow and knapsack relaxations are also known as *commodity-based* and *arc-based* relaxations, respectively.

The contribution of this paper is twofold. First, we present new node-based Lagrangian relaxations where the Lagrangian subproblem decomposes by node. We show that the Lagrangian dual bound improves upon the so-called strong linear programming (LP) bound, known to be equal to the Lagrangian dual bounds of the flow and knapsack relaxations. We evaluate the proposed relaxations on a set of benchmark instances. We compare the obtained bounds with the strong LP relaxation bounds. The average and maximum improvement of the best proposed node-based relaxation over the strong LP lower bounds are 1.7% and 20.5%, respectively. Second, inspired and motivated by the success of Lagrangian-based matheuristic approaches for the MCFND (see [53], [82], [27], and [60]), we present an effective algorithm based on the proposed node-based Lagrangian relaxations to solve the problem. We observe that the algorithm is competitive with the state-of-the-art heuristics in the literature.

The paper is organized as follows. Section 3.2 provides a review of the recent literature on the MCFND. Section 3.3 presents the formulation of the MCFND. In Section 3.4, we describe the proposed Lagrangian relaxations. Section 3.5 provides an overview about the subgradient and bundle methods which we use as non-differentiable optimization (NDO) solvers. In Section 5.3.2, we present a primal heuristic to be used in conjunction with the NDO solvers. Section 3.7 presents the proposed algorithm combining the NDO solvers and the primal heuristic. In Section 3.8, we present the experimental results and the analysis of the relaxations and the algorithm. In Section 3.9, we summarize this work, and we propose future research directions.

3.2 Literature Review

A number of efficient exact and heuristic solution methods have been proposed in the literature, mainly based on a combination of the following techniques: 1) mathematical programming, including Lagrangian relaxation, column generation, Benders decomposition [9], local branching [36] and Iterative Linear Programming, ILP, [84], 2) metaheuristics including tabu search [49], simulated annealing [58], evolutionary algorithms, path relinking and scatter search [50], and 3) heuristics including capacity scaling [55] and slope scaling [56]. In this section, we review the proposed solution methodologies in the literature with a focus on Lagrangian-based algorithms.

Gendron and Crainic [42] studied the classical flow and knapsack relaxations. The authors conclude that although the knapsack relaxation has nice practical and theoretical properties, more sophisticated procedures than the traditional subgradient method need to be developed to find better bounds. The flow relaxation, however, obtains the best performances with respect to the quality of the lower bounds. They also proposed a Lagrangian-based primal heuristic to find upper bounds. At each iteration of the subgradient procedure, a subset of arcs is obtained by applying specific rules on the solution of the Lagrangian subproblem. A restricted flow problem is then solved where only the subset of arcs exists. The best obtained upper bound is then used to guide the subgradient procedure.

Holmberg and Yuan [53] proposed a Lagrangian-based branch-and-bound algorithm for the MCFND problem using the knapsack relaxation. The scheme can be either an exact method or a heuristic when performed for a limited time. The authors chose a subgradient method to solve the Lagrangian dual where, similar to Gendron and Crainic [42], a restricted multicommodity flow problem is solved in the primal heuristic. The subset of arcs in the restricted flow problem is determined by either the branching information or the design solution of the Lagrangian subproblem. The best obtained upper bounds are used not only to guide the subgradient, but also to cut the branches of the branch-and-bound tree.

Crainic et al. [26] compared bundle and subgradient methods to optimize the Lagrangian dual of the two classical Lagrangian relaxations. The authors conclude that although bundle methods are more difficult to implement, they outperform subgradient approaches, as they converge in much fewer iterations and are more robust relative to the parameter values, problem specifications, and different relaxation types. In comparison with the standard simplex approach, also, the proposed bundle can find better lower bounds in a fraction of the time the simplex method needs. Further improvements on bundle method with an application to the MCFND problem can be found in Frangioni and Gorgone [40].

Sellmann et al. [82] proposed a Lagrangian-based branch-and-bound algorithm for the MCFND problem. The two traditional Lagrangian relaxations, shortest path and knapsack, are tested and compared. The results show that the knapsack relaxation is more effective with respect to the convergence of the subgradient algorithm. Although they used a simple subgradient method for the Lagrangian dual optimization, the results of the proposed exact branch-and-bound are competitive and encouraging in comparison with a state-of-the-art MIP solver. They also proposed a Lagrangian heuristic with different variable fixing techniques. The results of the proposed heuristic also outperformed the other heuristics reported in the literature.

Crainic et al. [27] presented a Lagrangian-based heuristic using a *slope scaling* scheme. The idea of slope scaling is to iteratively solve a linear multicommodity flow formulation, and to use the flow distribution to adjust the linear approximation at the next iteration. When the slope scaling method stalls, a perturbation move changes the initial linear approximation to start a new slope scaling procedure; for more details on slope scaling, see [27] and [56]. Once the number of performed slope scaling procedures reaches a predefined maximum number, a bundle-based process is launched to find a lower bound and new Lagrange multipliers. The Lagrange multipliers are then used to reinitialize the linearized costs and start another slope scaling procedure followed by a predefined number of perturbations.

Like Holmberg and Yuan [53] and Sellmann et al. [82], Kliewer and Timajev [60] proposed a Lagrangian-based branch-and-bound algorithm with additional valid inequal-

ities. The main contribution is how to apply the inequalities in the Lagrangian relaxation without destroying the known structure (knapsack or flow) of the subproblems. The results show that their proposed algorithm outperforms many other methodologies in the literature.

Beside the Lagrangian-based solution methods, other heuristic and exact methodologies are proposed in the literature. The heuristics include tabu search [24, 25, 28, 47], path relinking [48], scatter search [2, 78], capacity scaling [54, 55], simulated anealing [86], local branching [80], ILP [45] and a problem specific matheuristic proposed by Hewitt et al. [52]. The exact solution methods include Benders decomposition [20, 22], branch-and-cut [14], and branch-and-price-and-cut [43].

In summary, the Lagrangian relaxations proposed so far in the literature for the MCFND, to the best of our knowledge, are the traditional arc-based (knapsack) and commodity-based (flow) relaxations. The proposed Lagrangian heuristics, consequently, are also based on these relaxations. Therefore, the goal of this paper is to come up with new node-based Lagrangian relaxations and heuristics to find high quality lower and upper bounds.

3.3 Multicommodity Capacitated Fixed-charge Network Design Formulation

The MCFND is defined on a directed graph $G = (N, A)$, where N is the set of nodes and A is the set of arcs. For each node $i \in N$, we define the sets of forward and backward neighbours, N_i^+ and N_i^-, respectively. Each commodity $k \in K$ corresponds to an origin-destination pair such that d^k units of flow must travel between the origin $O(k)$ and the destination $D(k)$. The objective function to be minimized includes a cost $c_{ij}^k \geq 0$ for routing one unit of commodity $k \in K$ through arc $(i, j) \in A$ and a fixed cost $f_{ij} \geq 0$ for using arc $(i, j) \in A$, thus providing a capacity $u_{ij} > 0$ on the arc. A classical model for the MCFND introduces two sets of variables: x_{ij}^k is the flow of commodity $k \in K$ on arc $(i, j) \in A$, while y_{ij} is 1, if arc $(i, j) \in A$ is used, and 0, otherwise. The model is written as follows:

$$\min \sum_{(i,j)\in A} \sum_{k\in K} c_{ij}^k x_{ij}^k + \sum_{(i,j)\in A} f_{ij} y_{ij} \tag{3.1}$$

$$\sum_{j\in N_i^+} x_{ij}^k - \sum_{j\in N_i^-} x_{ji}^k = b_i^k \qquad \forall i \in N, \forall k \in K \tag{3.2}$$

$$\sum_{k\in K} x_{ij}^k \leq u_{ij} y_{ij} \qquad \forall (i,j) \in A \tag{3.3}$$

$$x_{ij}^k \leq s_{ij}^k y_{ij} \qquad \forall (i,j) \in A, \forall k \in K \tag{3.4}$$

$$x_{ij}^k \geq 0 \qquad \forall (i,j) \in A, \forall k \in K \tag{3.5}$$

$$y_{ij} \in \{0,1\} \qquad \forall (i,j) \in A \tag{3.6}$$

The objective function (3.1) is to minimize total routing and design costs. Constraints (3.2) are the usual *flow conservation* equations ensuring that the demands are routed from the origins to the destinations, where:

$$b_i^k = \begin{cases} +d^k, & \text{if } i = O(k) \\ -d^k, & \text{if } i = D(k) \\ 0, & \text{otherwise} \end{cases} \qquad \forall i \in N, \forall k \in K$$

Capacity constraints (3.3) ensure that the sum of flows on each arc $(i,j) \in A$ does not exceed its capacity u_{ij}. These are also known as *linking constraints* because they ensure that no flow is allowed on arc $(i,j) \in A$, unless it is open and its fixed cost is paid. Constraints (3.4) are *strong linking constraints*, where $s_{ij}^k = \min(d^k, u_{ij})$. Although constraints (3.4) are redundant (valid) for the MIP model, adding these inequalities significantly improves the lower bounds for the LP relaxation [42]. Therefore, the LP relaxation with the strong inequalities is called *strong LP relaxation*, while the LP relaxation without the strong linking constraints is said to be *weak LP relaxation*.

The so-called knapsack and flow relaxations are obtained by relaxing constraints (3.2) and (3.3)-(3.4), respectively. The Lagrangian subproblem of the knapsack relaxation decomposes by arc, i.e., a knapsack problem $\mathscr{P}_{Kn}^{(i,j)}$ for each arc $(i,j) \in A$. The

subproblem decomposes by commodity in the flow relaxation, i.e., a shortest path problem $\mathscr{P}^k_{Sh}$ for each commodity $k \in K$.

The model has to be slightly reformulated to come up with the new node-based relaxations. We introduce the following notation for each node $i \in N$:

— $K_i^O = \{k \in K \,|\, i = O(k)\}$, the commodities for which i is the origin;

— $K_i^D = \{k \in K \,|\, i = D(k)\}$, the commodities for which i is the destination;

— $K_i^T = \{k \in K \,|\, i \neq O(k), D(k)\}$, the commodities for which i is a transshipment node.

We also consider the following basic properties: 1) for each commodity $k \in K$, it is well-known that the flow conservation equation at $i = D(k)$ (or at $i = O(k)$) is redundant, and 2) because the costs are nonnegative, for each arc $(i, j) \in A$, $x_{ij}^k = 0$ if $k \in K_j^O$ (or $k \in K_i^D$). We then rewrite the flow conservation equations (3.2) as follows:

$$\sum_{j \in N_i^+} x_{ij}^k - \sum_{j \in N_i^-} x_{ji}^k = 0 \qquad \forall i \in N, \forall k \in K_i^T \tag{3.7}$$

$$\sum_{j \in N_i^+} x_{ij}^k = d^k \qquad \forall i \in N, \forall k \in K_i^O \tag{3.8}$$

$$x_{ij}^k = 0 \qquad \forall (i, j) \in A, \forall k \in K_j^O \cup K_i^D \tag{3.9}$$

Note that, since we rewrite the flow conservation constraints (3.2) at $i \in N, k \in K_i^O$ as constraints (3.8), constraints (3.9) must be added to the model. Otherwise, the model might be infeasible in some special cases. Suppose that in Figure 3.1, for example, a unit of demand has to be flown from node 1 to node 4. Then the routing in the cycle $(1, 2, 3, 1)$ is an optimal solution when we remove constraints (3.9) from the reformulation, while this solution is certainly not feasible for the original problem. Therefore, constraints (3.9) ensure that no flow comes back to the origin node in the reformulated flow conservation constraints.

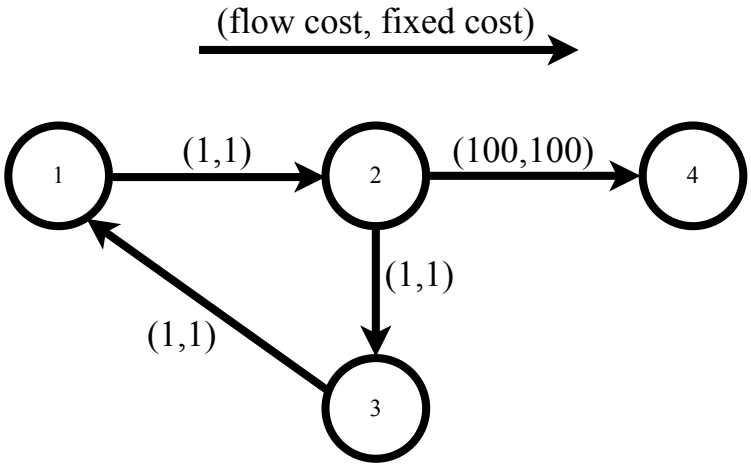

Figure 3.1 – Small network example.

3.4 Node-Based Lagrangian Relaxations

We now present three new node-based decomposition methods obtained by applying Lagrangian relaxation on three different reformulations. The first reformulation is obtained by replacing (3.2) with (3.7)-(3.9). In addition to this replacement, the two other reformulations use the Lagrangian decomposition technique [51] by introducing copies of design and flow variables, respectively denoted z_{ij}, $\forall (i,j) \in A$, and v^k_{ij}, $\forall (i,j) \in A$, $\forall k \in K$, which are defined with the following copy constraints:

$$z_{ij} - y_{ij} = 0 \qquad \forall (i,j) \in A \qquad (3.10)$$

$$v^k_{ij} - x^k_{ij} = 0 \qquad \forall (i,j) \in A, \forall k \in K \qquad (3.11)$$

The following redundant constraints are added to the two last reformulations to

strengthen the Lagrangian subproblems:

$$\sum_{j \in N_i^-} v_{ji}^k = d^k \qquad \forall i \in N, \forall k \in K_i^D \tag{3.12}$$

$$v_{ji}^k = 0, \qquad \forall (j,i) \in A, \forall k \in K_i^O \cup K_j^D \tag{3.13}$$

$$\sum_{k \in K} v_{ji}^k \leq u_{ji} z_{ji} \qquad \forall (j,i) \in A \tag{3.14}$$

$$v_{ji}^k \leq s_{ji}^k z_{ji} \qquad \forall (j,i) \in A, \forall k \in K \tag{3.15}$$

$$v_{ji}^k \geq 0 \qquad \forall (j,i) \in A, \forall k \in K \tag{3.16}$$

$$z_{ji} \in \{0,1\} \qquad \forall (j,i) \in A \tag{3.17}$$

The two last reformulations are based on replacing (3.2) with (3.7)-(3.9) and adding (3.10)-(3.17) to the formulation. In the third reformulation, the flow conservation equations (3.7) are also replaced by:

$$\sum_{j \in N_i^+} x_{ij}^k - \sum_{j \in N_i^-} v_{ji}^k = 0, \, i \in N, k \in K_i^T. \tag{3.18}$$

Since we associate no costs with the added variables, for any optimal solution to the original model, there exists a feasible solution to the reformulation with the same objective function value, and vice-versa. In the three following subsections, we present the proposed Lagrangian relaxations where constraints are relaxed in such a way that subproblems decompose by node.

3.4.1 First Reformulation-Lagrangian Relaxation

Recall that the first formulation consists of constraints (3.3)-(3.9). We relax constraints (3.7) in a Lagrangian way by introducing π_i^k, $\forall i \in N, \forall k \in K_i^T$, as the Lagrange multipliers of each of these constraints. The following valid inequalities are also added to improve the relaxation:

$$\sum_{j \in N_i^+} x_{ij}^k \leq g_i^k \qquad \forall i \in N, \forall k \in K_i^T \tag{3.19}$$

where $g_i^k = \min\{d^k, \sum_{j\in N_i^-} u_{ji}\}$, $\forall$i$\in N, \forall k \in K_i^T$. The resulting Lagrangian subproblem decomposes by node. The subproblem for each node $i \in N$ is then:

$$(\mathscr{P}_L^i) \quad Z_i^{xy}(\pi) = \min \sum_{j\in N_i^+} \left(\sum_{k\in K} c_{ij}^k(\pi) x_{ij}^k + f_{ij} y_{ij} \right) \tag{3.20}$$

$$\sum_{j\in N_i^+} x_{ij}^k = d^k \qquad \forall k \in K_i^O \tag{3.21}$$

$$\sum_{j\in N_i^+} x_{ij}^k \leq g_i^k \qquad \forall k \in K_i^T \tag{3.22}$$

$$x_{ij}^k = 0 \qquad \forall j \in N_i^+, \forall k \in K_i^D \cup K_j^O \tag{3.23}$$

$$\sum_{k\in K} x_{ij}^k \leq u_{ij} y_{ij} \qquad \forall j \in N_i^+ \tag{3.24}$$

$$x_{ij}^k \leq s_{ij}^k y_{ij} \qquad \forall j \in N_i^+, \forall k \in K \tag{3.25}$$

$$x_{ij}^k \geq 0 \qquad \forall j \in N_i^+, \forall k \in K \tag{3.26}$$

$$y_{ij} \in \{0,1\} \qquad \forall j \in N_i^+ \tag{3.27}$$

where

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

This node-based relaxation is called *location relaxation* because the resulting subproblem for each node $i \in N$ is a *Capacitated Facility Location Problem* (CFLP) $\mathscr{P}_L^i$. As illustrated in Figure 3.2, $K_i^O \cup K_i^T$ and N_i^+ are the sets of customers and facilities, respectively. Constraints (3.21) ensure that for each customer $k \in K_i^O$ the total supplies from all facilities in N_i^+ is equal to the demand of the customer d^k. For each customer $k \in K_i^T$, constraints (3.22) ensure the total flow from all the facilities is at most g_i^k. Constraints (3.22) prevent any flow to K_i^D and K_j^O customers. Inequalities (3.24) are the

linking constraints ensuring that there is no flow from a facility if it is close. Strong inequalities (3.25) have the same interpretation of the strong inequalities of the network design problem. Constraints (3.26) and (3.27) define the domain of the decision variables. In summary, the CFLP problem determines: 1) which facility at node $j \in N_i^+$ to open, and 2) what portion of each customer $k \in K_i^O \cup K_i^T$ demand is covered by which opened capacitated facility. A lower bound on the optimal value of the MCFND is then obtained as follows: $Z^1(\pi) = \sum_{i \in N} Z_i^{xy}(\pi)$. The best lower bound is obtained by solving the Lagrangian dual: $Z^1 = \max_\pi Z^1(\pi)$.

Proposition 3.1. *Let Z^{LP} be the strong LP relaxation lower bound then $Z^1 \geq Z^{LP}$.*

Proof. Since the CFLP does not have the integrality property, by solving the Lagrangian dual we obtain a lower bound that improves upon the strong LP relaxation lower bound. □

The *Dantzig-Wolfe* (DW) reformulation (primal form) of the Lagrangian dual is needed to implement the bundle method. To build this reformulation, let Q_i be the index set of the extreme points of the convex hull of the Lagrangian subproblem feasible domain for $i \in N$, i.e., $(x(q), y(q))_{q \in Q_i}, i \in N$, are these extreme points. Also let $\theta(q)$ be the variable representing the weight associated with the extreme point indexed by $q \in Q_i, i \in N$. The master problem of the DW reformulation of the Lagrangian dual is then written as:

$$Z^2 = \min \sum_{i \in N} \sum_{q \in Q_i} \theta(q) \left(\sum_{j \in N_i^+} \sum_{k \in K} c_{ij}^k x_{ij}^k(q) + \sum_{j \in N_i^+} f_{ij} y_{ij}(q) \right) \tag{3.28}$$

$$\sum_{j \in N_i^+} \sum_{q \in Q_i} \theta(q) x_{ij}^k(q) - \sum_{j \in N_i^-} \sum_{q \in Q_j} \theta(q) x_{ji}^k(q) = 0 \quad \forall i \in N, \forall k \in K_i^T(\pi_i^k) \tag{3.29}$$

$$\sum_{q \in Q_i} \theta(q) = 1 \quad \forall i \in N \tag{3.30}$$

$$\theta(q) \geq 0 \quad \forall i \in N, q \in Q_i \tag{3.31}$$

This is the disaggregated form of the master problem of the DW reformulation. There

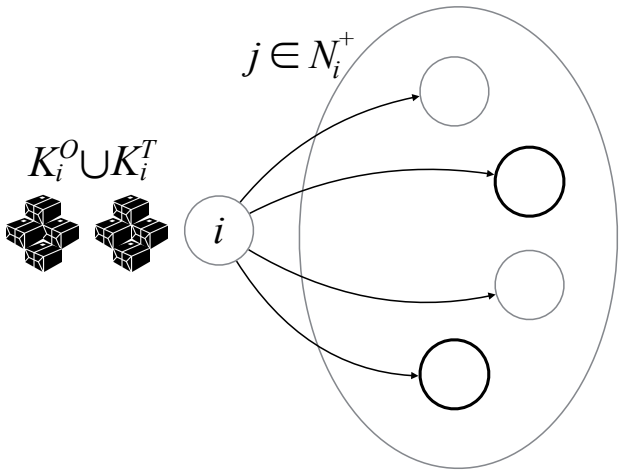

Figure 3.2 – Capacitated facility location subproblem of the location relaxation

is also the aggregated form where the Lagrangian subproblem is considered as a single model, rather than being decomposed by node. Then Q represents the index set of the extreme points of the convex hull of the Lagrangian subproblem feasible domain. In the experimental results, we consider both the disaggregated and the aggregated models.

3.4.2 Second Reformulation-Lagrangian Relaxation

As mentioned before, the second reformulation includes constraints (3.3)-(3.17). In the second relaxation, the copy constraints (3.10) and (3.11), as well as the flow conservation equations (3.7) are relaxed. In addition to (3.19), the following valid inequalities are also added to improve the relaxation:

$$\sum_{j \in N_i^-} v_{ji}^k \leq h_i^k, \forall i \in N, \forall k \in K_i^T, \tag{3.32}$$

where $h_i^k = \min\{d^k, \sum_{j \in N_i^+} u_{ij}\}$, $\forall i \in N, \forall k \in K_i^T$. The resulting Lagrangian subproblem decomposes not only by node but further, for each node, into two independent subproblems, one in variables (x, y) only, the other in variables (v, z) only.

As before, we denote by π_i^k, $\forall i \in N$, $\forall k \in K_i^T$, the Lagrange multipliers associated with the flow conservation equations (3.7). We also denote by γ_{ij}, $\forall (i,j) \in A$, and ω_{ij}^k, $\forall (i,j) \in A$, $\forall k \in K$, the Lagrange multipliers associated with (3.10) and (3.11), respectively.

The first subproblem, in variables (x,y), can be written as follows, for each node $i \in N$:

$$(\mathscr{P}_{LF}^i) \quad Z_i^{xy}(\gamma,\omega,\pi) = \min \sum_{j \in N_i^+} \left(\sum_{k \in K} c_{ij}^k(\omega,\pi) x_{ij}^k + f_{ij}(\gamma) y_{ij} \right) \tag{3.33}$$

$$(3.21) - (3.27)$$

where $f_{ij}(\gamma) = f_{ij} - \gamma_{ij}$, $\forall j \in N_i^+$, and

$$c_{ij}^k(\omega,\pi) = \begin{cases} c_{ij}^k - \omega_{ij}^k + \pi_i^k - \pi_j^k, & \text{if } k \in K_i^T \cap K_j^T, \\ c_{ij}^k - \omega_{ij}^k + \pi_i^k, & \text{if } k \in K_i^T \setminus K_j^T, \\ c_{ij}^k - \omega_{ij}^k - \pi_j^k, & \text{if } k \in K_j^T \setminus K_i^T, \\ c_{ij}^k - \omega_{ij}^k, & \text{if } k \in K_i^O \cap K_j^D, \end{cases} \quad \forall j \in N_i^+, \forall k \in K.$$

The second subproblem, in variables (v,z), can be written as follows, for each node $i \in N$:

$$(\mathscr{P}_{LB}^i) \quad Z_i^{vz}(\gamma,\omega) = \min \sum_{j \in N_i^-} \left(\sum_{k \in K} c_{ji}^k(\omega) v_{ji}^k + f_{ji}(\gamma) z_{ji} \right) \tag{3.34}$$

$$\sum_{j \in N_i^-} v_{ji}^k = d^k \qquad \forall k \in K_i^D \tag{3.35}$$

$$\sum_{j \in N_i^-} v_{ji}^k \leq h_i^k \qquad \forall k \in K_i^T \tag{3.36}$$

$$v_{ji}^k = 0 \qquad \forall j \in N_i^-, \forall k \in K_i^O \cup K_j^D \tag{3.37}$$

$$\sum_{k\in K} v^k_{ji} \leq u_{ji} z_{ji} \qquad \forall j \in N_i^- \qquad (3.38)$$

$$v^k_{ji} \leq s^k_{ji} z_{ji} \qquad \forall j \in N_i^-, \forall k \in K \qquad (3.39)$$

$$v^k_{ji} \geq 0 \qquad \forall j \in N_i^-, \forall k \in K \qquad (3.40)$$

$$z_{ji} \in \{0,1\} \qquad \forall j \in N_i^- \qquad (3.41)$$

where $f_{ji}(\gamma) = \gamma_{ji}, \forall j \in N_i^-$ and $c^k_{ji} = \omega^k_{ji}, \forall j \in N_i^-, \forall k \in K$.

The two subproblems of each node $i \in N$ are capacitated facility location problems. In the first ($\mathscr{P}^i_{LF}$) and second ($\mathscr{P}^i_{LB}$) subproblems, the “customers" correspond, respectively, to the sets $K^O_i \cup K^T_i$ and $K^D_i \cup K^T_i$, while the “facilities" correspond, respectively, to the sets N_i^+ (forward neighbours set) and N_i^- (backward neighbours set). This node-based relaxation is then called *forward-backward location relaxation*.

A lower bound on the optimal value of the MCFND is obtained as follows:

$$Z^2(\gamma,\omega,\pi) = \sum_{i\in N} (Z^{xy}_i(\gamma,\omega,\pi) + Z^{yz}_i(\gamma,\omega)).$$

To obtain the best possible lower bound, we have to solve the Lagrangian dual: $Z^2 = \max_{\gamma,\omega,\pi} Z^2(\gamma,\omega,\pi)$. Let R_i be the index set of the extreme points of the convex hull of the Lagrangian subproblem feasible domain for $i \in N$, i.e., $(x(r),y(r),v(r),z(r))_{r\in R_i}, i \in N$, are these extreme points. Also let $\theta(r)$ be the variable representing the weight associated with extreme point indexed by $r \in R_i, i \in N$. The master problem of the DW reformulation of the Lagrangian dual is then:

$$Z^2 = \min \sum_{i\in N} \sum_{r\in R_i} \theta(r) \left(\sum_{j\in N_i^+} \sum_{k\in K} c^k_{ij} x^k_{ij}(r) + \sum_{j\in N_i^+} f_{ij} y_{ij}(r) \right) \qquad (3.42)$$

$$\sum_{r\in R_j} \theta(r) z_{ij}(r) - \sum_{r\in R_i} \theta(r) y_{ij}(r) = 0 \quad \forall (i,j) \in A, (\gamma_{ij}) \qquad (3.43)$$

$$\sum_{r\in R_j} \theta(r) v^k_{ij}(r) - \sum_{r\in R_i} \theta(r) x^k_{ij}(r) = 0 \quad \forall (i,j) \in A, \forall k \in K, (\omega^k_{ij}) \qquad (3.44)$$

$$\sum_{j\in N_i^+}\sum_{q\in R_i}\theta(r)x_{ij}^k(r)-\sum_{j\in N_i^-}\sum_{r\in R_j}\theta(r)x_{ji}^k(r)=0 \quad \forall i\in N,\forall k\in K_i^T,(\pi_i^k) \tag{3.45}$$

$$\sum_{r\in R_i}\theta(r)=1 \quad \forall i\in N \tag{3.46}$$

$$\theta(r)\geq 0 \quad \forall i\in N, r\in R_i \tag{3.47}$$

Proposition 3.2. $Z^2\geq Z^1\geq Z^{LP}$.

Proof.

$$\begin{aligned}
Z^2 = & \max_{\gamma,\omega,\pi} Z^2(\gamma,\omega,\pi) = \\
& \max_{\gamma,\omega,\pi}\left\{\sum_{i\in N}\left(Z_i^{xy}(\gamma,\omega,\pi)+Z_i^{yz}(\gamma,\omega)\right)\right\}\geq \\
& \max_{\gamma=0,\omega=0,\pi}\left\{\sum_{i\in N}\left(Z_i^{xy}(\gamma,\omega,\pi)+Z_i^{yz}(\gamma,\omega)\right)\right\}= \\
& \max_{\pi}\left\{\sum_{i\in N}Z_i^{xy}(\pi)\right\}=Z^1
\end{aligned}$$

□

Proposition 3.2 states that, in comparison with the location relaxation, the forward-backward location relaxation produces better lower bounds. However, the number of subproblems are doubled in the forward-backward relaxation. In addition, the number of Lagrange multipliers is increased to $|A|+|A||K|+\sum_{i\in N}|K_i^T|$, while it is only $\sum_{i\in N}|K_i^T|$ in the location relaxation.

Aggregation schemes

One drawback of this reformulation-relaxation is the large number of copy constraints, in particular with respect to the flow variables, which introduce $|A||K|$ Lagrange multipliers. One could handle this issue by adding the copy constraints dynamically, only when they are violated by the current Lagrangian subproblem solution. However, it is very likely that a large number of copy constraints will be added this way, unless we add other constraints linking the flow variables to their copies. Another approach is to

replace the large number of copy constraints by some aggregated form, thus reducing the number of Lagrange multipliers. This comes at the cost of weakening the lower bound obtained from the Lagrangian dual. A third approach, which combines the two techniques, is to add *aggregated linking constraints* to the reformulation, while also adding the *disaggregated copy constraints* (3.11) dynamically. The constraints will be added every f^{add} iterations if the amount of violation is more than a tolerance ε^{add}.

There are many ways to define aggregated linking constraints. We seek aggregation schemes that reduce the number of Lagrange multipliers significantly, while at the same time, retaining the quality of the Lagrangian dual lower bound. To this aim, we enforce the following rule in choosing an appropriate aggregation scheme: *At optimality, the master problem gives two feasible multicommodity flow solutions (possibly the same), one expressed in terms of the original flow variables, the other in terms of the copy variables. Moreover, these two multicommodity flow solutions have the same cost.* With this rule in mind, we propose to add the following aggregated linking constraints to the reformulation:

$$\sum_{j \in N_i^+} x_{ij}^k - \sum_{j \in N_i^+} v_{ij}^k = 0 \qquad \forall i \in N, \forall k \in K_i^T \tag{3.48}$$

$$\sum_{j \in N_i^-} v_{ji}^k - \sum_{j \in N_i^-} x_{ji}^k = 0 \qquad \forall i \in N, \forall k \in K_i^T \tag{3.49}$$

$$\sum_{(j,i) \in A} \sum_{k \in K} c_{ji}^k v_{ji}^k - \sum_{(i,j) \in A} \sum_{k \in K} c_{ij}^k x_{ij}^k = 0 \tag{3.50}$$

These constraints ensure that the original flow variables and their copies both define feasible multicommodity flow solutions, even if all the disaggregated linking constraints (3.11) are relaxed. Equation (3.50) ensures that the original flow variables and their copies both have the same total routing cost. When this constraint is relaxed in a Lagrangian way, the Lagrange multiplier represents the relative weight given to the original variables and their copies in the objective function.

These constraints are then relaxed in a Lagrangian way, yielding the following subproblems for each node $i \in N$, where μ_i^k, ν_i^k, $i \in N$, $k \in K_i^T$, and δ are the Lagrange

multipliers associated with constraints (3.48), (3.49) and (3.50), respectively:

$$(\mathscr{P}^i_{LF,agg}) \quad Z^{xy}_i(\gamma,\omega,\pi,\delta,\mu,\nu) = \min \sum_{j\in N_i^+} \left(\sum_{k\in K} c^k_{ij}(\omega,\pi,\delta,\mu,\nu)x^k_{ij} + f_{ij}(\gamma)y_{ij} \right) \tag{3.51}$$

subject to (3.21)-(3.27), where

$$c^k_{ij}(\omega,\pi,\delta,\mu,\nu) = \begin{cases} (1-\delta)c^k_{ij} - \omega^k_{ij} + \pi^k_i - \pi^k_j + \mu^k_i - \nu^k_j, & \text{if } k \in K^T_i \cap K^T_j, \\ (1-\delta)c^k_{ij} - \omega^k_{ij} + \pi^k_i + \mu^k_i, & \text{if } k \in K^T_i \setminus K^T_j, \\ (1-\delta)c^k_{ij} - \omega^k_{ij} - \pi^k_j - \nu^k_j, & \text{if } k \in K^T_j \setminus K^T_i, \\ (1-\delta)c^k_{ij} - \omega^k_{ij}, & \text{if } k \in K^O_i \cap K^D_j, \end{cases} \quad \forall j \in N_i^+, \forall k \in K,$$

and

$$(\mathscr{P}^i_{LB,agg}) \quad Z^{vz}_i(\gamma,\omega,\delta,\mu,\nu) = \min \sum_{j\in N_i^-} \left(\sum_{k\in K} c^k_{ji}(\omega,\delta,\mu,\nu)v^k_{ji} + f_{ji}(\gamma)z_{ji} \right) \tag{3.52}$$

subject to (3.35)-(3.41), where

$$c^k_{ji}(\omega,\delta,\mu,\nu) = \begin{cases} \delta c^k_{ji} + \omega^k_{ji} - \mu^k_j + \nu^k_i, & \text{if } k \in K^T_i \cap K^T_j, \\ \delta c^k_{ji} + \omega^k_{ji} + \nu^k_i, & \text{if } k \in K^T_i \setminus K^T_j, \\ \delta c^k_{ji} + \omega^k_{ji} - \mu^k_j, & \text{if } k \in K^T_j \setminus K^T_i, \\ \delta c^k_{ji} + \omega^k_{ji}, & \text{if } k \in K^D_i \cap K^O_j, \end{cases} \quad \forall j \in N_i^-, \forall k \in K.$$

A lower bound on the MCFND problem is then:

$$Z^2(\gamma,\omega,\pi,\delta,\mu,\nu) = \sum_{i\in N} (Z^{xy}_i(\gamma,\omega,\pi,\delta,\mu,\nu) + Z^{vz}_i(\gamma,\omega,\delta,\mu,\nu)).$$

The best lower bound is obtained by the Lagrangian dual:

$$Z^2_{agg} = \max_{\gamma,\omega,\pi,\delta,\mu,\nu} Z^2(\gamma,\omega,\pi,\delta,\mu,\nu).$$

The corresponding master problem of the DW reformulation takes the following form:

$$Z^2_{agg} = \min \sum_{i\in N}\sum_{q\in Q_i} \theta(q) \left(\sum_{j\in N_i^+}\sum_{k\in K} c^k_{ij} x^k_{ij}(q) + \sum_{j\in N_i^+} f_{ij} y_{ij}(q) \right) \tag{3.53}$$

subject to (3.43)-(3.47), and

$$\sum_{j\in N_i^+}\sum_{q\in Q_i} \theta(q) x^k_{ij}(q) - \sum_{j\in N_i^+}\sum_{q\in Q_j} \theta(q) v^k_{ij}(q) = 0 \qquad i\in N, k\in K^T_i \quad (\mu^k_i) \tag{3.54}$$

$$\sum_{j\in N_i^-}\sum_{q\in Q_i} \theta(q) v^k_{ji}(q) - \sum_{j\in N_i^-}\sum_{q\in Q_j} \theta(q) x^k_{ji}(q) = 0 \qquad i\in N, k\in K^T_i \quad (\nu^k_i) \tag{3.55}$$

$$\sum_{q\in Q_i} \theta(q) \left(\sum_{(j,i)\in A}\sum_{k\in K} c^k_{ji} v^k_{ji}(q) - \sum_{(i,j)\in A}\sum_{k\in K} c^k_{ij} x^k_{ij}(q) \right) = 0 \quad (\delta) \tag{3.56}$$

3.4.3 Third Reformulation-Lagrangian Relaxation

Recall that the third reformulation consists of constraints (3.3)-(3.6) and (3.8)-(3.18). After applying Lagrangian relaxation on constraints (3.10) and (3.11), the Lagrangian subproblem decomposes by node, using the fact that A can be partitioned into forward sets, $A = \cup_{i\in N} N_i^+$, or into backward sets, $A = \cup_{i\in N} N_i^-$. If we denote by γ_{ij}, $(i,j)\in A$, and ω^k_{ij}, $(i,j)\in A$, $k\in K$, the Lagrange multipliers associated with constraints (3.10) and (3.11), respectively, the Lagrangian subproblem for each node $i\in N$ can be written as:

$$(\mathscr{P}^i_F) \quad Z_i(\gamma,\omega) = \min \sum_{j\in N_i^+} \left(\sum_{k\in K} c^k_{ij}(\omega) x^k_{ij} + f_{ij}(\gamma) y_{ij} \right) + \sum_{j\in N_i^-} \left(\sum_{k\in K} c^k_{ji}(\omega) v^k_{ji} + f_{ji}(\gamma) z_{ji} \right) \tag{3.57}$$

$$\sum_{j\in N_i^+} x^k_{ij} - \sum_{j\in N_i^-} v^k_{ji} = 0 \qquad \forall k\in K^T_i \tag{3.58}$$

$$\sum_{j\in N_i^+} x^k_{ij} = d^k \qquad \forall k\in K^O_i \tag{3.59}$$

$$x^k_{ij} = 0 \qquad \forall j\in N_i^+, \forall k\in K^D_i \cup K^O_j \tag{3.60}$$

$$\sum_{k\in K} x_{ij}^k \leq u_{ij} y_{ij} \qquad \forall j \in N_i^+ \qquad (3.61)$$

$$x_{ij}^k \leq d^k y_{ij} \qquad \forall j \in N_i^+, \forall k \in K \qquad (3.62)$$

$$x_{ij}^k \geq 0 \qquad \forall j \in N_i^+, \forall k \in K \qquad (3.63)$$

$$y_{ij} \in \{0,1\} \qquad \forall j \in N_i^+ \qquad (3.64)$$

$$\sum_{j\in N_i^-} v_{ji}^k = d^k \qquad \forall k \in K_i^D \qquad (3.65)$$

$$v_{ji}^k = 0 \qquad \forall j \in N_i^-, \forall k \in K_i^O \cup K_j^D \qquad (3.66)$$

$$\sum_{k\in K} v_{ji}^k \leq u_{ji} z_{ji} \qquad \forall j \in N_i^- \qquad (3.67)$$

$$v_{ji}^k \leq d^k z_{ji} \qquad \forall j \in N_i^-, \forall k \in K \qquad (3.68)$$

$$v_{ji}^k \geq 0 \qquad \forall j \in N_i^-, \forall k \in K \qquad (3.69)$$

$$z_{ji} \in \{0,1\} \qquad \forall j \in N_i^- \qquad (3.70)$$

where $f_{ij}(\gamma) = f_{ij} - \gamma_{ij}$, $j \in N_i^+$, $f_{ji}(\gamma) = \gamma_{ji}$, $j \in N_i^-$, $c_{ij}^k(\omega) = c_{ij}^k - \omega_{ij}^k$, $j \in N_i^+, k \in K$ and $c_{ji}^k(\omega) = \omega_{ji}^k$, $j \in N_i^-, k \in K$. This subproblem is a *multicommodity single-node fixed-charge network flow problem* $\mathscr{P}_F^i$, not studied before, to the best of our knowledge. We call this relaxation the *single-node flow relaxation*.

A lower bound on the optimal value of the MCFND is then obtained as follows: $Z^3(\gamma,\omega) = \sum_{i\in N} Z_i(\gamma,\omega)$. The best possible lower bound is obtained by solving the Lagrangian dual: $Z^3 = \max_{\gamma,\omega} Z^3(\gamma,\omega)$. Let P_i be the index set of the extreme points of the convex hull of the Lagrangian subproblem feasible domain for $i \in N$, i.e., $(x(p), y(p), v(p), z(p))_{p\in P_i}, i \in N$, are these extreme points. Also let $\theta(p)$ be the variable representing the weight associated with extreme point indexed by $p \in P_i, i \in N$. The master problem of the DW reformulation is then written as:

$$Z^3 = \min \sum_{i\in N} \sum_{p\in P_i} \theta(p) \left(\sum_{j\in N_i^+} \sum_{k\in K} c_{ij}^k x_{ij}^k(p) + \sum_{j\in N_i^+} f_{ij} y_{ij}(p) \right) \qquad (3.71)$$

$$\sum_{p\in P_j}\theta(p)z_{ij}(p)-\sum_{p\in P_i}\theta(p)y_{ij}(p)=0 \quad \forall(i,j)\in A \quad (\gamma_{ij}) \tag{3.72}$$

$$\sum_{p\in P_j}\theta(p)v^k_{ij}(p)-\sum_{p\in P_i}\theta(p)x^k_{ij}(p)=0 \quad \forall(i,j)\in A,\forall k\in K \quad (\omega^k_{ij}) \tag{3.73}$$

$$\sum_{p\in P_i}\theta(p)=1 \quad \forall i\in N \tag{3.74}$$

$$\theta(p)\geq 0 \quad \forall i\in N,\forall p\in P_i \tag{3.75}$$

Proposition 3.3. $Z^3\geq Z^2\geq Z^1\geq Z^{LP}$.

Proof. We denote by S^x the set of flow vector x that satisfy flow conservation equations (3.7). For each node $i\in N$, we introduce the sets S_i^{vx}, S_i^{xy} and S_i^{vz} that correspond, respectively, to the solutions that satisfy constraints (3.58), (3.59)-(3.64) and (3.65)-(3.70). Using this notation, the convex hull of the Lagrangian subproblem feasible domain for $i\in N$ can be written in a compact way as follows:

$$\begin{aligned}
Z^3 = &\max_{\gamma,\omega} Z^3(\gamma,\omega) = \\
&\max_{\gamma,\omega}\left\{\sum_{i\in N} Z_i(\gamma,\omega)\right\} = \\
&\min\left\{cx+fy | y=z, x=v, (x,y,v,z)\in \cap_{i\in N}\text{conv}(S_i^{vx}\cap S_i^{xy}\cap S_i^{vz})\right\}\geq \\
&\min\left\{cx+fy | y=z, x=v, (x,y,v,z)\in \cap_{i\in N}\left(\text{conv}(S_i^{vx})\cap\text{conv}(S_i^{xy}\cap S_i^{vz})\right)\right\} = \\
&\min\left\{cx+fy | y=z, x=v, (x,y,v,z)\in \cap_{i\in N}\left(S_i^{vx}\cap\text{conv}(S_i^{xy}\cap S_i^{vz})\right)\right\} = \\
&\min\left\{cx+fy | y=z, x=v, (x,v)\in \cap_{i\in N}S_i^{vx}, (x,y,v,z)\in \cap_{i\in N}\text{conv}(S_i^{xy}\cap S_i^{vz})\right\} = \\
&\min\left\{cx+fy | y=z, x=v, x\in S^x, (x,y)\in \cap_{i\in N}\text{conv}(S_i^{xy}), (v,z)\in \cap_{i\in N}\text{conv}(S_i^{vz})\right\} = \\
&\max_{\gamma,\omega,\pi}\left\{\sum_{i\in N}\left(Z_i^{xy}(\gamma,\omega,\pi)+Z_i^{vz}(\gamma,\omega)\right)\right\} = \\
&\max_{\gamma,\omega,\pi} Z^2(\gamma,\omega,\pi) = Z^2
\end{aligned}$$

□

Proposition 3.3 states that in comparison with the forward-backward location re-

laxation, the single-node flow relaxation produces better lower bound. However, the corresponding subproblem of the single-node flow relaxation is more difficult to solve as it does not decompose by forward and backward subproblems.

Aggregation schemes

In order to tackle the issue of the large number of Lagrange multipliers involved in the copy constraints of the flow variables, we proceed as in the forward-backward location relaxation approach. These constraints are relaxed and gradually added to the master problem. The relaxation is also strengthened by the addition of constraints that satisfy the same rule as above: *At optimality, the master problem gives two feasible multicommodity flow solutions having the same cost, one expressed in terms of the original flow variables, the other in terms of the copy variables.* Therefore, the same constraints, (3.48)-(3.50), as the forward-backward location relaxation are added.

If we denote by μ_i^k, ν_i^k, $i \in N$, $k \in K_i^T$, and δ, the Lagrange multipliers associated with constraints (3.48), (3.49) and (3.50), respectively, the Lagrangian subproblem for each node $i \in N$ can be written as:

$$
\begin{aligned}
(\mathscr{P}^i_{F,agg}) \quad Z_i(\gamma,\omega,\mu,\nu,\delta) = \min & \sum_{j\in N_i^+}\left(\sum_{k\in K} c_{ij}^k(\omega,\mu,\nu,\delta)x_{ij}^k + f_{ij}(\gamma)y_{ij}\right) \\
& + \sum_{j\in N_i^-}\left(\sum_{k\in K} c_{ji}^k(\omega,\mu,\nu,\delta)v_{ji}^k + f_{ji}(\gamma)z_{ji}\right)
\end{aligned}
\tag{3.76}
$$

subject to (3.58)-(3.70) where

$$
c_{ij}^k(\omega,\mu,\nu,\delta) = \begin{cases}
(1-\delta)c_{ij}^k - \omega_{ij}^k + \mu_i^k - \nu_j^k, & \text{if } k \in K_i^T \cap K_j^T, \\
(1-\delta)c_{ij}^k - \omega_{ij}^k + \mu_i^k, & \text{if } k \in K_i^T \setminus K_j^T, \\
(1-\delta)c_{ij}^k - \omega_{ij}^k - \nu_j^k, & \text{if } k \in K_j^T \setminus K_i^T, \\
(1-\delta)c_{ij}^k - \omega_{ij}^k, & \text{if } k \in K_i^O \cap K_j^D,
\end{cases} \quad \forall j \in N_i^+, \forall k \in K,
$$

$$c_{ji}^k(\omega,\mu,\nu,\delta) = \begin{cases} \delta c_{ji}^k + \omega_{ji}^k - \mu_j^k + \nu_i^k, & \text{if } k \in K_i^T \cap K_j^T, \\ \delta c_{ji}^k + \omega_{ji}^k + \nu_i^k, & \text{if } k \in K_i^T \setminus K_j^T, \\ \delta c_{ji}^k + \omega_{ji}^k - \mu_j^k, & \text{if } k \in K_j^T \setminus K_i^T, \\ \delta c_{ji}^k + \omega_{ji}^k, & \text{if } k \in K_i^D \cap K_j^O, \end{cases} \quad \forall j \in N_i^-, \forall k \in K.$$

Given a set of multipliers, $Z^3(\gamma,\omega,\mu,\nu,\delta) = \sum_{i\in N} Z_i(\gamma,\omega,\mu,\nu,\delta)$ is a lower bound on the objective value of the MCFND problem. The value of the Lagrangian dual, $Z^3_{agg} = \max_{\gamma,\omega,\mu,\nu,\delta} Z^3(\gamma,\omega,\mu,\nu,\delta)$, then gives the best possible lower bound. Let P_i be the index set of the extreme points of the convex hull of the Lagrangian subproblem feasible domain for $i \in N$, i.e., $(x(p),y(p),v(p),z(p))_{p\in P_i}, i \in N$, are these extreme points. Also let $\theta(p)$ be the variable representing the weight associated with extreme point indexed by $p \in P_i, i \in N$. The master of the corresponding DW reformulation problem is then written as:

$$Z^3_{agg} = \min \sum_{i\in N} \sum_{p\in P_i} \theta(p) \left(\sum_{j\in N_i^+} \sum_{k\in K} c_{ij}^k x_{ij}^k(p) + \sum_{j\in N_i^+} f_{ij} y_{ij}(p) \right) \tag{3.77}$$

subject to (3.72)-(3.75) and

$$\sum_{j\in N_i^+} \sum_{p\in P_i} \theta(p) x_{ij}^k(p) - \sum_{j\in N_i^+} \sum_{p\in P_j} \theta(p) v_{ij}^k(p) = 0, \forall k \in K_i^T \quad (\mu_i^k) \tag{3.78}$$

$$\sum_{j\in N_i^-} \sum_{p\in P_i} \theta(p) v_{ji}^k(p) - \sum_{j\in N_i^-} \sum_{p\in P_j} \theta(p) x_{ji}^k(p) = 0, \forall k \in K_i^T \quad (\nu_i^k) \tag{3.79}$$

$$\sum_{p\in P_i} \theta(p) \left(\sum_{(j,i)\in A} \sum_{k\in K} c_{ji}^k v_{ji}^k(p) - \sum_{(i,j)\in A} \sum_{k\in K} c_{ij}^k x_{ij}^k(p) \right) = 0 \quad (\delta) \tag{3.80}$$

3.5 Solving Lagrangian Dual

Given a particular Lagrangian relaxation, let α and $Z(\alpha)$ be the corresponding Lagrange multipliers and the objective function of the Lagrangian subproblem, respectively. The Lagrangian dual, $Z^* = \max_\alpha Z(\alpha)$, should then be solved to find the best possible lower bound. The Lagrangian function $Z(\alpha)$ is concave but unfortunately non-

differentiable. However, it is easy to compute a subgradient direction for any choice of the Lagrange multipliers. To solve the Lagrangian dual problems of the proposed relaxations, we consider two different non-differentiable optimization methods: subgradient and bundle.

3.5.1 Subgradient Method

A subgradient method includes moving from a current point α_i to a new point α_{i+1} using a step size s_i and a direction d_i in the following simple formula:

$$\alpha_{i+1} = \alpha_i + s_i d_i \tag{3.81}$$

In a simple subgradient method, d_i is equal to the current subgradient g_i computed as the violation of the relaxed constraints when the variables are fixed to the values found by solving the Lagrangian subproblem at the current point α_i. A simple approach to compute the step size is $s_i = \mu_i \left(Z_e^* - Z(\alpha_i)\right) / g_i d_i$ where Z_e^* and μ_i are an estimation for the value of Z^* and a scaling parameter, respectively.

We are using a unified subgradient-type algorithm described in Frangioni et al. [41], where more sophisticated strategies are used for computing both the step size and the direction. The direction, for example, is computed as $d_i = \beta_i g_i + (1-\beta_i) d_{i-1}$, where β_i is the so-called deflection coefficient. In this approach, α_i is not necessarily the point of the current iteration. It can remain unchanged if an ascent direction occurs. For details on the different strategies to compute the step size and the deflection coefficient see [41].

3.5.2 Bundle Method

The idea of the bundle method is to compute the direction using a bundle B of the subgradients found so far. The direction d_i with respect to the current point α_i is found by solving the following quadratic problem:

$$\min_{\theta} \left\{ \frac{1}{2} t \| \sum_{b \in B} g_b \theta_b \|^2 + \sum_{b \in B} E_b \theta_b; \quad s.t. \quad \sum_{b \in B} \theta_b = 1, \theta \geq 0 \right\} \tag{3.82}$$

where for each bundle member $b \in B$, g_b and $t > 0$ are the corresponding subgradient vector and the so-called trust region parameter, respectively. $E_b = Z(\alpha_b) + g_b(\alpha_i - \alpha_b) - Z(\alpha_i)$ is the current linearization error. The ascent direction is then computed using the convex combination of the subgradients defined by θ. Valuable information can be extracted form the solution of θ to produce integer feasible solutions (see Section 3.7). The quadratic problem in bundle methods is the Augmented Method of the standard DW approach, with a linear Lagrangian term related to the current point (stability center) and a quadratic term related to the stability parameter ($t > 0$).

Let ε^{Lin} be the relative precision required, then a general stopping condition is when the bundle finds an epsilon-subgradient g such that $t^*||g|| \leq \varepsilon^{Lin}|Z_e^*|$, where t^* is a parameter that estimates the longest step that can be performed and $||g||$ is a norm-like function. Different estimates can be used for Z_e^*, although using the best lower bound found so far is pretty common. A detailed explanation of the bundle method can be found in [39] and [26].

3.6 Primal Heuristic

To obtain upper bounds, we develop a Lagrangian-based primal heuristic using a slope scaling procedure. Given a design solution vector $\hat{y}$, the following multicommodity capacitated network flow (MCNF) problem is solved at each iteration of the procedure:

$$\min \sum_{(i,j)\in A} \sum_{k\in K} \overline{c}_{ij}^k x_{ij}^k \tag{3.83}$$

$$\text{s.t.} \quad (3.2),(3.5) \quad \text{and}$$

$$\sum_{k\in K} x_{ij}^k \leq u_{ij}, \forall (i,j) \in A \tag{3.84}$$

where the initial linearized cost $\overline{c}_{ij}^k$, $\forall (i,j) \in A$, $\forall k \in K$, is computed as follows:

$$\overline{c}_{ij}^k = (c_{ij}^k + \rho_{ij})(1 + M(1 - \hat{y}_{ij})) \tag{3.85}$$

where, $\rho_{ij} = f_{ij}/u_{ij}$ and M is a large positive value to avoid routing on the closed arcs of $\hat{y}$. When $\hat{y}_{ij} = 1$, $(i,j) \in A$, the cost of the corresponding arc is equal to the cost in the LP relaxation of the model obtained from the MCFND by removing constraints (3.4). If $\hat{y}_{ij} = 0$, $(i,j) \in A$, then the cost is set to a large positive value to avoid any flow on the corresponding arc. Note that, even if $\hat{y}_{ij}$ is fractional, the formula (3.85) can be used: arcs with small fractional values still have large costs.

After solving the MCNF problem, a flow solution $\bar{x}$ is obtained and the linearized costs are updated using the following equation to trigger another iteration:

$$\bar{c}^k_{ij} = \begin{cases} c^k_{ij} + \rho_{ij} & \text{if } \bar{x}^k_{ij} > 0 \\ \bar{c}^k_{ij} & \text{if } \bar{x}_{ij} = 0 \end{cases} \tag{3.86}$$

where, $\rho_{ij} = f_{ij} / \sum_{k \in K} \bar{x}^k_{ij}$, and $\bar{x}^k_{ij}, \forall (i,j) \in A, \forall K \in K$ is the flow solution obtained from solving the current MCNF problem. When $\bar{x}^k_{ij}$ is positive, this equation ensures that, at the next iteration, if the flow solution remains the same, the cost on each arc is equal to the fixed design cost plus the sum of the flow variable costs. If $\bar{x}^k_{ij} = 0$, we keep the same cost because it is large enough to avoid any flow.

At each slope scaling iteration, the algorithm produces a feasible solution for the original problem. The flow routing solution of the feasible solution is equal to $\bar{x}$, while its design solution $\bar{y}$ is obtained using the following simple rule:

$$\bar{y}_{ij} = \begin{cases} 1 & \text{if } \sum_{k \in K} \bar{x}^k_{ij} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.87}$$

Let z_t and z_{t-1} be the current and the previous objective values of the MCNF problem at each iteration t of the slope scaling procedure. Then the slope scaling procedure converges when $z_t = z_{t-1}$. Therefore, the termination condition of the slope scaling procedure is to reach either the same objective value for two successive iterations or a predefined maximum number of slope scaling iterations it^{slope} has been attained.

At the end of the slope scaling procedure, an intensification step is performed using a set of n^{intens} best feasible solutions selected during the process. For each chosen feasible

solution, an MCNF problem is constructed by 1) fixing the design variable vector to the corresponding design values of the solution, and 2) setting the costs to the original flow variable costs. The MCNF problem is then solved in the hope of obtaining a better flow distribution. The primal heuristic phase is illustrated in Algorithm 3.

The proposed primal heuristic is applied in conjunction with the subgradient or bundle methods described in Section 3.5. When a new upper bound is found in the whole procedure, the best upper bound found so far, UB^{best}, is updated if the solution is better than the current UB^{best}. We also define a set of feasible solutions found so far, F, with a limited size (set to 1000) which is also updated whenever a new solution is found. If the set is full and the solution found is better than the worst solution of F, then the worst solution is removed from F and the new found solution is added to the set. The set F is a long-term memory that is used for further improvements, as it is explained in the next section.

Algorithm 1: Primal heuristic procedure

```
Input design solution → ŷ
Initialize the linearized costs of MCNF problem using ŷ and formula (3.85)
Solve MCNF problem → ȳ, x̄, z_t
Compute upper bound
Update UB^best and feasible solution set F
repeat
    Update objective function of MCNF problem using x̄ and formula (3.86)
    Solve MCNF problem → ȳ, x̄, z_t, z_{t-1}
    Compute upper bound
    Update UB^best and feasible solution set F
until the number of iterations is greater than or equal to it^slope or z_t = z_{t-1}
Perform intensification
```

3.7 The Proposed Algorithm

The proposed Lagrangian-based matheuristic algorithm consists of three main phases: 1) **initialization**, where the weak or strong LP relaxation of the original problem, $\mathscr{P}_{LP}$, is solved in order to initialize the Lagrange multipliers of the proposed Lagrangian relaxation, 2) **Lagrangian**, including the bundle (or subgradient) procedure to find lower

bounds and the slope scaling procedure to obtain upper bounds, and 3) **post-optimization**, where a restricted MIP problem $\mathscr{P}_{PO}$ is solved to improve the upper bounds.

Different information can be extracted from different relaxations to run the slope scaling procedure. Therefore, the proposed algorithm is designed in such a way to not only support all the proposed node-based relaxations but also exploit as much as possible the provided information of each particular relaxation. In addition, the algorithm supports the traditional Lagrangian relaxations of the MCFND, the knapsack and flow relaxations, to compare their results with those of the proposed node-based relaxations. In the following subsections, we describe the three phases of the algorithm. The last subsection summarizes the algorithm and provides a pseudocode for the whole algorithm.

3.7.1 Initialization

The first step in the initialization phase is to solve $\mathscr{P}_{LP}$ that helps to have better initial Lagrange multipliers and better initial lower bound. We test both the weak or strong LP relaxation. We define a switch, s^{LP}, as a parameter in the algorithm to test the two cases. The Lagrange multipliers are then initialized to the corresponding dual values of the relaxed constraints. At the end of the initialization phase, the slope scaling procedure is called using the design solution of $\mathscr{P}_{LP}$, $\hat{y}_{lp}$. This slope scaling step provides an initial upper bound to be used in the bundle (or subgradient) procedure.

3.7.2 Lagrangian Phase

Here we explain the Lagrangian phase including 1) a bundle (or subgradient) procedure to find lower bounds, and 2) a slope-scaling-based primal heuristic procedure to be performed in conjunction with the bundle (or subgradient) procedure.

Bundle (or subgradient)

We use bundle (or subgradient) method, for it^{Lag} maximum number of iterations and in t^{Lag} time limit, to solve the Lagrangian dual. At each iteration of the bundle (or subgradient) procedure, the Lagrangian subproblem of the corresponding relaxation has to

be solved in order to obtain a lower bound. We use a MIP solver to solve the subproblems of the proposed node-based Lagrangian relaxations. To improve the performance of the MIP solver on the resolution of the CFLPs ($\mathscr{P}_L^i, \forall i \in N$) of the location relaxation, and the forward CFLPs ($\mathscr{P}_{LF}^i, \forall i \in N$) of the forward-backward location relaxation, the following knapsack cut is added to the CFLP problem:

$$\sum_{j \in N_i^+} u_{ij} y_{ij} \geq \sum_{k \in K_i^O} d^k \tag{3.88}$$

In the same way, the following knapsack cut is added to the backward CFLPs ($\mathscr{P}_{LB}^i$, $\forall i \in N$) of the forward-backward location relaxation:

$$\sum_{j \in N_i^-} u_{ji} z_{ji} \geq \sum_{k \in K_i^D} d^k \tag{3.89}$$

Both of these inequalities are also added to the third proposed relaxation to improve the performance of the MIP solver.

It is possible to improve more the performance of the MIP solver by adding a preprocessing procedure on the flow variables before starting the resolution of the CFLPs. The preprocessing includes the following rules, $\forall i \in N$:

— In $\mathscr{P}_L^i$: $x_{ij}^k = 0, \forall j \in N_i^+, \forall k \in K \setminus K_i^O$, if $c_{ij}^k(\pi) \geq 0$.

— In $\mathscr{P}_{LF}^i$: $x_{ij}^k = 0, \forall j \in N_i^+, \forall k \in K \setminus K_i^O$, if $c_{ij}^k(\omega, \pi) \geq 0$ and $f_{ij}(\gamma) \geq 0$.

— In $\mathscr{P}_{LF,agg}^i$: $x_{ij}^k = 0, \forall j \in N_i^+, \forall k \in K \setminus K_i^O$, if $c_{ij}^k(\omega, \pi, \delta) \geq 0$ and $f_{ij}(\gamma) \geq 0$.

— In $\mathscr{P}_{LB}^i$: $v_{ji}^k = 0, \forall j \in N_i^-, \forall k \in K \setminus K_i^D$, if $c_{ji}^k(\omega) \geq 0$ and $f_{ji}(\gamma) \geq 0$.

— In $\mathscr{P}_{LB,agg}^i$: $v_{ji}^k = 0, \forall j \in N_i^-, \forall k \in K \setminus K_i^D$, if $c_{ji}^k(\omega, \lambda, \delta) \geq 0$ and $f_{ji}(\gamma) \geq 0$.

Even with the above enhancements, solving the MIP subproblems might be costly with respect to the computational time. To accelerate the solution of the MIP subproblems, we use the two following approaches: 1) solving the MIP problem to optimality gap ε_{gap}, and 2) limiting the number of nodes of the branch-and-bound tree to a parameter $node_{lim}$. To define ε_{gap}, we use one of the internal procedures of the bundle method, which approximately solves the master problem. In this procedure, the relative precision required to solve the master problem is initially set to $\varepsilon_{init} = 10^{-2}$, and decreased down

to $\varepsilon_{final} = 10^{-6}$ by multiplying it by $\mu_{eps} = 0.95$ every $s_{eps} = 5$ steps. We use this precision to define ε_{gap}. $node_{lim}$ is initially set to 10 and increased by multiplying it by 10 each time the precision is changed.

Since we are solving a MIP model in each of the proposed node-based relaxations, rather than seeking to find the optimal solution of the Lagrangian subproblem, there is a possibility to introduce a number of feasible design solutions P to the bundle method to build a new subgradient vector. Parameter pop^{max} is the maximum number of the populated solutions to be introduced to the bundle method.

In case of the traditional Lagrangian relaxations, the knapsack subproblems $\mathscr{P}^a_{Kn}, \forall a \in A$, and the shortest path subproblems $\mathscr{P}^k_{Sh}, \forall k \in K$ have to be solved, respectively, in the arc-based and commodity-based relaxations.

Primal Heuristic

At each iteration of the bundle (or subgradient) procedure, the described slope scaling procedure is started if the obtained design solution is new and either the frequency of the total number of bundle (or subgradient) iterations is equal to a predefined number f^{prim} or the lower bound has improved significantly since the last time the primal heuristic was called. The improvement is considered to be significant if $(LB_c - LB_l)/LB_l > \delta^{dual}$, where δ^{dual} is a parameter and LB_c and LB_l are, respectively, the lower bound computed at the current iteration and the lower bound obtained the last time primal heuristic was triggered.

For the location and single-node flow relaxations, the design solutions of the corresponding subproblems $\hat{y}_L$ and $\hat{y}_F$ are used, respectively, to run the slope scaling procedure. For the forward-backward location and single-node flow relaxations, however, it is possible to not only use the forward design solution $\hat{y}_{LF}$ but also the backward design solution $\hat{y}_{LB}$ obtained, respectively, from the forward and backward subproblems. For the knapsack and flow relaxations, respectively, design solutions $\hat{y}_{Kn}$ and $\hat{y}_{Sh}$ are available to start the slope scaling procedure.

Recall that set P is the populated feasible design solutions to be given to the bundle method. We use this set to launch different slope scaling procedures considering param-

eter pop^{max} as the maximum number of populated solutions to be given as input to the slope scaling procedure. The design solution $p \in P$ for the location, forward-backward location, and single-node flow relaxations are, respectively, $\hat{y}^p_L$, $\hat{y}^p_{LF}$ (or $\hat{y}^p_{LB}$) and $\hat{y}^p_F$. Extracting the populated solutions, however, is not possible for the knapsack and flow relaxations, because their subproblems provide a single solution.

At each iteration of the bundle procedure, a convexified design solution can be generated using the convex combination of the previously generated design solutions defined by θ, the variables of the bundle quadratic problem, as follows:

$$y^{conv}_{ij} = \sum_{b \in B} \theta_b y^b_{ij}, \quad \forall (i,j) \in A \tag{3.90}$$

where y^{conv}_{ij} and y^b_{ij} are the convexified design solution and the design solution of the bundle member $b \in B$ of arc $(i,j) \in A$. The obtained convexified design solution $\hat{y}_{conv}$ is used to trigger another slope scaling step. We define a switch, s^{conv}, as a parameter to decide whether or not to use the convexified design solution.

At the end of each primal heuristic procedure, in addition to an intensification step described in Section 5.3.2, a diversification step is performed if the algorithm is not able to improve the upper bound for a predefined maximum number of primal heuristic steps, st^{prim}. A primal heuristic step is a complete slope scaling procedure with intensification. The idea of diversification is to avoid selecting the links that frequently appeared in the obtained feasible solutions so far. The design solution of the best upper bound found so far is considered to initialize the linearized cost of formula (3.85) where a large linearized cost is assigned not only to the closed arcs, but also to a set of arcs (set to 10) selected randomly from the most frequently opened ones in a percentage (set to 90%) of the set of feasible solutions, F. Then, a new slope scaling procedure is triggered using this new initial linearized costs. The selected arcs are marked to avoid choosing them again in the next diversification step. Therefore, at each diversification step, a new set of the most frequent arcs are considered to be fixed to a large cost. We restrict the total number of slope scaling steps in each diversification procedure and the total number of diversification steps to it^{div} and st^{div}, respectively.

At the end of the bundle (or subgradient) procedure, a slope scaling step is called using the design solution of the best obtained lower bound $\hat{y}_{best}$ in the hope of finding a better upper bound.

Termination Conditions

Three different termination conditions are considered to stop the whole procedure including the bundle (or subgradient) procedure and the primal heuristic steps: 1) if the elapsed time reaches a predefined time limit t^{Lag}, 2) if the value of the subgradient norm is less than ε, meaning that the bundle (or subgradient) converged, and 3) if the total number of the bundle (or subgradient) iterations reaches a predefined number it^{Lag}.

3.7.3 Post-Optimization Phase

At the end of the algorithm, a *post-optimization* phase is performed by solving a restricted MIP problem $\mathscr{P}_{PO}$ for a limited time $t^{PostOpt}$. To build the restricted MIP, we use a number of elite solutions, the feasible solutions selected from the long term memory F, with less than $\delta_p\%$ gap from the best known feasible solution. The design variables that are closed in all the elite solutions are fixed to 0. A simple iterative procedure is performed to set $\delta_p\%$ in such a way that $f\%$ (set to fall in the interval $[y_0^{min}, y_0^{max}]$) of the design variables are set to 0 in the restricted MIP. At the beginning of the procedure, all the solutions of F are selected as elite. If the percentage of the fixed variables with these elite solutions fall in the interval $[y_0^{min}, y_0^{max}]$, then the desired percentage is obtained; otherwise, a new set of elite solutions is built by removing the worst solution of F until the percentage falls into the desired range. If the percentage jumps over the range, the procedure back tracks to the last set of elite solutions.

3.7.4 Summary

The algorithm starts with an initialization phase where the dual values of $\mathscr{P}_{LP}$ are used to initialize the Lagrangian multipliers. At the end of initialization phase, a slope scaling procedure is triggered using the final design solution of the LP relaxation $\mathscr{P}_{LP}$,

$\hat{y}_{lp}$.

In the Lagrangian phase, we use a bundle (or subgradient) method to solve the Lagrangian dual. Using the obtained design solution of the Lagrangian subproblem, the slope scaling heuristic, which is empowered by the intensification and diversification procedures, finds feasible solutions. At the end of the bundle (or subgradient) procedure, another slope scaling step is performed using the design solution of the best known lower bound $\hat{y}_{best}$. At the end of the algorithm, the post-optimization phase improves the upper bound using the elite solutions stored in a long-term memory during the algorithm.

The whole algorithm is presented in Algorithm 4. The algorithm recieves as inputs: 1) a relaxation type, which can be either one of the proposed node-based or traditional relaxations, and 2) the NDO method including either the subgradient or bundle method.

3.8 Experimental Results

We are evaluating the lower bounds of the new Lagrangian relaxations and the performance of the proposed heuristic on C and C+ benchmark instances/., described in [25]. These instances are available online on http://pages.di.unipi.it/frangio/. Table 3.I summarizes the characteristics of the instances where $|A|$, $|N|$, and $|K|$ are the number of arcs, number of nodes, and number of commodities, respectively. There are instances with both loose and tight design capacities. The set also includes instances where the fixed costs are predominant relatively to the variable costs and the reverse. The characteristics of each instance are described in the result tables of Subsections 3.8.2 and 3.8.3.

In the two following subsections, we analyze and compare the five different Lagrangian relaxations, the node-based and traditional relaxations, in terms of the quality of the bounds, computing times, and number of iterations, as well as the performance of the corresponding heuristics in comparison with the state-of-the-art heuristics proposed so far in the literature for the MCFND.

We have tested the bundle and subgradient methods as the non-differentiable optimizer to solve the Lagrangian dual, see, e.g., [26] for more details. The results show

Algorithm 2: Lagrangian-based matheuristic for MCFND

Input relaxation type and NDO method
Solve $\mathscr{P}_{LP} \rightarrow \hat{y}_{lp}$
Initialize Lagrange multipliers
Run primal heuristic using $\hat{y}_{lp}$
repeat
  switch (relaxation type)
    case location**:**
      Solve $\mathscr{P}_L^i$ for all $i \in N \rightarrow \hat{y}_L^p, \forall p \in P$
      for all $p \in P$ **do**
        Run primal heuristic using $\hat{y}_L^p$, if starting condition is true
      end for
    case forward-backward location**:**
      Solve $\mathscr{P}_{LF}^i$ and $\mathscr{P}_{LB}^i$ for all $i \in N \rightarrow \hat{y}_{LF}^p, \hat{y}_{LB}^p, \forall p \in P$
      for all $p \in P$ **do**
        Run primal heuristic using $\hat{y}_{LF}^p$, if starting condition is true
        Run primal heuristic using $\hat{y}_{LB}^p$, if starting condition is true
      end for
    case single-node flow**:**
      Solve $\mathscr{P}_F^i$ for all $i \in N \rightarrow \hat{y}_F^p, \forall p \in P$
      for all $p \in P$ **do**
        Run primal heuristic using $\hat{y}_{LF}^p$, if starting condition is true
        Run primal heuristic using $\hat{y}_{LB}^p$, if starting condition is true
      end for
    case knapsack**:**
      Solve $\mathscr{P}_{Kn}^a$ for all $a \in A \rightarrow \hat{y}_{Kn}$
      Run primal heuristic using $\hat{y}_{Kn}$, if starting condition is true
    case flow**:**
      Solve $\mathscr{P}_{Sh}^k$ for all $k \in K \rightarrow \hat{y}_{Sh}$
      Run primal heuristic using $\hat{y}_{Sh}$, if starting condition is true
  end switch
  Compute lower bound $\rightarrow \hat{y}_{best}$
  switch (NDO method)
    case subgradient**:**
      Update Lagrange multipliers using subgradient
    case bundle**:**
      Update Lagrange multipliers using bundle $\rightarrow \hat{y}_{conv}$
      Run primal heuristic using $\hat{y}_{conv}$
  end switch
until one of the termination conditions is met
Run primal heuristic using $\hat{y}_{best}$
Run post-optimization by solving $\mathscr{P}_{PO}$

Table 3.I – C and C+ Instances of the MCFND

Name	Number of Instances	$\|A\|$	$\|N\|$	$\|K\|$
C	31	230,300,520,700	20,30	40,100,200,400
C+	12	100-400	25,100	10,30

that the bundle method outperforms the subgradient approach significantly in terms of lower bound, time, and number of iterations. Therefore, in this section, the results of the bundle-based method are used to analyze the performance of the relaxations.

3.8.1 Parameter Setting

We use a two-phase parameter setting process to find a suitable set of parameters. In the first phase, we tune the parameters associated with a particular relaxation and the bundle method. Table 3.II displays the tested values of the parameters. Columns FLW, KNP, LOC, FBL, and SNF represent the selected values for the flow, knapsack, location, forward-backward, and single-node flow relaxations, respectively. The initial values are shown in bold in column Tested Values. 1) the bundle method is working better with no initialization, and 2) although the proposed aggregation schemes accelerate the procedure for some of the instances, better results have been found by adding the disaggregated constraints a priori.

In the second phase, we calibrate the primal heuristic parameters in two steps. The first step is to set f^{prim} (primal heuristic frequency) and δ^{dual} (duality gap to start primal heuristic parameter). Since the total number of bundle iterations varies significantly for different relaxations, these two parameters are tuned for each relaxation separately. The rest of the primal heuristic parameters are the internal parameters of the slope scaling and post optimization procedures; therefore, a single value is set for all the relaxations. Table 3.III displays the tested and selected values of the parameters of the first step. Columns FLW, KNP, LOC, FBL, and SNF represent the selected values for the flow, knapsack, location, forward-backward, and single-node flow relaxations, respectively. The initial values are shown in bold in column Tested Values. Table 3.IV shows the parameters and the tested values of the second step. The selected values are shown in bold.

For both phases, we select 30% of the instances randomly for the parameter setting process. To find the best values of the parameters, we fix all the parameters, change one parameter at each time, and select the best value. For the second step of phase two, since a single set of the primal heuristic parameters is selected for all the relaxations, at each step of the calibrations, one of the relaxations is selected randomly for the parameter setting procedure.

3.8.2 Lower Bound Analysis

Table 3.V presents the comparison of the Lagrangian relaxations in terms of lower bound. The first column shows the characteristics of the instances:

— $|N|$: number of nodes;

— $|A|$: number of arcs;

— $|K|$: number of commodities;

— CDI: Cost Dominance Index defined as the variable flow cost average divided by the design cost average computed as follows:

$$\frac{(\sum_{k\in K}\sum_{(i,j)\in A} c_{ij}^{k})/|A| * |K|}{(\sum_{(i,j)\in A} f_{ij})/|A|}$$

F refers to the cases where the fixed costs are predominant relatively to the variable costs, and V means the reverse;

— FTI: Flow capacity Tightness Index defined as the flow capacity average divided

Table 3.II – Values of parameters tested for relaxations and Bundle method

			Selected Values				
Name	**Description**	**Tested Values**	**FLW**	**KNP**	**LOC**	**FBL**	**SNF**
s^{LP}	Lagrangian multipliers initialization switch, 0 no initialization, 1 weak LP, 2 cutting plane LP	0, **1**, 2	0	0	0	0	0
t^{LP}	Lagrangian multipliers initialization time Limit (seconds)	**5**, 10, 15	-	-	-	-	-
s^{agg}	aggregation switch, 1 aggregated master problem, 0 disaggregated	**0**, 1	0	1	0	0	0
pop^{max}	max number of populated solutions to be introduced to bundle	1, **2**, 5, 10	NA	NA	1	1	1
f^{add}	addition frequency of disaggregated constraints, 0 add in advance	0, 1, 5, **10**	NA	NA	NA	0	0
$s^{aggCons}$	1 add aggregated constraints, 0 otherwise	0, 1 **10**	NA	NA	NA	0	0
ε^{add}	precision to check disaggregated constraint violation	10^{-1}, 10^{-3}, 10^{-6}	NA	NA	NA	-	-
it^{Lag}	max number of iterations of NDO solver	500, **1000**, 5000, 10000	1000	10000	1000	1000	1000
t^{*}	stopping parameter of bundle	10^{0}, 10^{1}, 10^{2}, 10^{3}, 10^{4}, 10^{5}, 10^{6}	10^{1}	$10^{1}*$	$10^{1}**$	10^{3}	10^{3}
ε^{Lin}	stopping parameter of bundle	10^{-3}, 10^{-4},10^{-5}	10^{-4}	10^{-4}	10^{-4}	10^{-4}	10^{-4}
$\|B\|$	max bundle size	10000, **20000**, 30000, 40000, 50000	20000	50000	20000	50000	50000

*10^{6} is selected for C+ instances
** 10^{3} is selected for C+ instances

Table 3.III – Values of parameters tested for the primal heuristic

Name	Description	Tested Values	FLW	KNP	LOC	FBL	SNF
f^{prim}	primal heuristic frequency	1, 5, **10**, 15	1	5	10	5	20
δ^{dual}	duality gap to start primal heuristic parameter	**0.5**, 1, 2, 3	-	3	0.5	0.5	2

Table 3.IV – Values of parameters tested for the primal heuristic

Name	Description	Tested Values
t^{Lag}	time limit on Lagrangian procedure including subgradient and primal heuristic (hours)	2, **3**, 4
it^{slope}	max number of slope scaling iterations in each primal heuristic step	15, 20, **25**, 30
n^{intens}	number of solutions to be used in intensification	1, **4**, 8
y_0^{min}	min % of design variables to be fixed to 0 in post optimization	40, **50**, 60
y_0^{max}	max % of design variables to be fixed to 0 in post optimization	70, 80, **90**
st^{prim}	max number of primal heuristic steps with no improvement in upper bound to start diversification	10, 20, **30**, 40
it^{div}	total number of slope scaling steps in each diversification procedure	5, **10**, 15
st^{div}	total number of diversification steps	4, **5**, 6
pop^{max}	number of populated feasible solutions to be used in slope scaling procedure	**1**, 2, 5, 10
s^{conv}	use convexified solution switch, 1 use convexified solution to trigger different slope scaling, 0 otherwise	0, **1**

by the total demand computed as follows:

$$\frac{(\sum_{(i,j)\in A} u_{ij})/|A|}{\sum_{k\in K} d^k}$$

L stands for loose flow capacity, while T refers to tight flow capacity.

Note that t^{Lag}, time limit on Lagrangian procedure, is set to 2 hours for the lower bound analysis. Strong LP column is the strong LP lower bounds obtained by relaxing integrality requirements of the design variables. Columns Flow, Knapsack, Location, FB-Location, and SN-Flow show, respectively, the gaps between the obtained lower bounds of the flow, knapsack, location, forward-backward location and single-node flow relaxations to the strong LP lower bounds computed as $100 \times (LB_S - LB_R)/LB_S$ where LB_S and LB_R are the lower bounds of the strong LP and any of the relaxations, respectively.

Theoretically, the produced lower bounds of the flow and knapsack relaxations have to be equal to the strong LP lower bound, because of the integrality property of these relaxations. Therefore, columns Flow and Knapsack show the computational errors due to the time and iteration limits. The new Lagrangian relaxations, however, do not have the integrality property. The Location, FB-Location, and SN-Flow columns show the improvements of the relaxations upon the strong LP lower bound. The maximum and

average amounts of improvements are significant. The averages for location, forward-backward location, and single-node flow relaxations are 0.5%, 0.9%, and 1.7%, respectively, while the maximum amounts of improvements are 4.7%, 7.7%, and 20.5%, respectively.

Table 3.VI shows the gaps between the obtained lower bounds of the relaxations to the best known upper bounds in the literature (column "Best UB"). Columns "Flow" to "SN-Flow" are the gaps computed as $100 \times (UB_B - LB_R)/UB_B$ where UB_B and LB_R are the best known upper bound in the literature and the lower bound of any of the relaxations, respectively. The results show that, in comparison with the traditional relaxations, the location, forward-backward location and single-node flow relaxations improve, on average, the optimality gaps, by 11%, 20%, and 38%, respectively, if we fix the upper bounds to the best known upper bounds.

Figure 3.3 shows the average of the computational times of the different Lagrangian relaxations and the strong LP relaxation. Although the computational times are reasonable for the location relaxation, the results show a significant computational effort for the forward-backward and single-node flow relaxations. Figure 3.4 presents the portions of the times dedicated to solving Lagrangian subproblems (LagSub) and computing Lagrangian multipliers by solving the master problem (NDO). The figure shows the large amount of time devoted to computing Lagrange multipliers for the flow, knapsack, location and forward-backward relaxations, while most of the total time (65%) is dedicated to solving the Lagrangian subproblems for the single-node flow relaxation.

3.8.3 Upper Bound Analysis

Table 3.VII presents the obtained upper bounds of the Lagrangian heuristics using different Lagrangian relaxations. The results show that the location relaxation outperforms not only the traditional flow-based and knapsack-based heuristics, but also the forward-backward location and single-node flow relaxations. It finds the best upper bounds in 27 of the instances (out of 43), while the flow and knapsack relaxations find the best upper bounds in 18 and 25 of the instances, respectively.

Table 3.VIII presents the comparison of our proposed heuristic (the location-based

Table 3.V – Lagrangian lower bounds comparison with strong LP lower bound

		Lagrangian Gap to Strong LP				
$\|N\|$, $\|A\|$, $\|K\|$, CDI, FTI	Strong LP	Flow	Knapsack	Location	FB-Location	SN-Flow
25,100,10VL	14610.0	0.000	0.000	-0.074	-0.133	-0.224
25,100,10FL	13017.5	0.000	0.000	-1.459	-3.595	-6.794
25,100,10FT	43454.7	0.001	0.000	-2.402	-3.647	-10.332
25,100,30VT	364390.0	0.000	0.000	-0.044	-0.074	-0.145
25,100,30FL	33543.5	0.010	0.000	-0.842	-4.291	-6.815
25,100,30FT	82418.5	0.000	0.000	-0.555	-0.605	-2.168
20,230,40VL	422853.0	0.000	0.000	-0.011	-0.201	-0.207
20,230,40VT	368819.0	0.000	0.000	-0.186	-0.400	-0.613
20,230,40FT	633466.0	0.000	0.000	-0.302	-0.619	-0.878
20,300,40VL	427947.0	0.000	0.000	0.000	-0.256	-0.338
20,300,40FL	575255.0	0.000	0.001	-0.004	-0.011	-1.047
20,300,40VT	460930.0	0.000	0.000	-0.175	-0.312	-0.501
20,300,40FT	596839.0	0.000	0.000	-0.143	-0.269	-0.792
20,230,200VL	91300.6	0.000	0.022	-0.213	-0.189	-0.079
20,230,200FL	132036.0	0.000	0.025	-0.326	-0.273	-0.358
20,230,200VT	95669.3	0.008	0.019	-0.113	-0.098	-0.134
20,230,200FT	131544.0	0.002	0.023	-0.160	-0.125	-0.138
20,300,200VL	73126.9	0.006	0.016	-0.094	-0.091	-0.133
20,300,200FL	110926.0	0.000	0.012	-0.105	-0.075	-0.303
20,300,200VT	74002.7	0.008	0.005	-0.050	-0.045	-0.199
20,300,200FT	103633.0	0.005	0.009	-0.327	-0.315	-0.185
100,400,10VL	27465.3	0.000	0.007	-0.063	-0.598	-1.890
100,400,10FL	19748.1	0.002	0.000	-3.871	-6.351	-11.112
100,400,10FT	48375.4	0.000	0.000	-4.767	-7.713	-20.518
100,400,30VT	380858.0	0.001	0.000	-0.068	-0.135	-0.521
100,400,30FL	45332.4	0.000	0.002	-1.412	-2.299	-5.048
100,400,30FT	117196.0	0.002	0.000	-1.974	-4.790	-10.023
30,520,100VL	53022.9	0.007	0.002	-0.291	-0.309	-0.732
30,520,100FL	90174.2	0.009	0.045	-0.097	-0.067	-0.282
30,520,100VT	51325.6	0.002	0.005	-0.090	-0.397	-0.632
30,520,100FT	94010.8	0.003	0.021	-0.273	-0.301	-0.398
30,700,100VL	47308.1	0.000	0.001	-0.052	-0.061	-0.302
30,700,100FL	58207.2	0.003	0.018	-0.089	-0.094	-0.360
30,700,100VT	45077.7	0.000	0.006	-0.301	-0.388	-0.568
30,700,100FT	53660.9	0.005	0.015	-0.226	-0.307	-0.407
30,520,400VL	111763.0	0.007	0.011	-0.106	-0.070	0.278
30,520,400FL	146680.0	0.005	0.011	-0.085	-0.030	0.647
30,520,400VT	114061.0	0.007	0.012	-0.071	-0.053	0.309
30,520,400FT	149751.0	0.003	0.017	-0.083	-0.031	0.399
30,700,400VL	96605.0	0.000	0.015	-0.080	0.035	1.517
30,700,400FL	130724.0	0.004	0.017	-0.099	0.082	2.454
30,700,400VT	94011.9	0.006	0.010	-0.097	-0.018	1.581
30,700,400FT	127572.0	0.008	0.011	-0.056	0.023	1.397
	Average:	0.003	0.008	-0.508	-0.919	-1.781
	Maximum:	0.010	0.045	0.000	0.082	2.454
	Minimum:	0.000	0.000	-4.767	-7.713	-20.518

Table 3.VI – Lagrangian lower bounds comparison with best known upper bound

		Lagrangian Gap to Best UB				
$\|N\|$, $\|A\|$, $\|K\|$, CDI, FTI	Best UB	Flow	Knapsack	Location	FB-Location	SN-Flow
25,100,10VL	14712.00	0.69	0.69	0.62	0.56	0.47
25,100,10FL	14941.00	12.87	12.87	11.60	9.74	6.95
25,100,10FT	49899.00	12.92	12.91	10.82	9.74	3.92
25,100,30VT	365272.00	0.24	0.24	0.20	0.17	0.10
25,100,30FL	37060.00	9.50	9.49	8.73	5.60	3.32
25,100,30FT	85530.00	3.64	3.64	3.10	3.06	1.55
20,230,40VL	423848.00	0.23	0.23	0.22	0.03	0.03
20,230,40VT	371475.00	0.71	0.72	0.53	0.32	0.11
20,230,40FT	643036.00	1.49	1.49	1.19	0.88	0.62
20,300,40VL	429398.00	0.34	0.34	0.34	0.08	0.00
20,300,40FL	586077.00	1.85	1.85	1.84	1.84	0.82
20,300,40VT	464509.00	0.77	0.77	0.60	0.46	0.27
20,300,40FT	604198.00	1.22	1.22	1.08	0.95	0.44
20,230,200VL	94213.00	3.09	3.11	2.88	2.91	3.01
20,230,200FL	137642.00	4.07	4.10	3.76	3.81	3.73
20,230,200VT	97914.00	2.30	2.31	2.18	2.20	2.16
20,230,200FT	135866.00	3.18	3.20	3.03	3.06	3.05
20,300,200VL	74811.00	2.26	2.27	2.16	2.16	2.12
20,300,200FL	115539.00	3.99	4.00	3.89	3.92	3.70
20,300,200VT	74991.00	1.33	1.32	1.27	1.27	1.12
20,300,200FT	107102.00	3.24	3.25	2.92	2.93	3.06
100,400,10VL	28423.00	3.37	3.38	3.31	2.79	1.54
100,400,10FL	23949.00	17.54	17.54	14.35	12.30	8.38
100,400,10FT	63753.00	24.12	24.12	20.50	18.27	8.55
100,400,30VT	384802.00	1.03	1.02	0.96	0.89	0.51
100,400,30FL	49018.00	7.52	7.52	6.21	5.39	2.85
100,400,30FT	136250.00	13.99	13.98	12.29	9.86	5.36
30,520,100VL	53958.00	1.74	1.74	1.45	1.43	1.01
30,520,100FL	93967.00	4.04	4.08	3.94	3.97	3.77
30,520,100VT	52046.00	1.39	1.39	1.30	0.99	0.76
30,520,100FT	97107.00	3.19	3.21	2.92	2.90	2.80
30,700,100VL	47603.00	0.62	0.62	0.57	0.56	0.32
30,700,100FL	59958.00	2.92	2.94	2.83	2.83	2.57
30,700,100VT	45871.50	1.73	1.74	1.44	1.35	1.17
30,700,100FT	54904.00	2.27	2.28	2.04	1.96	1.87
30,520,400VL	112774.40	0.90	0.91	0.79	0.83	1.17
30,520,400FL	149335.40	1.78	1.79	1.69	1.75	2.41
30,520,400VT	114640.00	0.51	0.52	0.43	0.45	0.81
30,520,400FT	152510.00	1.81	1.83	1.73	1.78	2.20
30,700,400VL	97875.00	1.30	1.31	1.22	1.33	2.80
30,700,400FL	134589.80	2.88	2.89	2.78	2.95	5.26
30,700,400VT	95249.60	1.31	1.31	1.20	1.28	2.86
30,700,400FT	129909.60	1.81	1.81	1.74	1.82	3.17
Average:		3.90	3.91	3.46	3.10	2.39
Average Improvement:				0.44	0.80	1.51
Average Improvement (%):				11.35	20.46	38.76
Maximum:		24.12	24.12	20.50	18.27	8.55
Minimum:		0.23	0.23	0.20	0.03	0.00

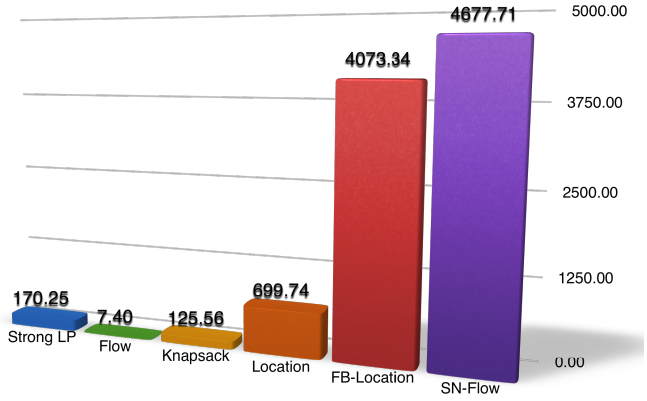

Figure 3.3 – Average time of Lagrangian relaxation bounding procedures and Strong LP relaxation

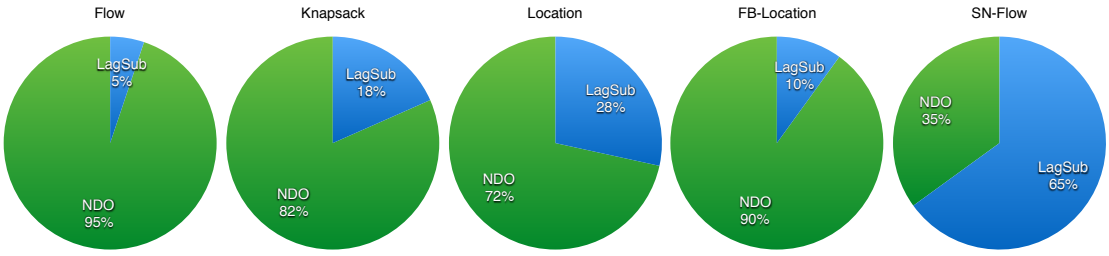

Figure 3.4 – Elapsed time analysis of Lagrangian relaxation bounding procedures

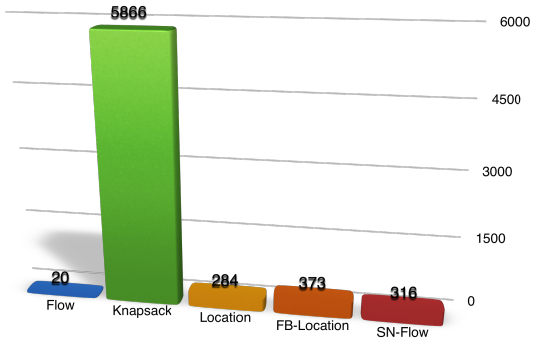

Figure 3.5 – Average number of iterations of Lagrangian relaxation bounding procedures

heuristic) with the state-of-the-art heuristics proposed so far in the literature. LMH column presents the upper bounds of our proposed bundle-based location Lagrangian heuristic; while CTS, PR, MCA, CSH, IPS, LocalB, SACG(2), CEA, CCL(2), and ILP columns are the gaps between the LMH upper bounds and, respectively, the upper bounds of the cycle-based tabu search [47], path relinking [48], multilevel cooperative algorithm [28], capacity scaling heuristic [55], integer programming search [52], local branching [80], simulated annealing column generation [86], cycle-based evolutionary algorithm [78], combined capacity scaling and local branching [54] and iterative linear programming [45]. The results show that the proposed algorithm outperforms all the previously proposed heuristics in the literature on average except the recent ones. However, the proposed algorithm produces almost the same upper bounds as the last two methods in much less computational times, see Table 3.IX.

Table 3.IX compares the computational times with those of the heuristics in the literature. The computational times of the heuristics are normalized based on the CPU type and the number of used cores, U. We use a method described in [35] and data from http://www.cpubenchmark.net/. The method is to take Passmark CPU Score (PCPUS) and normalize the computational times, t_{norm}, using the following equation:

$$t_{norm} = t_{real} * (PCPUS_h / PCPUS_l) * U \tag{3.91}$$

where t_{real}, $PCPUS_h$ and $PCPUS_l$ are the real computational time, PCPUS of the heuristic, and PCPUS of our machine.

The table shows that the proposed method is much faster than the best heuristics in the literature, except the recent one proposed by Gendron et al. [45]. Note that the PCPUSs are not available for CTS and PR methods, and the computational times of MCA are not reported in [28]. Therefore, the computaional times of these methods are not presented in the table.

Figure 3.6 analyzes the computational time of the heuristics and shows the portion of each phase of the algorithm: solving Lagrangian subproblems (LagSub), computing Lagrange multipliers (NDO), Lagrangian Heuristic (Heur), and post-optimization phase

Table 3.VII – Upper bound comparison of different Lagrangian Heuristics

$\|N\|$, $\|A\|$, $\|K\|$, CDI, FTI	Flow	Knapsack	Location	FB-Location	SN-Flow
25,100,10VL	**14712**	**14712**	**14712**	**14712**	**14712**
25,100,10FL	15540	14954	**14941**	**14941**	15239
25,100,10FT	50969	**50472**	50853	50969	50994
25,100,30VT	**365272**	**365272**	**365272**	**365272**	**365272**
25,100,30FL	38482	37747	37622	**37500**	37535
25,100,30FT	**85530**	**85530**	**85530**	**85530**	**85530**
20,230,40VL	425290	**423848**	**423848**	**423848**	**423848**
20,230,40VT	**371475**	**371475**	**371475**	**371475**	**371475**
20,230,40FT	644232	**643036**	**643036**	**643036**	645324
20,300,40VL	431263	**429398**	**429398**	**429398**	**429398**
20,300,40FL	586406	**586077**	**586077**	**586077**	**586077**
20,300,40VT	**464509**	**464509**	**464509**	**464509**	**464509**
20,300,40FT	**604198**	**604198**	**604198**	**604198**	**604198**
20,230,200VL	94228	94218	**94213**	94218	94468
20,230,200FL	137854	**137764**	**137764**	137854	141637
20,230,200VT	**97914**	**97914**	**97914**	**97914**	101718
20,230,200FT	136618	136102	**135962**	136303	137464
20,300,200VL	**74811**	74900	74900	74945	74874
20,300,200FL	116255	**115826**	115925	116241	115855
20,300,200VT	75358.7	**74991**	**74991**	**74991**	75432
20,300,200FT	**107102**	107315	**107102**	107574	107169
100,400,10VL	**28423**	**28423**	**28423**	**28423**	**28423**
100,400,10FL	25124	**23949**	24184	**23949**	**23949**
100,400,10FT	68192	64713	65983	**64662**	65112
100,400,30VT	**384802**	**384802**	**384802**	**384802**	**384802**
100,400,30FL	51710	**49018**	50022	49220	49633
100,400,30FT	**137466**	137883	141508	137798	141483
30,520,100VL	53968	53968	53968	53964	**53958**
30,520,100FL	**93967**	94033	94129	**93967**	94288
30,520,100VT	52134	**52046**	**52046**	**52046**	52119
30,520,100FT	97854	**97107**	97348	97348	97754
30,700,100VL	**47603**	**47603**	**47603**	**47603**	**47603**
30,700,100FL	60067	60017.3	**59995**	60056.5	60019
30,700,100VT	45879	45872	**45871.5**	**45871.5**	45879
30,700,100FT	54965	**54904**	**54904**	54955	54974
30,520,400VL	112897	**112784**	112968	112901	113404
30,520,400FL	149498	149551	**149329**	149832	150184
30,520,400VT	114687	**114640**	**114640**	**114640**	114851
30,520,400FT	152924	152922	**152722**	154000	154825
30,700,400VL	**97886**	97970	98031	97982	98151.2
30,700,400FL	**134789**	136176	137070	142489	164175
30,700,400VT	95358.7	**95331.8**	95396.1	95449.9	95652.5
30,700,400FT	130624	130794	**130312**	132074	136932
# Best UB out of 43:	15	26	28	23	14

Table 3.VIII – Proposed heuristic VS state-of-the-art heuristics in literature

$\|N\|$, $\|A\|$, $\|K\|$, CDI, FTI	LMH[1]	CTS	PR	MCA	CSH	IPS	LocalB	SACG	SACG2	CEA	CCL	CCL	ILP
25,100,10VL	14712	0.00	0.00	0.00	0.00	NA	0.00	0.00	0.00	0.00	NA	NA	0.00
25,100,10FL	14941	0.00	0.00	0.00	-0.64	NA	0.00	0.00	0.00	0.00	NA	NA	0.00
25,100,10FT	50853	1.88	1.88	1.80	0.16	NA	1.80	1.88	1.88	1.88	NA	NA	1.88
25,100,30VT	365272	-0.03	-0.03	-0.03	0.00	NA	0.00	0.00	0.00	0.00	NA	NA	0.00
25,100,30FL	37622	0.10	-0.09	0.04	0.40	NA	0.79	1.49	1.49	0.79	NA	NA	1.51
25,100,30FT	85530	-0.90	-1.05	-1.09	-0.32	NA	0.00	0.00	0.00	0.00	NA	NA	0.00
20,230,40VL	423848	-0.22	-0.13	-0.67	-0.05	-0.13	0.00	0.00	0.00	0.00	0.00	0.00	0.00
20,230,40VT	371475	-0.11	-0.09	0.00	-0.12	-0.08	0.00	0.00	0.00	0.00	0.00	0.00	0.00
20,230,40FT	643036	-0.43	-0.39	-1.53	-0.23	-0.02	0.00	0.00	0.00	-0.02	0.00	0.00	0.00
20,300,40VL	429398	-0.03	0.00	-0.10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
20,300,40FL	586077	-1.24	-0.74	-1.27	-0.29	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
20,300,40VT	464509	-0.05	0.00	-0.32	-0.01	0.00	0.00	-0.03	-0.03	0.00	0.00	0.00	0.00
20,300,40FT	604198	-0.48	-0.96	-2.48	0.00	0.00	0.00	-0.00	-0.00	0.00	0.00	0.00	0.00
20,230,200VL	94213	-5.08	-6.57	-4.64	-0.04	-0.94	-1.15	-1.15	-0.07	-0.27	0.00	0.00	0.00
20,230,200FL	137764	-6.37	-7.42	-3.91	0.09	-2.53	-4.12	-1.45	-0.06	-0.86	0.09	0.09	-0.29
20,230,200VT	97914	-6.98	-6.92	-4.20	-0.06	-1.53	-0.12	-0.07	0.00	-0.30	0.00	0.00	0.00
20,230,200FT	135962	-8.40	-8.53	-3.84	-0.12	-3.17	-3.80	-3.53	-0.82	-0.86	-0.05	0.07	-0.41
20,300,200VL	74900	-7.90	-4.38	-4.42	-0.02	-0.56	-1.97	-1.47	-0.00	-0.51	0.12	0.12	-0.09
20,300,200FL	115925	-6.40	-6.52	-5.20	0.12	-1.40	-2.78	-1.74	-0.44	-0.76	0.15	0.33	-0.39
20,300,200VT	74991	-6.17	-5.17	-3.01	-0.41	-1.61	-1.57	-0.57	0.00	-0.60	0.00	0.00	0.00
20,300,200FT	107102	-6.89	-6.05	-3.80	-0.71	-3.03	-2.53	-2.44	-1.43	-0.41	-0.20	0.00	-0.18
100,400,10VL	28423	-0.89	-0.22	-0.46	-0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
100,400,10FL	24184	0.97	0.67	0.67	-1.14	0.97	-2.09	0.97	0.97	0.97	0.97	0.97	0.97
100,400,10FT	65983	-1.56	1.07	-0.46	-11.49	0.15	-2.08	1.23	1.23	0.64	2.69	3.38	1.12
100,400,30VT	384802	-0.18	-0.03	-0.12	-0.02	-0.01	-0.00	0.00	0.00	-0.05	0.00	0.00	0.00
100,400,30FL	50022	-3.06	-2.60	-0.87	-3.87	0.66	0.30	0.76	1.54	1.11	2.01	2.01	2.01
100,400,30FT	141508	-2.57	0.11	-2.98	-1.98	0.10	-0.09	-0.02	0.35	1.39	2.37	3.72	1.65
30,520,100VL	53968	-1.83	-1.73	-3.31	-0.22	-0.27	-0.11	-0.03	-0.03	-0.24	0.02	0.02	0.02
30,520,100FL	94129	-5.80	-8.42	-6.04	-0.71	-0.28	-2.26	-0.19	0.07	-0.52	0.17	0.17	0.07
30,520,100VT	52046	-1.80	-1.87	-2.82	-0.45	-0.25	-0.16	-0.66	-0.39	-0.26	0.00	0.00	0.00
30,520,100FT	97348	-8.40	-9.02	-5.27	-1.53	-1.58	-3.86	-2.91	-1.23	-0.52	-0.04	0.25	-0.06
30,700,100VL	47603	-1.67	-2.35	-2.66	-0.07	-0.02	0.00	-0.83	0.00	0.00	0.00	0.00	0.00
30,700,100FL	59995	-4.13	-5.16	-6.27	-0.33	-1.18	-0.46	-0.98	-0.66	-0.91	0.06	0.06	-0.09
30,700,100VT	45871.5	-2.51	-2.92	-3.46	-0.65	-0.38	-0.07	-0.60	-0.18	-0.46	0.00	0.00	-0.08
30,700,100FT	54904	-5.43	-3.04	-3.65	-0.83	-1.28	-0.36	-0.20	-0.13	-0.42	0.00	0.00	0.00
30,520,400VL	112968	-6.80	-5.71	-2.39	0.11	-0.95	-1.24	-1.09	-0.67	-0.20	0.17	0.17	-0.01
30,520,400FL	149329	-7.88	-9.23	-4.87	-0.08	-3.27	-5.62	-4.36	-1.13	-1.22	-0.06	-0.00	-0.41
30,520,400VT	114640	-6.06	-4.82	-5.53	-0.00	-0.25	-0.52	-0.85	-0.82	-0.92	0.00	0.00	-0.14
30,520,400FT	152722	-9.96	-7.17	-4.91	-0.01	-1.23	-10.37	-4.75	NA	-1.12	0.10	0.14	-0.74
30,700,400VL	98031	-8.92	-7.23	-4.69	0.06	-0.70	-4.69	-3.77	-1.31	-0.71	0.16	0.16	-0.36
30,700,400FL	137070	-8.67	-5.80	-5.05	1.46	-11.31	-5.05	-17.11	2.26	-0.03	1.79	1.81	-1.89
30,700,400VT	95396.1	-6.58	-6.10	-3.98	0.09	-0.81	-3.98	-1.73	-0.15	-0.77	0.15	0.15	-0.35
30,700,400FT	130312	-9.57	-8.21	-6.10	0.13	-1.01	-6.10	-8.58	-0.89	-1.62	0.31	0.31	-0.64
Average:		-3.70	-3.32	-2.65	-0.55	-1.02	-1.49	-1.27	-0.02	-0.18	0.30	0.38	0.07
Maximum:		1.88	1.88	1.80	1.46	0.97	1.80	1.88	2.26	1.88	2.69	3.72	2.01
Minimum:		-9.96	-9.23	-6.27	-11.49	-11.31	-10.37	-17.11	-1.43	-1.62	-0.20	-0.00	-1.89
# of improvements:		38	35	37	30	28	27	27	20	25	4	1	16
# of draws		2	4	3	4	5	13	11	14	12	17	18	19
# of non-improvements		3	4	3	9	4	3	5	8	6	16	18	8

[1] The upper bounds found by the proposed Lagrangian Matheuristic.

[2] Columns 3-14 show the gaps between the Lagrangian Matheuristic and the existing heuristics in the literature. A negative value means that our Lagrangian heuristic provides better upper bound than the heuristic in the literature.

(PostOpt). The figure shows the high computational effort on the post-optimization phase for flow, knapsack, and location heuristics. For the forward-backward relaxation, computing the Lagrange multipliers and the post-optimization phase take most of the computational time. For the single-node relaxation, however, the computational time is shared almost evenly among the LagSub, NDO, and PostOpt portions.

3.9 Conclusion

We have proposed three new node-based Lagrangian relaxation-reformulations for the multicommodity capacitated fixed-charge network design problem. A Lagrangian-based matheuristic has also been proposed to find upper bounds. We have conducted significant computational experiments on the benchmark instances. We can summarize the obtained results as follows:

— The Lagrangian dual bound of the new Lagrangian relaxations improve significantly upon the strong LP bound (known to be equal to the Lagrangian dual bounds of the flow and knapsack relaxations).

— The proposed Lagrangian heuristic based on the location relaxation outperforms traditional flow and knapsack based heuristics.

— The proposed algorithm outperforms almost all the previously proposed heuristics in the literature on average.

Two main fascinating research avenues are: 1) developing exact solution methods by embedding the proposed Lagrangian relaxation into a branch-and-bound procedure, and 2) improving the solution time of the subproblems of the new relaxations using the

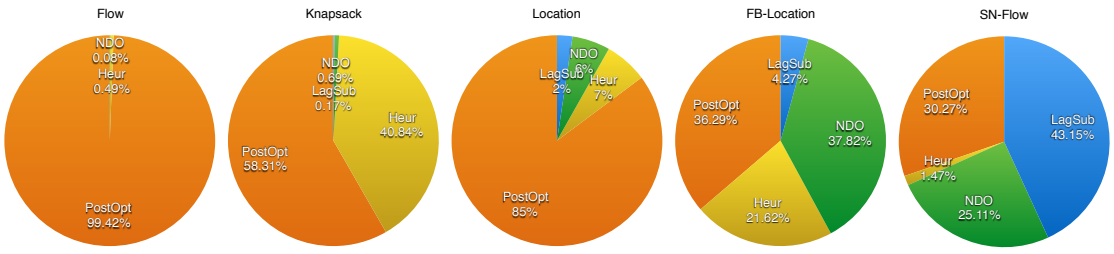

Figure 3.6 – Time analysis of Lagrangian-based heuristics

Table 3.IX – Comparing normalized computational time of proposed heuristic to state-of-the-art heuristics in literature

	LMH 5h	CSH	IPS	LocalB	SACG	SACG2	CEA	CCL	CCL	ILP
CPU:	Intel Xeon X5675 @ 3.07GHz	Intel Pentium D 940 @ 3.20GHz	Intel Xeon X3350 @ 2.66GHz	Intel Core2 Duo E4600 @ 2.40GHz	Intel Core2 Duo E6850 @ 3.00GHz	Intel Core2 Duo E6850 @ 3.00GHz	Intel Xeon E5507 @ 2.27GHz	Intel Core i7 4770 @ 3.40GHz	Intel Core i7 4770 @ 3.40GHz	Intel Core i7 4900MQ @ 2.80GHz
CPU PassMark:	8507	710	3876	1383.00	1951	1951	3144	9792	9792	9060
Used Cores:	1	2	8	2	2	2	1	4	4	1
$\|N\|$, $\|A\|$, $\|K\|$, CDI, FTI										
25,100,10VL	2.64	0.22	NA	3.96	11.47	11.47	4.73	NA	NA	0.10
25,100,10FL	10.4	1.19	NA	195.09	7.80	7.80	10.75	NA	NA	4.52
25,100,10FT	14.23	0.62	NA	195.09	13.30	13.30	323.86	NA	NA	57.96
25,100,30VT	44.18	0.53	NA	52.35	29.81	29.81	47.20	NA	NA	0.43
25,100,30FL	81.31	2.45	NA	195.09	33.94	33.94	290.60	NA	NA	60.48
25,100,30FT	87.49	1.67	NA	195.09	92.19	92.19	17.74	NA	NA	9.80
20,230,40VL	22.65	0.50	14.58	106.63	54.12	54.12	399.96	5.53	5.06	1.41
20,230,40VT	36.52	0.55	149.44	143.11	177.05	177.05	45.75	14.27	17.50	1.04
20,230,40FT	37.64	0.63	164.02	195.09	77.06	77.06	356.72	67.22	49.73	4.00
20,300,40VL	27.4	0.53	69.25	47.41	90.36	90.36	39.99	5.53	4.14	179.67
20,300,40FL	65.35	1.03	105.70	195.09	129.81	129.81	140.03	38.21	47.42	22.64
20,300,40VT	54.56	0.63	87.48	195.09	50.91	50.91	1277.30	17.96	17.04	0.49
20,300,40FT	63.84	0.73	247.86	195.09	84.40	84.40	98.75	20.26	17.50	137.85
20,230,200VL	4081.11	73.80	2996.19	195.09	275.21	1128.36	1391.54	14090.26	30106.00	2569.34
20,230,200FL	10284.3	276.76	2518.69	195.09	275.21	504.55	1498.34	20299.49	40973.31	333.19
20,230,200VT	1825.65	87.38	2992.54	195.09	275.21	1078.36	1219.72	9948.77	13328.26	255.11
20,230,200FT	11527.1	324.46	568.62	195.09	275.21	572.89	1156.48	20893.44	52461.27	1009.01
20,300,200VL	17949.1	58.04	2923.29	195.09	275.21	1359.53	1199.58	23241.12	42161.20	340.70
20,300,200FL	17948.5	215.31	2500.47	195.09	275.21	479.78	1519.56	35769.17	39517.46	2728.90
20,300,200VT	1408.59	71.56	1414.26	195.09	275.21	1844.82	1318.36	9935.42	11787.69	150.37
20,300,200FT	17934.4	287.39	1443.42	195.09	275.21	1140.28	2891.25	17487.70	37935.45	1337.67
100,400,10VL	50.69	0.97	127.57	177.83	74.77	74.77	36.29	8.75	10.59	413.97
100,400,10FL	127.82	15.57	32.80	195.09	88.98	88.98	209.62	489.43	505.54	1148.93
100,400,10FT	4548.57	9.28	2963.38	195.09	77.52	77.52	1397.78	12594.81	59757.56	2937.16
100,400,30VT	2792.06	2.97	1202.85	195.09	196.32	196.32	734.35	6920.13	9696.46	783.01
100,400,30FL	3245.25	103.74	3229.47	195.09	275.21	429.78	1479.49	3975.73	12169.84	649.49
100,400,30FT	7426.14	25.69	3236.76	195.09	275.21	553.63	4717.18	50774.29	112500.15	1933.90
30,520,100VL	1205.93	4.49	794.61	195.09	275.21	629.31	1967.45	3096.33	7684.88	1289.21
30,520,100FL	17953.4	67.85	823.77	195.09	275.21	425.66	2304.02	15039.19	30308.58	2934.08
30,520,100VT	17948	6.26	1658.47	195.09	275.21	4186.38	4759.50	9226.83	18713.34	1675.25
30,520,100FT	17950.3	33.33	2970.67	195.09	275.21	462.35	1359.34	22107.11	83229.81	2967.63
30,700,100VL	551.35	6.43	116.64	195.09	275.21	2652.55	1515.57	295.13	354.98	21.25
30,700,100FL	17941.1	19.11	2700.94	195.09	275.21	1831.05	4562.51	13298.33	32047.59	1650.13
30,700,100VT	17947	7.73	1352.29	195.09	275.21	1926.92	1774.42	25595.25	37311.12	872.87
30,700,100FT	17944.3	16.16	1410.61	195.09	275.21	3139.21	4225.09	20515.89	36425.73	1961.25
30,520,400VL	17954.6	94.81	1436.13	195.09	275.21	5918.82	5664.97	27242.18	48228.16	842.29
30,520,400FL	17955.5	435.73	2733.75	195.09	275.21	4463.43	4558.08	15593.07	83088.00	462.49
30,520,400VT	17952.5	38.39	2263.54	195.09	275.21	8256.26	1564.90	17154.36	41117.42	2561.76
30,520,400FT	17955.8	279.41	1698.57	195.09	275.21	4286.83	2385.07	23705.69	69290.57	1637.73
30,700,400VL	17957.2	79.25	809.19	195.09	275.21	8256.26	2345.23	21763.63	46253.42	2890.34
30,700,400FL	17966	297.60	3134.70	195.09	275.21	6413.28	1277.37	33664.59	71184.28	1135.52
30,700,400VT	17956.1	125.04	1330.42	195.09	275.21	8256.26	3715.44	29357.81	53427.23	941.52
30,700,400FT	17959.6	248.23	820.12	195.09	275.21	6552.72	4561.63	37591.52	72695.84	1567.50
Average:	8251	77.30	1487.65	180	196	1815	1683	14644	32282	988

existing state-of-the-art algorithms for the CFLP in the literature.

CHAPTER 4

A SURVEY OF MULTILAYER NETWORK DESIGN

In multilayer network design, decisions are represented by different potential networks, each at a given layer. In each layer, flow requirements for a set of commodities must be satisfied. To route the commodities, appropriate arcs have to be selected (or opened) in each layer. There are several types of coupling constraints between the layers. For example, to open an arc in a particular layer, supporting arcs in another layer have to be opened. Applications of the multilayer network design problem can be found in the fields of transportation and telecommunications. Although this is an important class of problems in combinatorial optimization, to the best of our knowledge, there is no survey on the topic which covers extensively multilayer network design problems. In this paper, we propose the first classification and a state-of-the-art survey of multilayer network design problems. The survey focuses on applications in transportation and telecommunications, as well as on solution methods. We also propose a general modeling framework that encompasses most multilayer network design problems found in the literature.

4.1 Introduction

Network design is a well-known and important class of problems in combinatorial optimization. *Multilayer network design* represents a special case of network design that has major applications in the fields of transportation [see, e.g., 17, 31, 88] and telecommunications [see, e.g., 33, 61]. Unlike a typical network design problem, in multilayer network design, there are several networks, each at a given layer. Each network has its nodes, potential arcs with (or without) limited capacities, and, possibly, commodities. The demands of commodities, if any, need to be routed from their origins to their destinations in each layer. To route the commodities, appropriate arcs have to be selected (or opened) by paying a fixed cost. A particular layer might not have any commodity,

but still has to be designed to support the routing of other layers. At least one layer has commodities to route.

In multilayer network design, there are two types of coupling constraints between the layers, *flow connectivity* and *design connectivity* requirements. A common type of design connectivity requirement arises when each link in a layer can be selected only if some arcs (typically forming a path or a cycle) are opened in another layer. The flows in a layer might also be related to the flows of another layer, corresponding to flow connectivity requirements. For example, the amount of flow on each arc in a particular layer might be computed based on the flow on several arcs in another layer.

Connectivity requirements between layers might be either one-to-one or one-to-many. When each layer is supporting or is supported by only one other layer, the connectivity requirement is one-to-one while a one-to-many connectivity requirement exists when at least one of the layers is supporting or is supported by more than one layer. Note that, for two-layer network design problems, only the one-to-one connectivity is possible. In general, the objective is to find a minimum cost design and routing for all layers, while satisfying typical network design constraints in each layer, as well as coupling constraints between layers.

Multilayer network design is often, but not only, used to integrate decisions at the same planning level or different planning levels: strategic, tactical and operational. Solving the multilayer network design problem typically generates an optimal solution that cannot be obtained by solving sequentially each of the single-layer network design problems, thus yielding significant cost savings.

An example of such an integration can be found in railway freight planning, where cars have to be classified in groups called *blocks*. Then, blocks are grouped into services to make up trains moving blocks between terminals. Grouping cars into blocks avoids performing operations on each car individually in each terminal, which reduces the number of operations to be performed in each terminal. Zhu et al. [88] presented both problems of determining blocks (which block to be built) and selecting services in a single *integrated freight rail service network design problem*. They represent the problem using a three-layer network including car, block and service layers. Each layer consists

of a *time-space network*, where the terminals (physical nodes) are duplicated over the time horizon to represent the time dependency. A node in such a network represents a terminal at a specific time, and each arc represents a transfer from a terminal at a given time to either the same terminal at another time or another terminal at another time. The service layer, includes moving and stop links of services. The block network determines the classification policy, which includes service section arcs (each corresponds to a chain of moving and stop arcs in the service layer) and block transfer arcs to move blocks between service sections. The car layer consists of block links (each corresponds to a chain of block transfer arcs and service sections in the block layer) and car arcs on which cars are moved in each terminal. To open a service section arc in the block layer, a chain of moving and stop link should be open in the service layer. To select a projected block link in the car layer, a chain of block and projected service section links need to be open in the block layer.

Another example can be found in telecommunications, where one layer might be an internet (virtual) network whose arcs are supported by the arcs in an optical fiber (physical) layer. A chain of supporting arcs has to be opened in the physical layer to open an arc in the internet network. In this example, there is an integration of a strategic decision (physical network design) with a tactical one (virtual network design).

The applications of network design models and their solution techniques have been surveyed in [64], [70], and [23]. In recent years, a growing number of applications of multilayer network design have appeared that are not covered in these surveys. To the best of our knowledge, the only survey paper is Kivelä et al. [59], which does not cover extensively multilayer network design problems (only a few references on telecommunications applications are cited).

The contribution of this paper is threefold. First, we propose a classification of multilayer network design problems, which emphasizes the multilayer features, such as the number of layers and the type of coupling constraints between layers. Second, we synthesize the applications in transportation and telecommunications, as well as the methods used to solve multilayer network design problems. Third, we propose a general modeling framework that encompasses most multilayer network design problems found in the

literature.

The paper is organized as follows. In Section 4.2, we propose a detailed definition of multilayer network design, as well as a general modeling framework for network design model. In Section 4.3, we present our classification of multilayer network design. Based on the classification and the applications currently proposed in the literature, we identify four existing classes of problems: 1) two-layer network design problems with design connectivity requirements; 2) two-layer network design problems with flow connectivity requirements; 3) three-layer network design problems with one-to-many flow connectivity requirements; 4) three-layer network design problems with one-to-one flow-design connectivity requirements. For each these four classes of problems, a detailed survey is presented in Sections 4.4, 4.5, 4.6, and 4.7, respectively. In particular, we show how the proposed general modeling framework models most of the problems presented in the literature. In Section 4.8, we provide a survey on the proposed methods for solving multilayer network design problems. In Section 4.9, we summarize this work and we discuss future research directions.

4.2 Multilayer Network Design, Definition and Formulation

In this section, we first propose a definition of multilayer network design. We then propose a general modeling framework.

4.2.1 Definition

In network design, given a potential network (for simplicity, we assume all arcs are potential) that might have capacitated arcs, several commodities such as goods, data or people, have to be routed between different origin and destination points. A network has to be constructed by opening appropriate arcs between pairs of nodes to route the commodities. In addition to *flow costs*, *design costs* are associated to each arc. Flow costs are incurred when routing commodities on each arc, while a design cost is incurred when opening (or selecting) an arc. The problem is to select the arcs such that the demands for the commodities can be routed on the constructed network, and arc capacities

are respected. The designed network and the final routing must minimize the total cost.

In multilayer network design, instead of one network, there are several networks. Each network corresponds to a layer that consists of nodes and potential arcs. In each layer, there might be several commodities to be routed in the network to satisfy demands between origin and destination nodes. A network has to be designed in each layer to satisfy the demands for the commodities. Note that some layers might not have any commodity, but their arcs have to be opened to support the routing of the commodities in other layers. When there is only one layer that has commodities to route, we have a *multilayer single flow-type network design problem*. When we have commodities on more than one layer, we obtain a *multilayer multiple flow-type network design problem*.

In addition to flow and design costs, as well as flow capacities that might be associated with arcs, in multilayer network design, there are typically two types of coupling constraints between layers: *design connectivity* and *flow connectivity*. The first one, design connectivity, means that an arc opened in a given layer requires some arcs to be opened in another layer. If arc a of layer l' requires a set of arcs (for example a set including arcs b and c) to be opened in layer l, then l' is said to be *supported* by l, and l is said to be *supporting* of l'. In addition, a is *supported* by b and c, while b and c are *supporting* a.

An illustration is given in Figure 4.1, where arcs 1 and 2 in layer l' are supported, respectively, by paths (3,4) and (5,6,4) in layer l. Therefore, l' is supported by l, and l is supporting l'. In this particular example, to use arc 1 in layer l', all its supporting arcs in layer l, including arcs 3 and 4, have to be opened. For instance, in the integrated freight rail service network design problem, the design connectivity constraint consists of opening a chain of supporting services in the service layer to select the corresponding block in the block layer. In telecommunications, the design connectivity constraint forces an arc opened in the virtual network (the network supported by the physical layer) to be supported by a chain of physical arcs in the physical layer (the supporting network of the virtual layer). Note that, design connectivity requirements are not limited to the above examples. Another type arises when an arc in a layer requires at least one of the supporting arcs to be opened in another layer (for more details, see the next subsection

where we describe different types of design connectivity requirements).

Based on the design connectivity constraints, we can define the *design capacity* constraints, a new concept in multilayer network design. The design capacity constraint of arc b in supporting layer l limits the number of selected arcs in supported layer l' for which arc b is the supporting arc. To clarify the definition of design capacity constraints, consider Figure 4.1. If the design capacity of arc 4 in layer l is equal to 1, then at most one of arcs 1 or 2 in layer l' can be opened in a feasible solution. For example, in the integrated freight rail service network design problem, a design capacity is defined for each service s and limits the number of selected blocks for which service s serves as the supporting service. In telecommunications, a design capacity constraint might be defined for each physical arc p to limit the number of opened virtual arcs for which arc p serves as a supporting arc.

The second type of connectivity constraints, flow connectivity, relates the flows between different layers. The simplest such constraint arises when the flow on arc b is equal to the summation of the flows on all arcs for which b is a supporting arc. In Figure 4.1, for example, the flow on arc 4 would be equal to the summation of the flows on arcs 1 and 2. Note that, with this particular type of flow connectivity requirements, when only one layer has commodities to route, the flows on other layers can be deduced from the flows on that single layer. Such a problem would be considered as a multilayer single flow-type network design problem, even though there are flows on several layers. In the next subsection, we describe other types of flow connectivity constraints.

In some applications of multilayer network design, certain arcs of a layer can be independent of other layers. For example, in the integrated freight rail service network design problem, there are some arcs to move cars in each terminal that are not related to any arc of other layers.

Some problems introduced in the literature appear at first sight to be similar to the multilayer network design problem. These problems include the *multi-echelon network design problem* [18, 29], the *multilevel network design problem* [7, 21], and the *hierarchical network design problem* [63, 72]. There are two main differences between these problems and multilayer network design. First, in multilayer network design, each layer

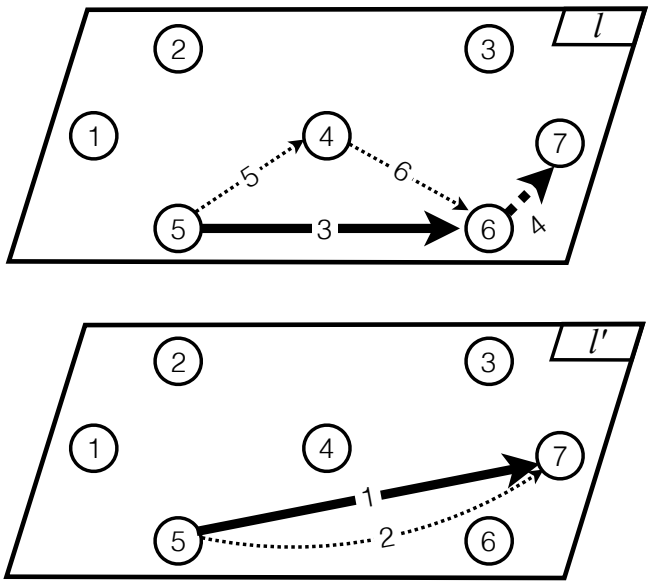

Figure 4.1 – Multilayer network design example illustrating design connectivity constraints: arc 1 in layer l' is supported by the path made of arcs 3 and 4 in layer l, and arc 2 in layer l' is supported by the path made of arcs 5, 6, and 4

corresponds to a network including nodes, potential arcs, commodities to be routed on the designed network, while the concepts of echelon, level, and hierarchy do not necessarily correspond to a network with all these characteristics. The second main difference is the flow and design connectivity constraints between layers, which do not explicitly exist in the above problems.

4.2.2 Formulation

Given a set of layers L and a network $G_l = (N_l, A_l)$ for each layer $l \in L$, where N_l and A_l are the sets of nodes and arcs of layer $l \in L$, respectively, we define u_{al} and v_{al} as the flow capacity and the design capacity of arc $a \in A_l$ in layer $l \in L$. Let $A_l^+(n)$ and $A_l^-(n)$ represent the sets of outgoing and incoming arcs of node $n \in N_l$. A set of commodities K_l has to be routed through the network of layer $l \in L$. The set K_l might be empty, which means that there is no commodity to be routed in layer l. The amount of each commodity $k \in K_l$ that must flow from its origin $O(k) \in N_l$ to its destination $D(k) \in N_l$ is d^k.

We denote by C the set of ordered pairs (l,l') such that $l' \in L$ is a layer supported by $l \in L$. In other words, C contains the pairs of layers having a (design or flow) connectivity requirement between one another. Hence, we call C the set of connectivity requirement pairs. Let $B^{al}_{l'}$ be the set of arcs in layer l' supported by arc $a \in A_l$. For example, in Figure 4.1, this set for arc 4 in layer l is $\{1,2\}$. Let $D^{l}_{bl'}$ be the set of arcs in layer l supporting arc $b \in A_{l'}$. In Figure 4.1, this set for arc 1 in layer l' is $\{3,4\}$.

Two sets of decision variables are considered to formulate the problem, design and flow variables. The design variables could be binary or integer, depending on the particular application. When the decision is to open (select) or close (not to select) arc $a \in A_l$ of layer $l \in L$, then the design variable y_{al} assumes binary values. When the goal is to determine the number of capacity units on each arc $a \in A_l$ of layer $l \in L$, then the design variable y_{al} has integer values. The flow variables could take binary or continuous values depending on the problem. When the flow of each commodity has to be routed through a single path from its origin to its destination (non-bifurcated flows), then the flow variables take binary values. The variable x^k_{al} then indicates if commodity $k \in K_l$ of layer $l \in L$ uses arc $a \in A_l$ or not. When the flow of each commodity can be distributed through several paths, then the flow variable x^k_{al} is continuous, representing the fraction of the demand of commodity $k \in K_l$ of layer $l \in L$ on arc $a \in A_l$. Sets X and Y define required side constraints, as well as the domains of the flow and design variables, respectively. Set $(X,Y)_{ll'}$ defines the coupling constraints for each pair of layers $(l,l') \in C$, which captures some application-specific connectivity requirements. We present several possible coupling constraints later in this section.

We use notations $\Psi(x)$ and $\Phi(y)$ to represent the total routing cost function and the total design cost function, respectively. The proposed general multilayer network design formulation (MLND) can be stated as follows:

$$\min \quad \Psi(x) + \Phi(y) \tag{4.1}$$

subject to

$$\sum_{a \in A_l^+(n)} x_{al}^k - \sum_{a \in A_l^-(n)} x_{al}^k = w_n^k \qquad \forall l \in L, \quad \forall n \in N_l, \quad \forall k \in K_l \tag{4.2}$$

$$\sum_{k \in K_l} d^k x_{al}^k \leq u_{al} y_{al} \qquad \forall l \in L, \quad \forall a \in A_l \tag{4.3}$$

$$(x, y) \in (X, Y)_{ll'} \qquad \forall (l, l') \in C \tag{4.4}$$

$$x \in X \tag{4.5}$$

$$y \in Y \tag{4.6}$$

The objective of the MLND model, (4.1), is to minimize the total routing and design cost. Constraints (4.2) are the usual *flow conservation equations*, ensuring that the demands are routed from their origins to their destinations in each layer, where $w_n^k = 1$ if $n = O(k)$, $w_n^k = -1$ if $n = D(k)$, and 0 otherwise. The *flow capacity constraints* (4.3), ensure that the sum of the flows on each arc $a \in A_l$ in layer $l \in L$ does not exceed its flow capacity u_{al}.

Constraints (4.5) and (4.6) define side constraints and the domains of the decision variables. There are several side constraints that can be added to a network design problem, among which *design balance* and *budget constraints* are the most important ones. Design balance constraints arise when, at each node, the number of incoming opened arcs (representing, for example, resources or vehicles) must be equal to the number of outgoing opened arcs. A budget constraint limits the cost for building the whole network to a total budget.

Constraints (4.4) are a set of *coupling constraints* that, for each connectivity requirement (l, l'), link together the domains of the decision variables (x, y) of layer l' to those of layer l. Several types of coupling constraints are encountered in the literature, depending on the applications.

In particular, *design capacity constraints* ensure that, for each arc $a \in A_l$, the number

of selected arcs supported by a in layer l', represented by set $B^{al}_{l'}$, does not exceed its design capacity v_{al}:

$$y_{al} \leq \sum_{b \in B^{al}_{l'}} y_{bl'} \leq v_{al} y_{al} \qquad \forall (l,l') \in C, \quad \forall a \in A_l. \tag{4.7}$$

The left inequality of (4.7) ensures that if we open arc a in supporting layer l, then at least one of its supported arcs has to be opened in the supported layer l'.

A second type of coupling constraints is the *multilayer all-design linking constraints*:

$$y_{bl'} \leq y_{al} \qquad \forall (l,l') \in C, \quad \forall a \in A_l, \quad \forall b \in B^{al}_{l'}. \tag{4.8}$$

These constraints simply mean that, to open arc $b \in A_{l'}$, all its supporting arcs have to be opened in all supporting layers l. Such constraints arise, for example, in the integrated freight rail service network design problem, where to open a block, all its supporting services have to be opened in the service layer.

A third type of coupling constraints is the *multilayer min-design linking constraints*:

$$y_{bl'} \leq \sum_{a \in D^{l}_{bl'}} y_{al} \qquad \forall (l,l') \in C, \quad \forall b \in A_{l'}. \tag{4.9}$$

These constraints ensure that, for each arc b in a supported layer l', at least one arc has to be opened in the supporting layer $l \in L$. Note that, constraints (4.8) implies (4.9), therefore, in general, one of them might be included in the formulation.

In addition to the above design connectivity coupling constraints, flow connectivity requirements between layers, whenever they are needed, might be added to the formulation. A first type of flow connectivity requirements is the *flow accumulation constraints*:

$$x^k_{al} = \sum_{b \in B^{al}_{l'}} x^k_{bl'} \qquad \forall (l,l') \in C, \quad \forall a \in A_l, \quad \forall k \in K. \tag{4.10}$$

These constraints simply mean that the flow on each arc a in layer l is equal to the flow on all the arcs in layer l' supported by arc a.

Note that, in some particular cases, constraints (4.10) might contradict flow conservation constraints (4.2). An example is when an arc in layer l supports two or more *reachable arcs* in layer l'. Two arcs (x_1, y_1) and (x_2, y_2) are said to be reachable if there is a path from x_1 to y_2 or from x_2 to y_1. Consider Figure 4.2 as an example where arcs a and b are supported, respectively, by paths $(1,2)$ and $(4,6,2,7)$ in layer l. Suppose that there is a path between arcs a and b (the dashed arc in layer l'), i.e., arcs a and b are reachable. Arc 2 in layer l supports both arcs a and b in layer l'. Suppose that a commodity with demand d has to be routed from node A to node B using arcs a and b, as well as the path between these two arcs (dashed arc). Based on equation (4.10), the flow on arc 2 is equal to the summation of the flows on arcs a and b, which is $2d$. If we consider the destination node of arc 2, its total incoming flow is $2d$, but its total outgoing flow (on arc 7) is d. So equations (4.10) contradict flow conservation constraints (4.10). Therefore, constraints (4.2) are not correct on general networks.

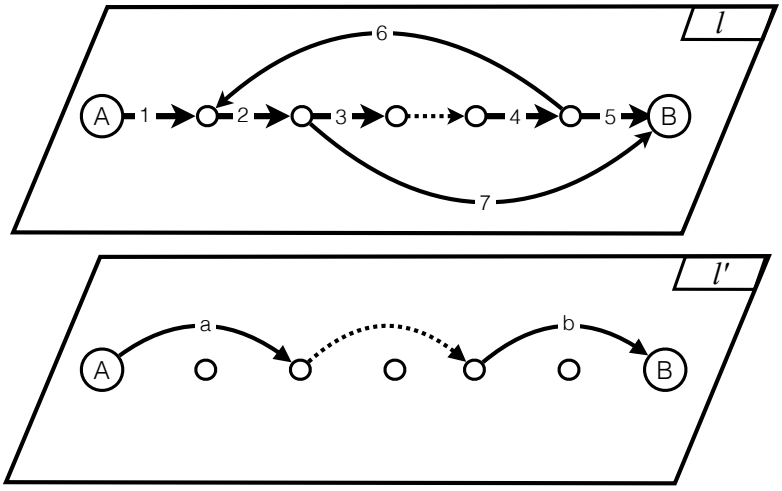

Figure 4.2 – Multilayer network design example showing flow connectivity constraints contradict flow conservation constraints

This issue does not arise in time-space networks, such as those used in Zhu et al. [88]. In this type of network, since the arcs are pointing to the next planning horizon, it is not possible that an arc supports two reachable arcs in another layer.

Another form of flow connectivity requirements might exist when flows are non-bifurcated, which means the flow of each commodity has to be routed through a single

path from its origin to its destination. In this case, the following *non-bifurcated flow connectivity constraints*, might be added to the model:

$$\sum_{k \in K_{l'}} x^k_{bl'} \leq \sum_{k \in K_l} x^k_{al} \qquad \forall (l, l') \in C, \quad \forall a \in A_l, \quad \forall b \in B^{al}_{l'}. \tag{4.11}$$

For each arc $a \in A_l$ and $b \in B^{al}_{l'}$, these constraints state that the flow of commodities $K_{l'}$ can move on arc b in layer l' only if there is a flow of commodities K_l on arc a in layer l.

4.3 Multilayer Network Design Taxonomy

In this section, we propose a classification of multilayer network design problems, and we provide an overview of the existing literature, in light of the proposed classification.

4.3.1 Multilayer Network Design Classification

Multilayer network design problems can be categorized into different classes based on three main dimensions. The first dimension is the number of layers. The second dimension is the degree of connectivity between layers that includes *one-to-one connectivity* and *one-to-many connectivity*. A one-to-one connectivity exists when each layer is supporting or is supported by only one other layer. A one-to-many connectivity exists when at least one of the layers is supporting or is supported by more than one layer. Note that, for two-layer network design problems, the only possible degree of connectivity is the one-to-one connectivity.

The last dimension is the type of connectivity that includes design connectivity, flow connectivity, and *flow-design connectivity*. The last term is used when both types of connectivity requirements present at the same time. In the integrated freight rail service network design problem, for example, not only the designs of the layers are connected, but also the flow of each service is equal to the summation of the flows on its supported blocks. Figure 4.3 illustrates these three dimensions.

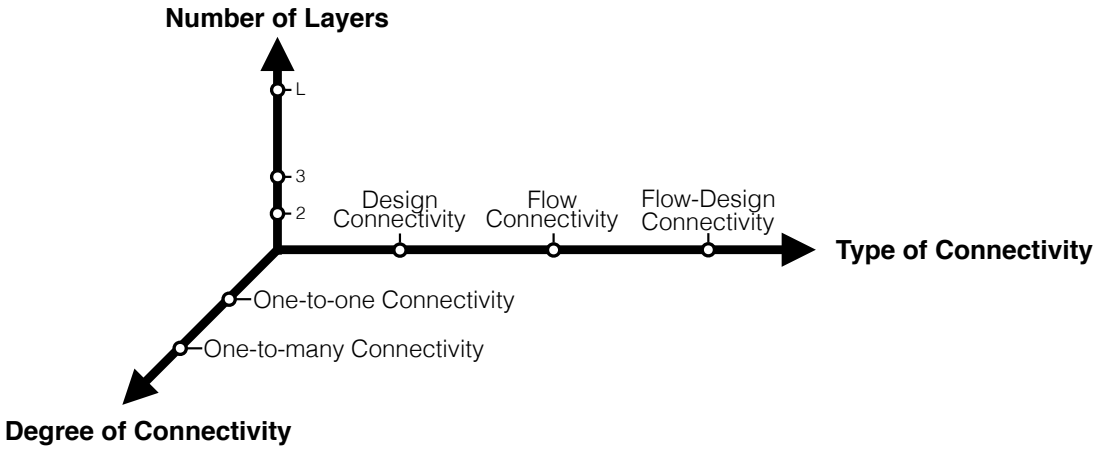

Figure 4.3 – Classification dimensions of multilayer network design problems

4.3.2 Overview of the Literature

Given the proposed taxonomy, the existing literature fits into 4 classes only: 1) two-layer network design problems with design connectivity; 2) two-layer network design problems with flow connectivity; 3) three-layer network design problems with one-to-many flow connectivity; 4) three-layer network design problems with one-to-one flow-design connectivity. There are some works in the literature on telecommunications applications that introduce multilayer network design models with L arbitrary layers [61, 74], but they focus exclusively on two-layer applications.

The first category, two-layer network design problems with design connectivity, includes the *service network design with resource management* [31, 32] and most of the telecommunications applications [8, 12, 33, 38, 61, 62, 66, 67, 73, 79]. The second one, two-layer network design problems with flow connectivity, consists of the *integrated crew scheduling and aircraft routing problem* [10, 16, 17, 68, 69, 81, 83]. The third class, three-layer network design problems with one-to-many flow connectivity, includes the *integrated crew pairing and assignment problem* proposed in Zeighami and Soumis [87]. The only research work that falls into the forth category, three-layer network design problems with one-to-one flow-design connectivity, is the *integrated rail freight service network design problem* proposed by Zhu et al. [88].

Figure 4.4 summarizes all the proposed multilayer network design problems in the

literature. This figure shows that the only paper that considers one-to-many connectivity is Zeighami and Soumis [87]. It also shows that almost all contributions in the literature consider two-layer network design problems with flow or design connectivity. The only paper that considers both flow and design connectivity requirements is Zhu et al. [88].

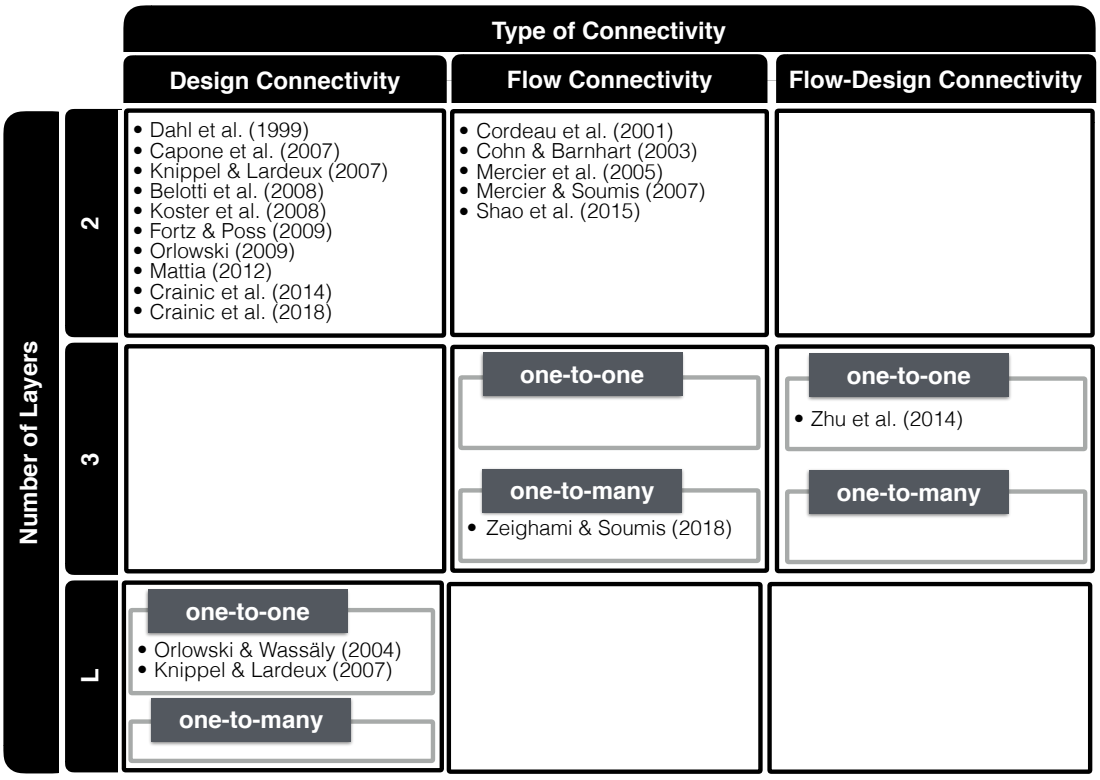

Figure 4.4 – Classification of existing multilayer network design problems in the literature

The next three sections present a comprehensive survey on the proposed multilayer network design models in the fields of transportation and telecommunications for each of the three main classes of problems identified above.

4.4 Two-Layer Network Design Problems with Design Connectivity

In the following subsections, we review the service network design problem with resource management and the telecommunications applications, the two main classes of

problems that fall into the category of two-layer network design problems with design connectivity.

4.4.1 Service Network Design with Resource Management

Service network design with resource management is the only application in the literature of transportation that falls into the class of two-layer network design problems with design connectivity. In the following subsections, we first define the problem and review the existing literature, then we show how the proposed general modeling framework can model this problem.

4.4.1.1 Problem Definition and Literature Review

Service network design models are broadly used in the field of transportation to formulate tactical planning problems. Most of these models assume the necessary resources (such as crews, power units, specific vehicles) are available at each terminal when needed. To overcome this simplification, researchers proposed models recognizing resource management aspects in service network design (see, e.g., [3, 4]). Crainic et al. [31] enlarged the considered range of resource management issues. The required resources to perform services are considered to be assigned to the terminals to which they must ultimately return, and a limited number of resources are assigned to each terminal. The problem is to select services in the time-space network and to route the commodities on the selected services, while the services have to be supported by appropriate resource cycles in the resource layer, and the total cost of the designed network, resource assignment and routing has to be minimized.

To model the problem, the authors proposed a two-layer network including a time-space service network and a time-space resource cycle network. In the time-space service network, an arc is a service moving between terminals and times. In the time-space resource cycle network, an arc is defined as a resource cycle, which corresponds to a path of services from a terminal in time t to the same terminal in time $t + TMAX$, where $TMAX$ is the maximum schedule length. There is a design connectivity constraint in the

problem where each arc (a resource cycle) in the resource layer corresponds to a cycle of the supported services in the service layer. We illustrate these notions in Figure 4.5, where cycles $r1$ and $r2$ in the resource layer support the sets of services $\{s1, s2, s3\}$ and $\{s4, s5, s6\}$ in the service layer, respectively. Using the explained two-layer network, the authors proposed a cycle-based model to formulate the problem.

Crainic et al. [31] assumed that there is only one type of resources and that the assignment of resources to terminals has been determined a priori. Crainic et al. [32] extended this research in two ways: considering multiple types of resources, as well as the strategic decision of fleet acquisition and assignment. The problem has two layers and includes assignment or location decisions in the resource layer.

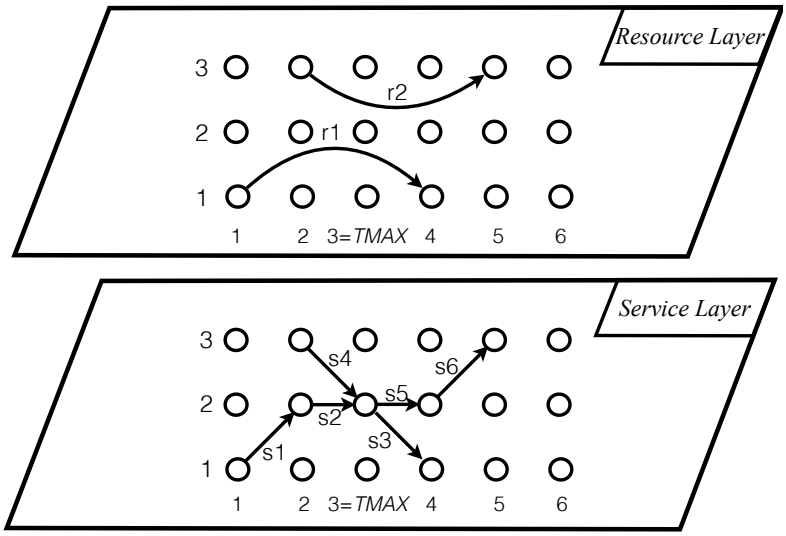

Figure 4.5 – Resource cycles and their supported services in the service network design problem with resource management

4.4.1.2 MLND Formulation

To formulate the service network design problem with resource management with the general model presented in Section 4.2.2, we use the problem description in Crainic et al. [31]. We denote by K the set of commodities (single flow-type) and by d^k the demand for commodity $k \in K$. In this problem, we have a service layer ($l = 1$) and a resource layer ($l = 2$). The connectivity set C is defined as $\{(2,1)\}$, meaning that the

service layer is supported by the resource layer. To open an arc $b \in A_1$ in the service layer, one of the supporting resource arcs in set D^2_{b1} should be opened in the resource layer. Let V be a set of terminals, then θ_v is the set of resource arcs of layer 2 that depart from terminal $v \in V$ during the scheduling length. We also denote by h_v the limit on the number of resources that depart at each terminal $v \in V$. Let f_{al} be the fixed cost of each arc $a \in A_l, l \in L$ that has to be paid to open the corresponding service or resource arc. We also denote by c^k_{a1} the routing cost of commodity $k \in K$ on arc $a \in A_1$. For the service layer, the decision variable x^k_{a1} determines the flow of each commodity $k \in K$ on each arc $a \in A_1$. Let y_{a1} and y_{a2} be the design variables of service $a \in A_1$ and resource $a \in A_2$, respectively. Then, the problem can be formulated as follows:

$$\min \quad \sum_{k \in K} \sum_{a \in A_1} c^k_{a1} x^k_{a1} + \sum_{l \in L} \sum_{a \in A_l} f_{al} y_{al} \tag{4.12}$$

subject to

$$\sum_{a \in A_1^+(n)} x^k_{a1} - \sum_{a \in A_1^-(n)} x^k_{a1} = w^k_n \qquad \forall n \in N_1, \quad \forall k \in K \tag{4.13}$$

$$\sum_{k \in K} d^k x^k_{a1} \leq u_{a1} y_{a1} \qquad \forall l \in L, \quad \forall a \in A_1 \tag{4.14}$$

$$y_{b1} \leq \sum_{a \in D^2_{b1}} y_{a2} \qquad \forall b \in A_1 \tag{4.15}$$

$$\sum_{a \in \theta_v} y_{a2} \leq h_v \qquad \forall v \in V \tag{4.16}$$

$$x^k_{a1} \geq 0 \qquad \forall a \in A_1, \quad \forall k \in K \tag{4.17}$$

$$y_{al} \in \{0,1\} \qquad \forall l \in L, \quad a \in A_l \tag{4.18}$$

The objective function, (4.12), is to minimize the summation of the routing costs of the service layer and the design costs of the service and resource layers. Constraints (4.13) are the flow conservation equations ensuring the demands are satisfied in the service layer. *Service flow capacity constraints* (4.14) ensure that the total flow on each service arc is less than or equal to the flow capacity of the service and that the service must be open in order to route the commodities. *Service-resource coupling constraints*

(4.15) show that, to open a service arc, at least one of the resource arcs should be open in the resource layer. Constraints (4.13), (4.14) and (4.15) are equivalent to constraints (4.2), (4.3) and (4.9) of the general modeling framework, respectively. *Terminal resource capacity constraints* (4.16) are the side constraints that impose a limit on the number of resources of layer 2 that depart from terminal $v \in V$ during the scheduling length. Constraints (4.17) and (4.18) define the domains of the decision variables.

4.4.2 Telecommunications Applications

In this subsection, we first provide an overview of the multilayer network design problems in telecommunications and a survey on the related literature. Then, we describe how a typical telecommunications application can be modeled using the proposed general modeling framework.

4.4.2.1 Problem Definition and Literature Review

In telecommunications applications, there are generally two layers: a *virtual layer* (also called *logical layer*) and a *physical layer* (or optical transport network). There is a set of identical nodes that are duplicated in both layers. These nodes might represent switch points. The nodes and the links can have several features: design cost, flow cost, *virtual flow capacity*, *physical design capacity*, and *node capacity*. The virtual flow capacity limits the flow of commodities on each logical link. The physical design capacity limits the number of logical links from the logical layer that can be supported by a physical link. The node capacity limits the number of virtual or physical links that can originate from, or end to, a particular node.

A set of commodities with specific demand quantities need to be routed in the logical layer. A network has to be designed in the logical layer to transfer the commodities and satisfy their demands. Several links have to be opened, or facilities have to be installed, between different pairs of nodes to design the network. Opening a link in the logical layer depends on opening a path in the physical layer. A typical example of two-layer network in telecommunications is an internet backbone network that has to be designed

based on a physical fiber network. A chain of links (a path) has to be opened in the physical layer to establish a connection in the internet network. Figure 4.6 shows a simple example of a two-layer network in telecommunications applications. Link a1 in the virtual layer corresponds to links b1 and b2 in the physical layer, and link a2 corresponds to links b3, b4, and b5. To open or install facilities on a link in the logical layer, all corresponding links in the physical layer have to be opened or need to have appropriate facilities. For example, to open link a2, all arcs b3, b4 and b5 have to be opened.

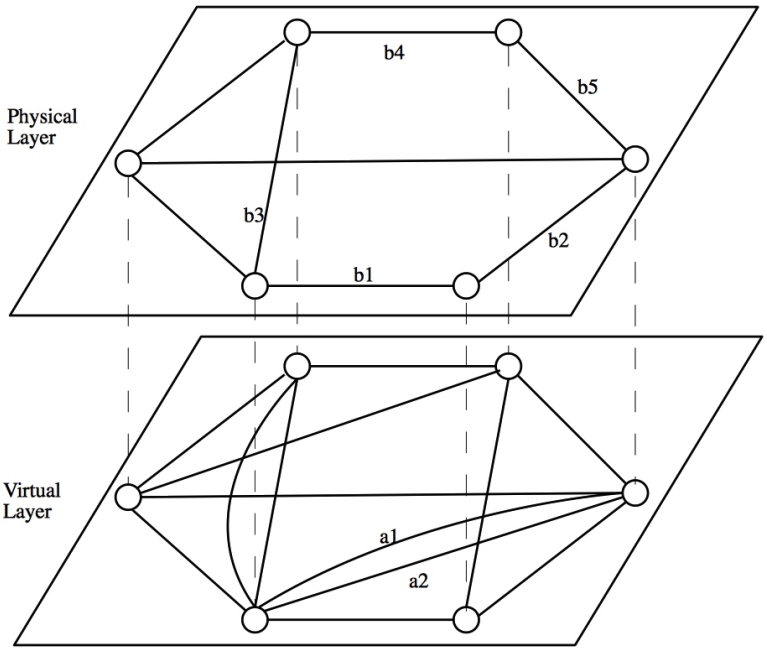

Figure 4.6 – A two-layer telecommunications network

In telecommunications applications, the researchers mostly proposed two-layer networks. All telecommunications applications also fall into the design connectivity category where the design of each link in the virtual layer corresponds to the design of all corresponding links in the physical layer. To the best of our knowledge, the concept of layered networks in telecommunications date backs to 1991 which is illustrated in Balakrishnan et al. [6]. Dahl et al. [33] proposed a two-layer network for a telecommunications application. The problem is known as the *PIPE*, where the objective is to find a minimum cost pipe (virtual links) selection and routing while considering the design ca-

pacity of the physical links. Note that, in this problem, each demand has to be routed on a single virtual path (demands are not splittable). Therefore, the routing and the design variables are binary in the proposed formulation.

Capone et al. [12] proposed a model for a two-layer network design problem with node capacity and *multicast traffic* demand where instead of point-to-point commodities, each commodity has an origin and multiple destinations. Therefore, a flow solution for each commodity is a tree, not a path.

Knippel and Lardeux [61] proposed a two-layer network design formulation, as well as a model with L arbitrary layers. The problem has fixed costs for the virtual and physical arcs, while no flow costs are considered. The model minimizes the total design cost of both layers. Parallel arcs are not used in the virtual layer; instead, the authors assumed each virtual flow capacity could be routed on several physical paths. Therefore, two types of continuous variables are introduced to determine the amount of each commodity on each logical path, and the amount of each installed logical traffic routed on each physical path. *Metric inequalities* are developed from the dual of the path-based formulation to represent the feasible space of capacity vectors.

Koster et al. [62] proposed a formulation for a problem with a predefined set of logical links. The problem includes the selection of nodes and the *survivability requirements* against physical node and link failures. According to survivability requirements, a particular set of demands should be satisfied even if there is any single physical node or link failure. In the proposed formulation, the survivability requirements are presented using *survivability constraints*, where the demands are doubled, and the flow through an intermediate node is restricted to half of the demand value.

Mattia [66] proposed a model that is similar to the one proposed in Knippel and Lardeux [61] where the goal is to install minimum cost integer capacities on the links of both layers to route the commodities on them. In addition, survivability conditions are added to ensure that in every failure scenario the routing of the associated commodities must be guaranteed. Instead of adding the survivability constraints, several failure scenarios are defined in each of which just a restricted number of links are available. For each failure scenario, a two-layer network is defined containing only the available links,

with variables defined for each scenario.

There are other contributions in the literature on telecommunications applications that focus mostly on solution methods [8, 38, 67, 73, 75, 79]. We review these papers in Section 4.8.

4.4.2.2 MLND Formulation

We use the problem description in Dahl et al. [33] as a representative of a telecommunications application that can be formulated using the general modeling framework presented in Section 4.2.2. We denote by $L = \{1,2\}$ the set of layers including the virtual layer ($l = 1$) and the physical layer ($l = 2$). Let $C = \{(2,1)\}$ be the set of connectivity requirements, where the ordered pair $(2,1)$ means that the physical layer is supporting the design of the virtual layer. We denote by K the set of single flow-type commodities to be routed on the virtual layer and by d^k the demand of each commodity $k \in K$. Let c^k_{a1} be the flow cost of routing one unit of commodity $k \in K$ on arc $a \in A_1$, and f_{al} be the fixed cost of opening an arc $a \in A_l, l \in L$. The binary flow variable x^k_{a1} determines if the demand of commodity $k \in K$ flows on the virtual arc $a \in A_1$. The binary design variables y_{a1} and y_{a2} determine, respectively, if a virtual arc $a \in A_1$ and if a physical arc $a \in A_2$ is open. Using the above notation, the problem can be formulated as follows:

$$\min \quad \sum_{k\in K}\sum_{a\in A_1} c^k_{a1}x^k_{a1} + \sum_{l\in L}\sum_{a\in A_l} f_{al}y_{al} \tag{4.19}$$

subject to

$$\sum_{a\in A_1^+(n)} x^k_{a1} - \sum_{a\in A_1^-(n)} x^k_{a1} = w^k_n \qquad \forall n \in N_1, \quad \forall k \in K \tag{4.20}$$

$$\sum_{k\in K} d^k x^k_{a1} \le u_{a1}y_{a1} \qquad \forall l \in L, \quad \forall a \in A_1 \tag{4.21}$$

$$y_{a2} \le \sum_{b\in B_1^{a2}} y_{b1} \le v_{a2}y_{a2} \qquad \forall a \in A_2 \tag{4.22}$$

$$x^k_{a1} \in \{0,1\} \qquad \forall a \in A_1, \quad \forall k \in K \tag{4.23}$$

$$y_{al} \in \{0,1\} \qquad \forall a \in A_l \tag{4.24}$$

The objective function (4.19) is to minimize the total cost including the total routing cost of the virtual layer and the summation of the design costs of the virtual and physical layers. Constraints (4.20) are the flow conservation equations that ensure the demands are satisfied in the virtual layer. Flow capacity constraints (4.21) are imposed for the virtual layer. Design capacity constraints (4.22) are coupling constraints ensuring that, to open an arc in the virtual layer the supporting arcs in the physical layer should be open and that the maximum number of selected virtual arcs is limited to the design capacity of the corresponding physical arc. These constraints correspond to the design capacity constraints of the general modeling framework (4.7). Constraints (4.23) and (4.24) define the feasible domains of the decision variables. In telecommunications applications, the flow variables are binary, to ensure that the flow of each commodity follows a single path from the origin to the destination. Note that, when the links are undirected in a telecommunications application, the problem can still be modeled using the proposed formulation by replacing an undirected link with two directed arcs.

4.5 Two-Layer Network Design Problems with Flow Connectivity

The integrated crew scheduling and aircraft routing problem is, to the best of our knowledge, the only application that falls into the category of two-layer network design problems with flow connectivity. In Subsection 4.5.1, we provide an overview of the problem, and we review the literature on this topic. In Subsection 4.5.2, we explain how the model we propose in Section 4.2.2 can formulate this application.

4.5.1 Integrated Crew Scheduling and Aircraft Routing: Literature Review

The first step in the airline planning process is *flight scheduling* to define the origin and the destination, as well as the departure and arrival times, for each flight leg to be flown during a given period. The next step is to assign an aircraft type to each flight leg to maximize the profit, which is called the *fleet assignment problem*. Then, for each aircraft type, the *aircraft routing problem* is solved to determine the sequence of flight legs that have to be covered by each aircraft. In this problem, each flight leg has to be

covered exactly once while ensuring aircraft maintenance requirements. The next step is called the *crew scheduling problem*, which consists in two steps: *crew pairings* followed by *crew assignment*. A crew pairing is a sequence of duty and rest periods starting and ending at the same location called crew base. A duty period is a sequence of flight legs separated by short rest periods. The duties are also separated by long (overnight) rest periods. The crew assignment is to build monthly scheduling out of generated pairings for each crew member.

Traditionally, most airlines use a sequential procedure to solve these problems. The sequential procedure reduces the complexity of the problem, but might result in a solution far from the global optimum of the integrated problem. The integrated crew scheduling and aircraft routing problem is a an attempt to handle this issue. The problem is defined on a two-layer network including 1) an aircraft routing network, and 2) a crew scheduling network. In both networks, each node corresponds to a flight leg, and the arcs represent the connections between two legs. The integrated problem is to find the minimum total cost of aircraft and crew routing (one path for each aircraft and one path for each crew), while the two following conditions are satisfied: 1) each flight leg has to be covered only once by a crew and only once by an aircraft, and 2) if a connection time for a link is too short (*short time links*) then the corresponding legs can be covered by the same crew only if both legs are covered by the same aircraft; otherwise, the connection time is insufficient for the crew. The second condition corresponds to the second type of flow connectivity requirements shown in constraints (4.11).

Cordeau et al. [17] proposed a path-based formulation for the integrated crew scheduling and aircraft routing problem. Cohn and Barnhart [16] contributed to the literature on the integrated aircraft routing and crew scheduling problem by proposing an extended crew pairing formulation. In the proposed formulation, the aircraft routing variables represent a complete solution of a routing problem. Mercier et al. [69] improved the integrated approach of Cordeau et al. [17] by introducing *restricted connection arcs* in addition to short time connections. A connection is restricted if the connection time is larger than a minimum threshold, but it is still smaller than another given threshold. If the two legs of such a connection are covered by the same crew, a penalty is imposed in

the objective function if both legs of this connection are not covered by the same aircraft. Mercier and Soumis [68] extended the integrated approach by adding the *flight re-timing* feature, where flight legs have different possible departure times among which the best one has to be selected to minimize the cost.

Shao et al. [83] integrated the fleet assignment problem to the integrated crew scheduling and aircraft routing problem. This new integration adds the decision of assigning a proper fleet type to each flight leg (node) of the network.

Salazar-González [81] proposed an arc-based formulation where both aircraft routes and crew pairs are presented using arc-based variables. The main disadvantage of the proposed formulation is that a large number of inequalities need to be added to avoid infeasible crew routes. Two alternative formulations are also proposed in Cacchiani and Salazar-González [10] including a path-based model as in Cordeau et al. [17] and an arc-path-based model using arc-based variables and path-based variables to represent aircraft routes and crew pairings, respectively.

4.5.2 MLND Formulation for Integrated Crew Scheduling and Aircraft Routing

Using the description of Cordeau et al. [17] as a representative of integrated crew scheduling and aircraft routing problems, we describe how to formulate the problem using the general modeling framework presented in Section 4.2.2. Let $L = \{1,2\}$ be the set of layers including a crew layer ($l = 1$) and an aircraft layer ($l = 2$). The nodes in both layers are the flight legs, while the arcs are the crew and aircraft connections, respectively, in the crew and aircraft layers. A set of crews (K_1) and aircrafts (K_2) should be routed on the crew and the aircraft layers, respectively. Note that the flows are non-bifurcated. We partition the nodes of each layer $l \in L$ into $N_l^O = \{n \in N_l \,|\, \exists k \in K_l, n = O(k)\}$, $N_l^D = \{n \in N_l \,|\, \exists k \in K_l, n = D(k)\}$ and $N_l^I = N_l \backslash N_l^O \cup N_l^D$. Binary decision variable x_{a1}^k and x_{a2}^k determine, respectively, if crew $k \in K_1$ uses arc $a \in A_1$ and if aircraft $k \in K_2$ uses arc $a \in A_2$. The set C is defined as $\{(2,1)\}$ representing the coupling constraints which indicates the aircraft layer supports the crew layer. Using the described notation, the problem can be formulated as:

$$\min \quad \Psi(x) \tag{4.25}$$

$$\sum_{a \in A_1^+(n)} x_{a1}^k - \sum_{a \in A_1^-(n)} x_{a1}^k = w_n^k \qquad \forall n \in N_1, \quad \forall k \in K_1 \tag{4.26}$$

$$\sum_{a \in A_2^+(n)} x_{a2}^k - \sum_{a \in A_2^-(n)} x_{a2}^k = w_n^k \qquad \forall n \in N_2, \quad \forall k \in K_2 \tag{4.27}$$

$$\sum_{k \in K_1} x_{b1}^k \leq \sum_{k \in K_2} x_{a2}^k \qquad \forall a \in A_2, \quad \forall b \in B_1^{a2} \tag{4.28}$$

$$\sum_{a \in A_l^+(n)} \sum_{k \in K_l} x_{al}^k = 1 \qquad \forall l \in L, \quad \forall n \in N_l^O \cup N_l^I \tag{4.29}$$

$$\sum_{a \in A_l^-(n)} \sum_{k \in K_l} x_{al}^k = 1 \qquad \forall l \in L, \quad \forall n \in N_l^D \cup N_l^I \tag{4.30}$$

$$x_{al}^k \in \{0,1\} \qquad \forall a \in A_l, \quad \forall k \in K \tag{4.31}$$

The objective function (4.25) minimizes the total routing costs on both layers. In airline applications, since the objective is typically a non-linear function of arc-based flow variables, researchers usually propose path-based formulations for which the objective function is linear. Crew flow conservation equations (4.26) and aircraft flow conservation equations (4.26) guarantee, respectively, the routing of the flows of the crews and aircrafts on the crew and aircraft layers. Constraints (4.28) ensure that a crew does not change aircraft when the connection time is too short. These constraints correspond to the flow connectivity inequalities (4.10) of the general modeling framework. Constraints (4.29) and (4.30) are the side constraints ensuring that a flight leg is covered by exactly one crew and one aircraft. Constraints (4.31) define the domain of the decision variables.

4.6 Three-Layer Network Design Problems with One-to-Many Flow Connectivity

To the best of our knowledge, the only paper that considers one-to-many connectivity requirements is Zeighami and Soumis [87]. In the following subsections, we describe the integrated crew pairing and assignment problem, and then we explain how the MLND framework can formulate this problem.

4.6.1 Integrated Crew Pairing and Assignment: Literature Review

The crew scheduling problem constructs individual schedules for a set of available crew members. Because of its complexity, this problem is usually solved in two steps: crew pairing followed by crew assignment. The sequential procedure reduces the complexity of the problem but might result in a solution far from the global optimum of the integrated problem since the schedule constraints and objectives are not taken into account during the construction of the pairings.

Zeighami and Soumis [87] proposed an integrated crew pairing and personalized assignment problem for a given set of pilots and copilots. They considered a set of vacation requests (VRs) for each pilot and copilot each month. The problem is defined on a three-layer network including 1) a crew pairing network, 2) a crew pilot assignment network, and 3) a crew copilot assignment network. In the crew pairing network, each node corresponds to a departure and an arrival stations of the flights. The arcs represent the flights and the connections between the flights. In the pilot (copilot) assignment network, each node corresponds to the start and end of pairings, and the arcs represent the pairings, the connections between pairings, and the vacations of the pilots (copilots). The objective function finds a trade-off between maximizing the number of satisfied VRs and minimizing the total cost of the pairings (one path for each pairing and one path for each crew pilot (copilot) assignment), while the two main conditions are satisfied: 1) each flight is covered by exactly one pairing, and 2) each pairing is covered by exactly one pilot and one copilot.

4.6.2 MLND Formulation for Integrated Crew Pairing and Assignment

To model the problem using the general modeling framework presented in Section 4.2.2, we define the following notation. Let $L = \{1,2,3\}$ be the set of layers including the crew pairing ($l = 1$), pilot assignment ($l = 2$), and copilot assignment ($l = 3$) layers. $G_l = (N_l, A_l)$ defines the network of each layer $l \in L$. In the crew pairing layer, we partition the arcs into the sets of flight arcs A_1^f and connection arcs A_1^c. We assume that all the potential pairings are exists in the pilot and copilot assignment layers. There

are a set of pilot K_2 and copilots K_3 that are need to be routed in the pilot and copilot assignment layers, respectively. $O(k) \in N_l$ and $D(k) \in N_l$ are, respectively, the origin and the destination of each crew $k \in K_l$ in layers $l \in \{2,3\}$.

We denote by $C = \{(1,2),(1,3)\}$ the set of connectivity requirements, where $(1,2)$ and $(1,3)$ means that, respectively, the pilot and copilot assignment layers are supported by the crew pairing layer. To use a pairing arc in the assignment layer, all the corresponding arcs need to be selected in the crew pairing layer. Let B_2^{a1} and B_3^{a1} be the sets of arcs (pairings), respectively, in the pilot and copilot layers supported by arc $a \in A_1$ in the pairing layer. Binary flow variable y_{a1} determines whether arc $a \in A_1$ is selected or not. Binary flow variables x_{a2}^k and x_{a3}^k determine, respectively, whether or not pilot $k \in K_2$ and copilot $k \in K_3$ selects arc $a \in A_2$ and $a \in A_3$. Using the described notation, the problem is formulated as follows:

$$\min \quad \Psi(x,y) \tag{4.32}$$

$$\sum_{a \in A_2^+(n)} x_{a2}^k - \sum_{a \in A_2^-(n)} x_{a2}^k = w_n^k \qquad \forall n \in N_2, \quad \forall k \in K_2 \tag{4.33}$$

$$\sum_{a \in A_3^+(n)} x_{a3}^k - \sum_{a \in A_3^-(n)} x_{a3}^k = w_n^k \qquad \forall n \in N_3, \quad \forall k \in K_3 \tag{4.34}$$

$$\sum_{k \in K_2} x_{b2}^k \leq y_{a1} \qquad \forall a \in A_1, \quad \forall b \in B_2^{a1} \tag{4.35}$$

$$\sum_{k \in K_3} x_{b3}^k \leq y_{a1} \qquad \forall a \in A_1, \quad \forall b \in B_3^{a1} \tag{4.36}$$

$$\sum_{b \in B_2^{a1}} \sum_{k \in K_2} x_{b2}^k = 1 \qquad \forall a \in A_1^f \tag{4.37}$$

$$\sum_{b \in B_3^{a1}} \sum_{k \in K_3} x_{b3}^k = 1 \qquad \forall a \in A_1^f \tag{4.38}$$

$$x_{al}^k \in \{0,1\} \qquad \forall l \in \{2,3\}, \quad \forall a \in A_l, \quad \forall k \in K_l \tag{4.39}$$

$$y_{a1} \in \{0,1\} \qquad \forall a \in A_1 \tag{4.40}$$

$$x \in X \tag{4.41}$$

The objective function (4.32) minimizes the total routing and design costs on the

three layers. Pilot flow conservation equations (4.33) guarantee the routing of each pilot $k \in K_2$ in the second layer. The same type of flow conservation constraints (4.34) exist for the copilot layer. In this equations $w_n^k = 1$ if $n = O(k)$, $w_n^k = -1$ if $n = D(k)$, and 0 otherwise. Constraints (4.35) and (4.36) are the coupling constraints ensuring the flow connectivity between layers. Constraints (4.37) and (4.38) are the covering constraints which ensure, each flight arc is covered by exactly one pairing. Constraints (4.39) and (4.40) define the domain of the decision variables. Constraints (4.41) are a set of side constraints including the constraints which are correspond to VRs.

4.7 Three-Layer Network Design Problems with One-to-One Flow-Design Connectivity

The only paper that applies both flow and design connectivity requirements is Zhu et al. [88]. In the following subsections, we first describe the problem proposed in Zhu et al. [88]. Then, in the second subsection, we explain how the MLND framework can model this problem.

4.7.1 Integrated Rail Freight Service Network Design: Literature Review

Zhu et al. [88] proposed a three-layer network to model a problem in rail freight transportation planning where typically a double consolidation policy is performed. First, cars are grouped into so-called blocks, and then the blocks are grouped into services to make up trains. Cars that are in a terminal at the same time can be sorted and arranged into a block. This process is called blocking, and its goal is to reduce operations in terminals by moving blocks instead of each car individually. A block is then a unit that will be transferred between terminals using a sequence of services until it reaches its destination. At the block destination, the block is broken down, and the cars that arrived at their destinations are delivered, while the cars that did not reach their destinations are grouped into other blocks. The process of arranging the blocks into services is known as train makeup. The next step is to select the services and define their frequencies.

In Zhu et al. [88], both blocking and service selection problems are considered to-

gether. To do so, a three-layer network including car network, block network, and service network is used. The network of each layer is a time-space network. The first layer, the car layer, consists of *car-waiting arcs*, *classification arcs* and *car-holding arcs*. Car-waiting arcs show the waiting of cars between two time periods in one terminal. Classification arcs show the classification process of cars between two time periods in one terminal. Car-holding arcs show waiting process of the classified cars between two time periods in one terminal. The car-waiting, classification, and car-holding arcs are not related to any path of the block layer. The second layer, the block layer, includes block arcs from the origins to the destinations of the blocks to support car movements. The third layer, the service layer, includes service arcs to support block movements.

In the car layer, the flows of commodities are moved via the car arcs and the projected block arcs from the block layer. A chain of services has to be opened in the service layer (design connectivity) to open a block arc. The flow of each service is equal to the summation of the flows on all its supported blocks (flow connectivity). The problem is to find a minimum cost blocking and service design, and flow routing of the cars, while considering flow capacity of the blocks and the services, blocking capacity of each terminal, and flow and design connectivity requirements between the layers.

4.7.2 MLND Formulation for Integrated Rail Freight Service Network Design

We define the following notation to model the problem using the general modeling framework presented in Section 4.2.2. Let K be a set of single flow-type commodities. There are three layers, the car layer ($l = 1$), the block layer ($l = 2$) and the service layer ($l = 3$). The car layer includes car arcs and block arcs projected from the block layer. The block layer includes block holding, transfer and moving arcs. The service layer consists of service waiting and moving arcs. We denote by $C = \{(3,2),(2,1)\}$ the set of connectivity requirements, where $(2,1)$ means that the block layer is supporting the car layer, and $(3,2)$ indicates that the service layer is supporting the block layer. In the car layer, each projected block arc is supported by a path of block holding, transfer and moving arcs in the block layer. In the block layer, a block moving arc is supported by a chain of service moving and waiting arcs of the service layer. For each layer $l \in L$,

continuous flow variable x^k_{al} determines the flow of commodity $k \in K$ on arc $a \in A_l$. For the car layer, the binary design variable y_{a1} determines the selection of a car arc or a projected block arc $a \in A_1$. In the block ($l = 2$) and service ($l = 3$) layers, the binary design variable y_{al} is 1 if arc $a \in A_l$ is selected. We denote by T, V and E the set of time periods, yards, and track segments, respectively. Let $H(v,t)$ be the set of blocks built simultaneously at yard $v \in V$, and let $S(e,t)$ be the set of services moved simultaneously at track segment $e \in E$. We denote by h_v and s_e, respectively, the maximum number of blocks and services that can be built simultaneously at each yard $v \in V$ and track segment $e \in E$. Using the described notation, the problem is formulated as follows:

$$\min \quad \sum_{l\in L}\sum_{k\in K}\sum_{a\in A_l} c^k_{al}x^k_{al} + \sum_{l\in L}\sum_{a\in A_l} f_{al}y_{al} \tag{4.42}$$

$$\sum_{a\in A_1^+(n)} x^k_{a1} - \sum_{a\in A_1^-(n)} x^k_{a1} = w^k_n \qquad \forall n \in N_1, \quad \forall k \in K \tag{4.43}$$

$$x^k_{al} = \sum_{b\in B^{al}_{l'}} x^k_{bl'} \qquad \forall (l,l') \in C, \quad \forall a \in A_l, \quad \forall k \in K. \tag{4.44}$$

$$\sum_{k\in K_l} d^k x^k_{al} \le u_{al}y_{al} \qquad \forall l \in L, \quad \forall a \in A_l \tag{4.45}$$

$$y_{a3} \le \sum_{b\in B^{a3}_2} y_{b2} \le v_{a3}y_{a3} \qquad \forall a \in A_3 \tag{4.46}$$

$$y_{bl'} \le y_{al} \qquad \forall (l,l') \in C, \quad \forall a \in A_l, \quad \forall b \in B^{al}_{l'}. \tag{4.47}$$

$$\sum_{a\in H(v,t)} y_{a2} \le h_v \qquad \forall v \in V, \quad \forall t \in T \tag{4.48}$$

$$\sum_{a\in S(e,t)} y_{a3} \le s_e \qquad \forall e \in E, \quad \forall t \in T \tag{4.49}$$

$$x^k_{al} \ge 0 \qquad \forall l \in L, \quad \forall a \in A_l, \quad \forall k \in K \tag{4.50}$$

$$y_{al} \in \{0,1\} \qquad \forall l \in L, \quad \forall a \in A_l \tag{4.51}$$

The objective function (4.42) minimizes the total routing and design costs on the three layers. Constraints (4.43) guarantee that the demands are routed on the car layer, while constraints (4.44) compute the flow on each arc of the block and service layers

based on the corresponding arcs of the car layer. Flow capacity constraints (4.45) ensure that the flow on each arc is less than or equal to the capacity and that the arc should be opened in order to route the commodities. Constraints (4.47) ensure that, to open an arc in the car layer, all the corresponding block arcs must be opened in the block layer, and that, to open an arc in the block layer, all the corresponding arcs must be opened in the service layer. Design capacity constraints (4.46) limit the number of blocks that can be moved on the corresponding arc of the service layer. These constraints correspond to the design capacity constraints (4.7) of the general modeling framework. Constraints (4.48) and (4.49) are the side constraints that limit, respectively, the number of blocks and services to be created at each yard and track segment.

4.8 Solution Approaches for Multilayer Network Design Problems

In this section, we summarize the solution methods proposed in the literature for multilayer network design problems. The solution methods proposed in the literature can be classified as 1) exact solution methods [10, 17, 33, 38, 61, 62, 66, 67, 69, 79, 83], and 2) heuristic solution methods [8, 12, 31, 32, 73, 75, 81, 88]. In the two following subsections, we review, respectively, the exact and heuristic methods proposed for solving multilayer network design problems.

4.8.1 Exact Solution Methods

In exact solution methods, researchers mostly focus on either embedding Benders decomposition and classical network design cuts into branch-and-bound or combining Benders decomposition and column generation. In the two following subsections, we review: 1) cutting plane and Benders decomposition methods [33, 38, 61, 62, 66, 67, 73, 75, 79], and 2) combined column generation and Benders decomposition methods [10, 17, 69, 83].

4.8.1.1 Cutting Plane and Benders Decomposition Methods

Dahl et al. [33] used a branch-and-cut algorithm for a two-layer telecommunications network design problem. At each node of the branch-and-bound tree, several valid inequalities are added to the linear programming (LP) relaxation. A variable fixing heuristic is called when the branch-and-cut algorithm does not find any violated inequality. Several instances derived from a real-world application are used to test the algorithm. The algorithm is able to solve to optimality all instances with no fixed costs, without making any branching (these instances are solved at the root). For the instances with positive fixed costs, the algorithm could not solve the problems to optimality but found solutions within an average optimality gap of 9%.

Knippel and Lardeux [61] proposed a Benders decomposition algorithm for a two-layer telecommunications network design problem with no flow cost where the master problem handles the design variables, and two subproblems determine the value of the flow variables. The algorithm first solves the master problem as an integer program with no cuts. Then, the algorithm checks the feasibility of the obtained solution by solving two subproblems, one for the logical layer and the other for the physical layer. If the solution is infeasible, then the corresponding cuts are added to the master problem. The solution time of the master problem is larger than that of the subproblems because of the integrality conditions on the master problem variables. To reduce the solution time, different approaches for the generation of cuts are proposed.

Fortz and Poss [38] improved the Benders decomposition method proposed in Knippel and Lardeux [61]. The idea is to embed the Benders decomposition into a branch-and-cut algorithm. In this way, instead of solving the master problem as an integer program, the algorithm solves the LP relaxation of the master problem at each node of the branch-and-bound tree. If the solution is integral, it adds the corresponding Benders cuts by solving the subproblem. If the solution is not integral, it generates branches in the tree and adds the branching constraints to the master problem. This way, the solution time is significantly reduced compared to that of the tradiitonal Benders decomposition approach. To avoid generating too many infeasible nodes, cuts are added a priori to

the master problem by solving its LP relaxation with the cutting plane algorithm proposed by Knippel and Lardeux [61]. The results are compared with the cutting plane approaches of Knippel and Lardeux [61], and with CPLEX. The results show that the algorithm outperforms CPLEX, and that the proposed branch-and-cut algorithm is always faster than the Benders decomposition approaches.

Koster et al. [62] proposed a cutting plane method embedded in a branch-and-bound framework for the telecommunications application with survivability requirements against physical node and link failures. In the proposed cutting plane method, the authors used the cuts that were applied before in the literature on the single layer network design problem. The algorithm is tested on three different sets of instances. In the case of unprotected demands (no survivability requirement), the cutting plane algorithm significantly improves the LP relaxation lower bounds. In the case of protected demand, where the size of the problem dramatically increases compared to the unprotected case, the cutting plane algorithm only slightly improves the LP relaxation lower bounds.

Raack and Koster [79] studied a packing problem derived from a two-layer network design problem. The authors proved the NP-hardness of the problem and defined two classes of facet-defining inequalities that generalize the well-known cutset inequalities to two-layer network design.

Mattia [66] proposed a branch-and-cut scheme using metric inequalities for the same problem as in Knippel and Lardeux [61], but with survivability requirements. Further work on the same problem can be found in Mattia [67], where the polyhedron of a two-layer network design formulation is considered. The results are extended to multilayer network design models.

4.8.1.2 Combined Column Generation and Benders Decomposition Methods

Orlowski [73] proposed several techniques combining Benders decomposition and column generation embedded into a branch-and-cut framework for a multilayer network design problem with survivability requirements. The solution methods are based on the approaches proposed in Fortz and Poss [38] and Koster et al. [62]. Similar to Fortz and Poss [38], metric inequalities are used to create a Benders master problem by project-

ing out the flow variables to a subproblem. The LP relaxation of the Benders master problem is solved at each node of the branch-and-cut tree. Whenever an integer solution is found, the algorithm checks the feasibility in a routing subproblem, and adds cuts to the master problem, if the design is not feasible according to the routing constraints. The cuts proposed in Koster et al. [62] are generated, along with cuts derived from survivability constraints. Several primal heuristics are proposed to improve the feasible integer solutions. A column generation approach is developed to generate flow variables dynamically in the large-scale routing subproblems. The algorithm could find feasible solutions and lower bounds for large-scale instances that could not be solved by a commercial solver. However, for large and dense instances, the obtained feasible solutions are still far from optimality (57% and 28% optimality gap on average for the instances with and without the survivability conditions, respectively).

Orlowski et al. [75] combined the approaches in Orlowski [73] and Koster et al. [62]. In addition to the cutting plane approach presented in Koster et al. [62], the heuristic methods presented in Orlowski [73] were also used. The heuristics are called at various places of the branch-and-cut tree. The algorithm is tested on several real-world industrial instances.

Cordeau et al. [17] proposed a Benders decomposition approach for the integrated crew scheduling and aircraft routing problem. The classical solution method is to use branch-and-bound where, at each node, the LP relaxation is solved using a column generation method. However, in the computational experiments, the authors observed that the column generation restricted master problem becomes difficult to solve because it has too many constraints. Therefore, a Benders decomposition approach is proposed where the crew scheduling variables are projected out into a subproblem. The Benders decomposition approach is embedded into a branch-and-bound algorithm. The LP relaxation at each node is solved by Benders decomposition, and both the Benders master problem and the Benders subproblem are solved using column generation. The solution method is tested on instances derived from real data. The proposed solution method is compared with a pure column generation approach. The results show that the combined Benders decomposition and column generation approach produces integer solutions faster than

the pure column generation method. The obtained solutions are compared with the solutions of the traditional sequential planning process. The results show that the integrated approach produced significant savings in comparison to the traditional sequential one.

Two Benders decomposition methods are proposed and compared in Mercier et al. [69] for the integrated crew scheduling and aircraft routing problem with restricted connection arcs. The methods are based on the combined Benders decomposition and column generation method proposed in Cordeau et al. [17]. The first decomposition considers the aircraft routing problem as the master problem and the crew scheduling problem as the subproblem, like it is done in Cordeau et al. [17], while in the second one, the decomposition is reversed. The effect of Pareto-optimal cuts on the convergence of these two Benders decomposition approaches is analyzed. The results show that Pareto-optimal cuts accelerate the convergence of Benders decomposition. The results also show that the second decomposition outperforms the one proposed in Cordeau et al. [17].

Shao et al. [83] proposed a combined Benders decomposition and column generation approach for the integrated crew scheduling and aircraft routing problem with fleet assignment. The Benders decomposition approach is enhanced by several acceleration techniques. In addition, a stabilization technique is used for the column generation procedure of the crew pairing subproblem. The proposed Benders decomposition approach is tested on real-world data obtained from a U.S.-based airline carrier. The results show that the integrated approach yields 8.4% improvement, on average, in comparison with a traditional sequential decision process.

Cacchiani and Salazar-González [10] proposed two different exact methods for path-based and arc-based formulations of the integrated crew scheduling and aircraft routing problem. Both solution methods include three main steps: 1) solving the LP relaxation of the corresponding model to optimality using column generation on the path-based model and consequently finding a lower bound; 2) running a primal heuristic to obtain an upper bound; 3) finding the optimal solution. The third step for the path-based formulation consists of a branch-and-price algorithm, while for the arc-path-based model, a restricted mixed-integer program is used. A single bounding cut that significantly accelerates the

solution process is also presented. The proposed algorithms are tested on real-world instances. The results show that the proposed method for the arc-path-based model outperforms not only the one proposed for the path-based model, but also the heuristic method proposed in Salazar-González [81].

Zeighami and Soumis [87] proposed a solution methodology based on Benders decomposition and column generation for the integrated crew pairing and assignment problem. The pairings are generated by the Benders master problem. The monthly schedules for pilots and copilots are generated by the Benders subproblems. Benders master problem and Benders subproblems are solved by column generation.

4.8.2 Heuristic Solution Methods

For the heuristic solution methods, researchers mostly focus either on combining heuristic and mathematical programming approaches to come up with matheuristics [8, 31, 32, 81, 88], or on developing neighborhood-based heuristics [12]. The two following subsections are dedicated to reviewing these two types of heuristic methods.

4.8.2.1 Matheuristics

Crainic et al. [31] proposed a matheuristic solution method for the service network design problem with resource management. Their proposed solution method can be described in two main phases. The first phase is to solve the LP relaxation of the original problem, and the second one is an iterative *slope scaling* procedure. The idea of slope scaling is to iteratively solve a linear approximation of the formulation, and to use the resulting flow distribution to adjust the fixed cost approximation at the next iteration. When the slope scaling stalls, a perturbation procedure changes the initial linearization factors to start a new phase of the slope scaling procedure (see, e.g., [27, 56]). In the proposed solution method, the LP relaxation is solved by a column generation approach that generates cycles dynamically. The linearization factors of the slope scaling procedure are then initialized using the information from the LP relaxation phase. On a set of small-size instances, the results show that the algorithm produces high-quality solutions

in comparison with a commercial MIP solver, as it only generates a small fraction of the possible cycles. The algorithm is also benchmarked on a set of large-scale instances against a column generation-based heuristic developed by the authors. The results show that the algorithm again can produce high-quality solutions.

Crainic et al. [32] proposed a matheuristic that extends solution techniques proposes in Crainic et al. [31]. The matheuristic applies column generation to determine the set of resource cycles. To produce feasible solutions, the solution method uses both slope scaling and different matheuristics such as Archetti et al. [5], De Franceschi et al. [34], Hewitt et al. [52], Vu et al. [85]. The results show that the proposed approach outperforms both a commercial MIP solver and the heuristic method in Crainic et al. [31].

Zhu et al. [88] proposed a matheuristic solution method based on slope scaling for the integrated rail freight service network design problem. Two approaches are proposed: a basic approach that linearizes all design variables and a dynamic approach, where the service design variables are linearized and a metaheuristic is used to generate the blocks dynamically. The approaches are tested on instances based on the setting of the main-line network of a major North American railroad. The results show that the optimality gap obtained by CPLEX increased dramatically with instance size, and that CPLEX is unable to find an optimal solution within 10 hours of CPU time. The basic approach obtained better solutions than CPLEX for more than 90% of the instances. The dynamic approach also outperformed CPLEX, but in comparison with the basic one, it performed somewhat worse on small and medium size instances due to the additional effort required by the block generation feature. However, it starts outperforming the basic approach when the instance dimensions grow.

Salazar-González [81] proposed a two-phase matheuristic for the integrated crew scheduling and aircraft routing problem. The first phase is a greedy search to find the pairings to cover all the flights, and the second phase creates aircraft routes.

Belotti et al. [8] proposed a Lagrangian relaxation method for a multilayer network design problem in telecommunications. A Lagrangian relaxation is used to relax the virtual flow capacity constraints. In this way, there is no relation between the flow variables and the design variables in the relaxed problem. Therefore, the relaxed problem

is decomposed into shortest path subproblems for each commodity, plus one capacity assignment subproblem to determine the design variables. Since the number of virtual arcs increases exponentially when increasing the problem size, a column generation approach is used to solve the capacity assignment subproblem. A subgradient method is applied to find the Lagrangian lower bound. To find feasible solutions for the problem, a local search heuristic is developed. It starts with an initial solution and tries to improve it by rerouting the commodities on the virtual links and rerouting the virtual capacities on the physical links. The proposed Lagrangian relaxation method is able to find both lower and upper bounds for almost all the tested instances in a reasonable time.

4.8.2.2 Neighborhood-Based Heuristics

Capone et al. [12] proposed a heuristic for a two-layer telecommunications network design problem with node capacity and multicast traffic. The heuristic constructs an initial solution using a greedy-based procedure. Then, it improves the solution using two different neighbourhood structures including 1) changing the routing of the commodities in the virtual layer, and 2) re-routing the virtual links on the physical links. The results show that the algorithm performs better with the second neighbourhood structure.

4.9 Conclusions

We have proposed a taxonomy of multilayer network design problems using different features including the number of layers as well as the type and degree of connectivity requirements between layers. We have also presented a state-of-the-art review on multilayer network design models and methods found in telecommunications and transportation applications. The review shows that there is one research contribution in the literature that considers one-to-many connectivity Zeighami and Soumis [87]. The review also shows that almost all contributions in the literature considered two-layer network design problems with design or flow connectivity. The only contribution that considers both flow and design connectivity requirements is Zhu et al. [88]. Studying new problems with flow-design connectivity and with more than two layers is a fascinating

research direction.

From the solution methodology point of view, a first interesting research avenue is to adapt to multilayer network design the advanced solution methods proposed for single layer network design problems. In particular, Lagrangian relaxation methods have been used to solve single layer network design problems (see [53], [82], [27], and [60] for example), but, as we have seen, they have been very rarely applied to address multilayer network design problems (the only exception is the work of Belotti et al. [8]). Another interesting research avenue is to take advantage of the multilayer network design structure to derive new valid inequalities to be used in cutting plane methods. Finally, because of their inherent difficulty, multilayer network design problems can be solved by exact solution methods only for relatively small instances. Solving large-scale instances requires a combination of decomposition methods (cutting planes, Benders decomposition, column generation, Lagrangian relaxation) and metaheuristics. The development of matheuristics capable of solving multilayer network design problems of increasing complexity constitutes a major research challenge.

CHAPTER 5

A LAGRANGIAN-BASED MATHEURISTIC FOR MULTILAYER SINGLE FLOW-TYPE MULTICOMMODITY CAPACITATED FIXED-CHARGE NETWORK DESIGN

In multilayer network design, there are several networks, each at a given layer. To route a set of commodities from the origins to the destinations, at each layer, appropriate capacitated links have to be opened by paying a fixed cost per link. The so-called design connectivity constraints need to be enforced. They impose the requirement that a link is opened in a layer only if a chain of supporting links is opened in another layer. The problem is to find a minimum cost design and routing for all layers that satisfies capacity and connectivity constraints as well as demands. In this paper, we address the Multilayer Single flow-type Multicommodity Capacitated Fixed-Charge Network Design problem (MSMCFND), where commodities are routed only in the first layer. The amount of flow on each link in a particular layer is equal to the amount of flow on its supported links in the first layer. We propose a general formulation for the MSMCFND, for which we develop an effective solution method based on Lagrangian relaxation to address large-scale instances. The resulting Lagrangian-based matheuristic method is enhanced by intensification, diversification, and post-optimization procedures using long-term and short-term memories. The results show that the proposed algorithm is competitive with (and often significantly better than) a state-of-the-art MIP solver on a large set of randomly generated instances, not only with respect to the obtained upper bounds, but also in terms of optimality gaps.

5.1 Introduction

Applications of network design models are extensive in the fields of transportation, telecommunications, logistics and production planning [64, 70]. In network design, given a potential network with capacitated links, several commodities such as goods,

data or people, have to be routed between the different origin and destination points. A network has to be constructed by opening appropriate links to route the commodities. A *fixed design cost* has to be paid to open a link. In addition to the fixed design costs, a per unit routing cost, or *flow variable cost*, is associated with each link and must be paid to route each unit of commodity demand on the link. The problem is to select a set of links and route the demands from the origins to the destinations on the constructed network such that the costs are minimized and capacities are respected.

In this paper, we consider multilayer network design models, with applications in the fields of transportation [17, 31, 88] and telecommunications [33, 61]. Multilayer network design models differ from a typical single-layer network design model in two main aspects. First, instead of a single network, there are several networks, each at a given layer. Each network has its nodes, potential links with capacities, and possibly commodities. To route the demands of the commodities from the origins to the destinations, appropriate links have to be opened by paying a fixed cost per link. Second, in multilayer network design, there are *design connectivity* and *flow connectivity* requirements between the layers. Design connectivity means that opening a link in a layer depends on opening a chain of supporting links (a path) in another layer, named as supporting layer. Flow connectivity states that the amount of flow on each link in a particular layer depends on the flow on its supported links in another layer. It is possible that some layers have no commodities to route, but these layers also support the routing of other layers through constraints that connect the layers. In particular, when there is only one layer where commodities are routed, the problem is a *multilayer single flow-type network design*.

In this paper, we address the Multilayer Single flow-type Multicommodity Capacitated Fixed-Charge Network Design problem (MSMCFND), where commodities are routed only in layer $l = 1$, given a set of layers $L = \{1, 2, .., |L|\}$. The amount of flow on each link in a particular layer $l \geq 2$ is equal to the amount of flow on its supported links in layer $l - 1$ and consequently to the amount of flow on its supported links in layer $l = 1$. A comprehensive review and taxonomy on multilayer network design models and solution methods for such problems can be found in [1]. The MSMCFND represents a set of service network design problems with multiple consolidation levels, which arise

in transportation applications, see Zhu et al. [88] and Crainic et al. [31] for example.

Lagrangian relaxation is among the most effective methods to address the single-layer multicommodity capacitated fixed-charge network design problem (MCFND), see [53], [82], [27], and [60] for example. Motivated by these developments, we propose a Lagrangian-based matheuristic for the MSMCFND. Although there are effective Lagrangian-based heuristics for the MCFND, these methods cannot be adapted in a straightforward way to the MSMCFND. Indeed, the challenge in multilayer network design is how to handle the significant additional complexity incurred by the design connectivity constraints between the layers. On the one hand, these additional constraints complicate the task of developing Lagrangian relaxations that provide a good trade off between the quality of the lower bound and the computational effort of solving the Lagrangian subproblem. On the other hand, the derivation of effective feasible solutions becomes significantly more difficult in the presence of these constraints.

We propose a Lagrangian-based matheuristic solution method based on a *slope scaling* scheme. The idea of slope scaling is to iteratively solve a linear multicommodity flow formulation, and to use the flow distribution to adjust the linear approximation at the next iteration. When the slope scaling method stalls, a perturbation move changes the initial linear approximation to start a new slope scaling procedure; for more details on slope scaling, see [27] and [56]. Note that [88] also used a slope scaling procedure but it was not integrated in a Lagrangian relaxation scheme, as proposed in this paper. In particular, our Lagrangian relaxation approach provides not only the lower bounds to assess the quality of the obtained feasible solutions, but also guides the slope scaling search by providing initial solutions and by defining the perturbation moves.

Our main contribution, apart from presenting a general formulation for the MSMCFND, is to propose an effective solution method based on Lagrangian relaxation to solve large-scale MSMCFND instances. In the proposed algorithm, a subgradient method is used to find lower bounds. A primal heuristic procedure based on slope scaling is developed to find upper bounds using the design information derived from the Lagrangian subproblem. The best upper bound is then used to guide the subgradient method. We are comparing our results to those obtained by a state-of-the-art mixed-integer programming

(MIP) solver. We observe that the algorithm produces better results not only in terms of the upper bounds (for 90% of the instances) but also in terms of the optimality gaps (for 60% of the instances).

The paper is organized as follows. We state the problem and the mathematical formulation in Section 5.2. In Section 5.3, we explain how to compute the lower and upper bounds through Lagrangian relaxation and slope scaling. Section 5.4 summarizes the algorithm that combines Lagrangian relaxation and slope scaling. In Section 5.5, we present the numerical results on randomly generated instances. In Section 5.6, we summarize this work and propose future research directions.

5.2 Problem Statement and Formulation

In this section, we first formally define the problem. We then present a formulation for the MSMCFND. Finally, we describe a preprocessing procedure that reduces the flow capacities.

5.2.1 Problem Definition

In the MSMCFND, multiple networks are considered, one per layer. Given a set of layers $L = \{1, 2, ..., |L|\}$, each layer $l \in L$ consists of nodes and potential links. There is a fixed design cost to open a link in each layer, while the flow costs are defined only for the links of layer $l = 1$, called the *base layer*, for which there are commodities with specific demands that have to be routed between their origin and destination nodes. In the MSMCFND, there are two types of connectivity requirements between the layers: design connectivity and flow connectivity. The first one, design connectivity, means that a link opened in the base layer requires a chain of *supporting links* (a path) to be opened in layer 2. Similarly, a link opened in a given layer l requires a path to be opened in its upper layer, i.e., layer $l+1$. If link a of layer l requires a path (that includes links b and c) to be opened in layer $l+1$, then l is said to be a *supported* by $l+1$, and $l+1$ is said to be a *supporting* layer of l. In addition, a is *supported* by b and c, while b and c are *supporting* link a. Note that a link in a particular layer might be in more than one path; therefore,

it might support more than one link in its lower layer, which means that the fixed design cost of the link has to be paid as soon as one of the potential supported links of the lower layer is designed. Design connectivity requirements include *design capacity* constraints, which, for link b in layer l, limits the number of designed links in layer $l-1$ supported by link b. The second type of connectivity constraints, flow connectivity, imposes that the flow on link b in a given layer $l \geq 2$ is equal to the summation of the flows of all the links in the base layer supported by link b. In summary, in the MSMCFND, the goal is to determine a minimum cost design and routing in all layers to satisfy demands, while considering capacity constraints, as well as flow and design connectivity requirements.

One of the applications of the MSMCFND is the integrated rail freight service network design problem described in Zhu et al. [88]. They proposed a three-layer network to model the problem where a double consolidation policy is performed. First, cars are grouped into so-called blocks (blocking process). Then, they are grouped into services to make up trains (train make-up process). The next step is to select the services and define their schedules. To propose a formulation determining blocking, train make-up, and service selection decisions simultaneously, Zhu et al. [88] considered a three-layer network including: 1) a car layer that consists of links on which cars are moved in each terminal; 2) a block layer that includes block links from the origins to the destinations of the blocks to support car movements; and 3) a service layer that includes service links to support block movements. In the car layer, the flows of commodities are moved via the car links and the projected block links from the block layer. Appropriate block links have to be designed or opened to route the cars. Each block corresponds to a chain of services in the service layer that support the movement of the block. To design a block link, a chain of supporting services has to be opened in the service layer. The flow of each service is equal to the summation of the flows on all its supported blocks. The problem is to find a minimum cost blocking and service design, and flow routing of the cars to satisfy demands, while considering flow capacity of the blocks and services, blocking capacity of each terminal, and design connectivity of the block and service layers. Zhu et al. [88] proposed a mixed-integer programming (MIP) formulation for the problem. We generalize the problem setting and formulation to an arbitrary number of layers.

Furthermore, since we focus on the interaction between the layers, we do not consider application-specific requirements such as terminal block-building workload constraints.

5.2.2 Formulation

We are given a set of layers L and a network $G_l = (N, A_l)$ for each layer $l \in L$, where N and A_l are the sets of nodes (the same for all layers) and directed arcs for layer $l \in L$, respectively. The sets $A_l^+(n)$ and $A_l^-(n)$ represent, respectively, the outward and inward arcs of node $n \in N$ in layer $l \in L$. A set K of commodities have to be routed through the base layer network. The amount of each commodity $k \in K$ that must be routed from its origin $O(k) \in N$ to its destination $D(k) \in N$ is $d^k > 0$. We define $u_{al} > 0$ and v_{al}, a positive integer, as the flow capacity and the design capacity of arc $a \in A_l$ in layer $l \in L$, $c_{a1}^k \geq 0$ as the variable flow cost of commodity $k \in K$ on arc $a \in A_1$, and $f_{al} \geq 0$ as the fixed design cost of arc $a \in A_l$ of layer $l \in L$. We also use the following additional notation: for each $n \in N$ and $k \in K$, $w_n^k = d^k$, if $n = O(k)$, $w_n^k = -d^k$, if $n = D(k)$, and $w_n^k = 0$, if $n \neq O(k), D(k)$; for each $l \in L$, $a \in A_l$ and $k \in K$, $h_{al}^k = \min\{d^k, u_{al}\}$.

To model the connectivity constraints, we introduce the following notation: B^{al} is the set of arcs in layer $l-1$ that are supported by arc $a \in A_l$ for $l \geq 2$ and B_1^{al} is the set of arcs in layer 1 that are supported by arc $a \in A_l$, $l \in L$ (we assume that $B_1^{a1} = \{a\}$ for arc $a \in A_1$). Two sets of decision variables are introduced to formulate the problem: binary design variables and continuous flow variables. The design variable y_{al} is 1 if arc $a \in A_l$ is selected in layer $l \in L$, and 0 otherwise. The flow variable x_{a1}^k is the amount of flow of commodity $k \in K$ on arc $a \in A_1$. The MSMCFND model takes the following form:

$$\min \quad \sum_{a \in A_1} \sum_{k \in K} c_{a1}^k x_{a1}^k + \sum_{l \in L} \sum_{a \in A_l} f_{al} y_{al} \tag{5.1}$$

$$\sum_{a\in A_1^+(n)} x_{a1}^k - \sum_{a\in A_1^-(n)} x_{a1}^k = w_n^k \forall n \in N, \quad \forall k \in K \tag{5.2}$$

$$\sum_{k\in K}\sum_{b\in B_1^{al}} x_{b1}^k \leq u_{al} y_{al} \qquad \forall l \in L, \quad \forall a \in A_l \tag{5.3}$$

$$\sum_{b\in B_1^{al}} x_{b1}^k \leq h_{al}^k y_{al} \quad \forall l \in L, \quad \forall a \in A_l, \quad \forall k \in K \tag{5.4}$$

$$\sum_{b\in B^{al}} y_{b(l-1)} \leq v_{al} y_{al} \forall l \in L, \quad l \geq 2, \quad \forall a \in A_l \tag{5.5}$$

$$y_{b(l-1)} \leq y_{al} \forall l \in L, \quad l \geq 2, \quad \forall a \in A_l, \quad \forall b \in B^{al} \tag{5.6}$$

$$x_{al}^k \geq 0 \qquad \forall a \in A, \quad \forall k \in K, \quad \forall l \in L \tag{5.7}$$

$$y_{al} \in \{0,1\} \qquad \forall a \in A, \quad \forall l \in L \tag{5.8}$$

The objective of the model, (5.1), is to minimize the total routing and design costs. Constraints (5.2) are the usual *flow conservation equations*, which ensure that the demands are routed from the origins to the destinations in the base layer. The *flow capacity constraints* (5.3) ensure that the total flow on each arc $a \in A_l$ in layer $l \in L$, which is equal to the sum of the flows on the arcs supported by a in the base layer, does not exceed its flow capacity u_{al}. These constraints also impose that: 1) no commodity can be routed on a link unless it is opened; and 2) if there is some flow on an arc in a particular layer, then this flow contributes to the flow of its supporting arcs in the upper layer, and consequently the supporting arcs will be forced to be opened. The *strong linking constraints* (5.4) impose that no flow of any commodity can go through a closed arc. Although these inequalities are redundant for the MIP formulation, our tests show that they improve the linear programming (LP) and the Lagrangian relaxation lower bounds significantly. Similar inequalities have also been used in the literature on network design problems to improve the lower bounds; see [14] for example. The *design capacity constraints* (5.5) ensure that, for each arc $a \in A_l$, the number of designed arcs that are supported by a in layer $l-1$ does not exceed its design capacity v_{al}. *Upward design linking constraints* (5.6) ensure that an arc can be opened only if all its supporting arcs in the upper layer are opened. It is easy to see that these constraints are already satisfied by constraints (5.3) or by constraints (5.5). Although constraints (5.6) are redundant for the MIP formulation,

our tests show that they improve the LP and the Lagrangian relaxation lower bounds significantly. Constraints (5.7) and (5.8) define the domains of the variables.

5.2.3 Preprocessing Procedure

Using the multilayer structure of the MSMCFND, we can reduce the flow capacities in two ways. First, we know that each arc a in layer $l \leq |L|-1$ corresponds to a path in the upper layer. According to constraints (5.3), the total flow on a (hence, the capacity u_{al}) has to be less than or equal to the flow capacity of each corresponding arc in the upper layer:

$$u_{al} \leq \min_{b \in F_a^{l+1}} \{u_{b(l+1)}\} \qquad \forall l \in L, \quad l \leq |L|-1, \quad \forall a \in A_l, \tag{5.9}$$

where $F_a^{l+1} = \{b \in A_{l+1} | a \in B^{b(l+1)}\}$, i.e., F_a^{l+1} is the set of arcs in layer $l+1$ supporting arc a in layer l. Second, we know that any given arc a in layer $l \geq 2$ supports several arcs in the lower layer, and that at most v_{al} arcs can be supported by arc a. Therefore, the total flow on a (hence, its capacity u_{al}) has to be less than or equal to the sum of the flow capacities of the first v_{al} arcs in layer $l-1$ supported by a, where the arcs are sorted in non-increasing order of to their flow capacities:

$$u_{al} \leq \sum_{b \in \widetilde{B}^{al}} u_{b(l-1)} \qquad \forall l \in L, \quad l \geq 2, \quad \forall a \in A_l, \tag{5.10}$$

where $\widetilde{B}^{al}$ is the set containing the first v_{al} arcs supported by arc $a \in A_l$ in the lower layer.

Based on the above arguments, we derive a preprocessing procedure that: 1) scans each layer in descending order (from $|L|-1$ to 1) and reduces the flow capacities based on (5.9); 2) scans each layer in ascending order (from 2 to $|L|$) and reduces the flow capacities based on (5.10); and 3) if at least one flow capacity is reduced in 2), it goes back to 1).

5.3 Bounding Procedures

In this section we describe how to compute the lower and upper bounds through Lagrangian relaxation and slope scaling, respectively. The algorithm combining these bounding procedures is presented in the next section.

5.3.1 Computing Lower Bounds through Lagrangian Relaxation

Relaxing constraints (5.3) and (5.4) in a Lagrangian way, we obtain the following Lagrangian subproblem:

$$(\mathscr{P}_{\mathscr{LR}}(\alpha,\beta)) \quad Z_{LR}(\alpha,\beta) = \min \sum_{a\in A_1}\sum_{k\in K} c^k_{a1}x^k_{a1} + \sum_{l\in L}\sum_{a\in A_l} f_{al}y_{al} + \sum_{l\in L}\sum_{a\in A_l}\alpha_{al}\left(\sum_{k\in K}\sum_{b\in B^{al}_1} x^k_{b1} - u_{al}y_{al}\right) + \sum_{l\in L}\sum_{a\in A_l}\sum_{k\in K}\beta^k_{al}\left(\sum_{b\in B^{al}_1} x^k_{b1} - d^k_{al}y_{al}\right) \tag{5.11}$$

$$s.t.\,(5.2),(5.5),(5.6),(5.7),(5.8),$$

where $\alpha_{al} \geq 0$ and $\beta^k_{al} \geq 0$ are the Lagrange multipliers corresponding to constraints (5.3) and (5.4), respectively. The Lagrangian subproblem decomposes into $|K|+1$ subproblems: $|K|$ shortest path subproblems, $\mathscr{P}^k_{SP}(\alpha,\beta)$, one for each commodity $k \in K$, in the base layer, and a subproblem that depends only on the design variables, called the *design subproblem* and denoted $\mathscr{P}_D(\alpha,\beta)$. The cost of each arc $a \in A_1$ in each shortest path subproblem $\mathscr{P}^k_{SP}(\alpha,\beta)$, $k \in K$, is given by:

$$c^k_{a1} + \sum_{l\in L}\sum_{b\in F^l_{a1}}\left(\alpha_{bl} + \beta^k_{bl}\right) \geq 0, \tag{5.12}$$

where $F^l_{a1} = \left\{b \in A_l | a \in B^{bl}_1\right\}$. The shortest path subproblems are solved using Dijkstra algorithm.

The design subproblem is as follows:

$$(\mathscr{P}_D(\alpha,\beta)) \quad Z_{\mathscr{P}_D(\alpha,\beta)} = \min \quad \sum_{l\in L}\sum_{a\in A_l}\left(f_{al} - \alpha_{al}u_{al} - \sum_{k\in K}\beta^k_{al}d^k_{al}\right)y_{al} \qquad (5.13)$$

$$s.t.\,(5.5),(5.6),(5.8)$$

The design subproblem is to find a set of arcs that minimizes objective function (5.13) while satisfying design connectivity constraints, (5.5) and (5.6), as well as design integrality constraints (5.8). Before commenting on how we solve the design subproblem, we first ask ourselves whether or not it has the integrality property [46], i.e., if its LP relaxation always has an integral optimal solution.

Proposition 5.1. *The design subproblem does not have the integrality property.*

Proof. We define an instance with two layers where all design capacities are equal to 1. Then, the upward design linking constraints (5.6) are redundant for the LP relaxation of the design subproblem. Further, we assume that all design costs on layer $l = 2$ are equal to 0. As a result, all design variables for layer $l = 2$ assume value 1 in an optimal solution of the LP relaxation of the design subproblem. The resulting LP relaxation of the design subproblem reduces to the LP relaxation of the well-known set packing problem, which has fractional optimal solutions [76]. □

Corollary 5.1.

$$Z_{LD} = \max_{\alpha,\beta\geq 0} Z_{LR}(\alpha,\beta) \geq Z_{LP},$$

where Z_{LP} is the optimal value of the LP relaxation. There are instances where the inequality is strict.

This result illustrates a significant difference between the single-layer MCFND and the MSMCFND: when we relax the capacity and the strong linking contraints in the MCFND, the Lagrangian subproblem has the integrality property and the Lagrangian dual bound Z_{LD} is equal to the LP relaxation lower bound Z_{LP} [26].

In Section 5.4, where we describe the subgradient method for solving the Lagrangian dual, we see that the design subproblem is rarely solved to optimality: either we solve

its LP relaxation or we use a state-of-the-art MIP solver as a heuristic. In both cases, the design subproblem produces a design solution, which can be fractional when solving the LP relaxation, but is integral when using a MIP solver in a heuristic fashion. Note that, this approach allows us to keep the Lagrangian subproblem tractable, but might result in a lower bound that is inferior to Z_{LP}, even for instances where $Z_{LD} > Z_{LP}$.

5.3.2 Computing Upper Bounds through Slope Scaling

To obtain upper bounds, we develop a primal heuristic based on a slope scaling procedure, which is repetitively called during the subgradient method described in Section 5.4. The following *multilayer multicommodity capacitated network flow problem* (MMCNF) is solved at each iteration of the slope scaling procedure:

$$\min \quad \sum_{a \in A_1} \sum_{k \in K} \overline{c}^k_{a1} x^k_{a1} \tag{5.14}$$

$$\text{s.t.} \quad (5.2), (5.7) \quad \text{and}$$

$$\sum_{k \in K} \sum_{b \in B^{al}_1} x^k_{b1} \leq u_{al} \qquad \forall l \in L, \quad \forall a \in A_l, \tag{5.15}$$

where the linearized costs $\bar{c}$ are initialized, as follows, based on an input design solution $\widehat{y}$ (fractional or integral) of the design subproblem obtained when performing the subgradient method:

$$\overline{c}^k_{a1} = (c^k_{a1} + \rho^0_{a1})(1 + M(1 - \widehat{y}_{a1})) \qquad \forall a \in A_1, \forall k \in K, \tag{5.16}$$

where $\rho^0_{a1} = \sum_{l \in L} \sum_{b \in F^l_{a1}} f_{bl}/u_{bl}$, and M is a large positive value to avoid routing on closed arcs. When $\widehat{y}_{a1} = 1$, $a \in A_1$, the costs of the corresponding arcs are equal to the costs in the LP relaxation of the model obtained from the MSMCFND by removing constraints (5.4)-(5.6). If $\widehat{y}_{a1} = 0$, $a \in A_1$, then the costs are set to large positive values to avoid any flow on the corresponding arc. Note that, even if $\widehat{y}_{a1}$ is fractional, formula (5.16) can still be used: arcs with small fractional values are then assigned large costs.

After solving the MMCNF, a flow solution $\bar{x}$ is obtained and the linearized costs are updated using the following equation to trigger the next iteration:

$$\bar{c}^k_{a1} = \begin{cases} c^k_{a1} + \rho_{a1} & \text{if } \bar{x}^k_{a1} > 0 \\ \bar{c}^k_{a1} & \text{if } \bar{x}^k_{a1} = 0 \end{cases} \qquad \forall a \in A_1, \forall k \in K, \tag{5.17}$$

where $\rho_{a1} = (\sum_{l \in L} \sum_{b \in F^l_{a1}} f_{bl}) / \sum_{k \in K} \bar{x}^k_{a1}$. When $\bar{x}^k_{a1}$ is positive, this equation ensures that, at the next iteration, if the flow solution remains the same, the cost on each arc is equal to the fixed design cost plus the sum of the flow variable costs. If $\bar{x}^k_{a1} = 0$, we keep the same cost because it is large enough to avoid any flow. Let z_t and z_{t-1} be the current and the previous objective values of the MMCNF problem at each iteration t of the slope scaling procedure. The termination condition of the slope scaling procedure is either to obtain the same objective value for two successive iterations ($z_t = z_{t-1}$), as suggested in [56], or to reach a predefined maximum number of slope scaling iterations it^s_{max}.

At each slope scaling iteration, a feasible solution might be produced for the MSMCFND. Given the flow solution $\bar{x}$, a design solution $\bar{y}$ is obtained using the following rule:

$$\bar{y}_{al} = \begin{cases} 1 & \text{if } \sum_{k \in K} \sum_{b \in B^{al}_1} \bar{x}^k_{b1} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.18}$$

When the obtained design solution is feasible, i.e., it satisfies the design capacity constraints (5.5), then an upper bound is found for the original problem.

At the end of the slope scaling procedure, an *intensification* step is performed using up to n of the best feasible solutions obtained during the procedure, where n is a parameter typically set to a small value (less than 10). For each of these feasible solutions, a restricted MMCNF is created by: 1) fixing the design variables to their values in the solution, and 2) setting the costs to the original flow variable costs. The resulting MMCNF is solved in order to obtain an optimal flow distribution when the design variables are fixed.

If the initial MMCNF yields a feasible solution to the MSMCFND, the slope scaling procedure proceeds as described above. If, however, this is not the case, we solve the

following *auxiliary design subproblem* to obtain a feasible solution:

$$\min \quad \sum_{l \in L} \sum_{a \in \widehat{A}_l} f_{al} y_{al} \tag{5.19}$$

$$\text{s.t.} \quad (5.2), (5.3), (5.5), (5.7), (5.8),$$

where $\widehat{A}_l = \{a \in A | \widehat{y}_{al} = 0\}$. The solution to this auxiliary design subproblem is used as input design solution $\widehat{y}$, and the linearized costs are re-initialized using formula (5.16). The objective of the auxiliary design subproblem is to select a minimum cost design among the closed arcs of the initial design solution $\widehat{y}$ in such a way that all the constraints that define a feasible solution to the MSMCFND are satisfied. Note that the redundant constraints (5.4) and (5.6) are removed in the formulation of the auxiliary design subproblem. Since the resulting model has no commodity-dependent costs and capacities, we can reduce the number of commodities by aggregating the commodities with the same origin into a single commodity. This significantly reduces the computational complexity of solving the auxiliary design subproblem. Besides, we stop solving the auxiliary design subproblem after finding the very first feasible solution. Note that, solving the auxiliary design subproblem only guarantees the feasibility at the first iteration of the slope scaling procedure, but not at every iteration.

We provide a pseudo code for the primal heuristic in Algorithm 3. When a new upper bound is found, the best upper bound found so far (since the beginning of the subgradient method), UB^{best}, is updated if it is improved by the current solution. We also define the set of m best feasible solutions found so far, F, which is updated whenever a new solution is found, where m is a large value (we use 1000 in our experiments). The set F is a long-term memory that will be used for further improvements described in details in the next section.

Algorithm 3: Primal heuristic

Input design solution $\widehat{y}$
Initialize the linearized costs of MMCNF problem using $\widehat{y}$ and formula (5.16)
Solve MMCNF problem $\rightarrow \overline{x}, \overline{y}$
if $\overline{y}$ is feasible **then**
    Compute upper bound
    Update UB^{best} and feasible solution set F
else
    Solve auxiliary design subproblem $\rightarrow \widehat{y}$
    Re-initialize the linearized costs of MMCNF problem using $\widehat{y}$ and formula (5.16)
    Solve MMCNF problem $\rightarrow \overline{x}, \overline{y}$
end if
repeat
    Update objective function of MMCNF problem using $\overline{x}$ and formula (5.17)
    Solve MMCNF problem $\rightarrow \overline{x}, \overline{y}$
    if $\overline{y}$ is feasible **then**
        Compute upper bound
        Update UB^{best} and feasible solution set F
    end if
until the number of iterations is greater than or equal to it^{s}_{max} **or** $z_t = z_{t-1}$
Perform intensification

5.4 The Lagrangian-Based Matheuristic

The proposed algorithm consists of three main phases: 1) **initialization**, where the LP relaxation of the original problem, $\mathscr{P}_{LP}$, is solved to initialize the Lagrange multipliers; 2) **Lagrangian** phase, including a subgradient procedure to find lower bounds and the primal heuristic enhanced by *intensification* and *diversification* procedures to obtain upper bounds; 3) **post-optimization**, where a restricted MIP problem, $\mathscr{P}_{PO}$, is solved to improve the upper bounds. The algorithm is equipped with *long* and *short term memories* storing specific information obtained during the algorithm execution to run the post-optimization phase as well as the intensification and diversification procedures. In the following subsections, 5.4.1 to 5.4.3, we describe the three phases of the algorithm. Subsection 5.4.4 summarizes the algorithm.

5.4.1 Initialization Phase

The first step in the initialization phase is to solve $\mathscr{P}_{LP}$, the LP relaxation of the MSMCFND formulation (5.1)-(5.8), by a cutting plane method performed for a limited time t^{LP}, where the violated strong linking constraints (5.4) are added iteratively to the model. Solving $\mathscr{P}_{LP}$ helps to obtain good initial Lagrange multipliers and lower bound. The Lagrange multipliers are initialized to the values of the dual variables associated to constraints (5.3) and (5.4). Note that some of the strong linking constraints will not appear in the final LP relaxation of the cutting plane procedure. For these constraints, we initialize the corresponding Lagrange multipliers to zero. Preliminary tests show that adding the upward design linking constraints (5.6) increases significantly the difficulty of the LP relaxations. Therefore, taking into account the limited time to perform the cutting plane method, these constraints are not added to the LP relaxation.

In the next step of the initialization phase, a *graph traversal algorithm*, presented in [15], is performed to reduce the size of the shortest path subproblems. Starting from each node, the algorithm traverses the network twice, forward and backward, to find the arcs that are reachable for each commodity, i.e., the arcs that belong to at least one path between the origin and the destination of each commodity. We then restrict the network of each shortest path subproblem to these reachable arcs, thus significantly reduce the time to solve the shortest path computations.

At the end of the initialization phase, a slope scaling procedure is called using the final design solution of $\mathscr{P}_{LP}$ as the input design solution $\widehat{y}$. The goal is to obtain an initial upper bound to be used in the subgradient method described in the next subsection.

5.4.2 Lagrangian Phase

The Lagrangian dual takes the form of a non-differentiable optimization problem:

$$Z_{LD} = \max_{\mu \geq 0} Z_{LR}(\mu), \tag{5.20}$$

where $\mu = (\alpha, \beta)$. We solve the Lagrangian dual with a subgradient method. At each iteration i of the subgradient method, the current point, μ^i, is moved to a new point, μ^{i+1}, using the formula $\mu^{i+1} = \mu^i + t^i d^i$, where d^i and t^i are the direction and the step size, respectively. The following formula is used to compute the direction:

$$d^i = g^i + \theta^i d^{i-1}, \tag{5.21}$$

where g^i is a subgradient and $\theta^i \in (0,1)$ is a parameter adjusted by the following *modified Camerini-Fratta-Maffioli rule* [11]:

$$\theta^i = \begin{cases} ||g^i||/||d^{i-1}|| & \text{if } g^i d^{i-1} < 0 \\ 0 & \text{otherwise} \end{cases} \tag{5.22}$$

This rule is suggested in [26] for a subgradient method when the capacity and strong linking constraints are relaxed in the MCFND problem. It is also recommended in [26] to project the direction d^i in equation (5.21), meaning to set to 0 the negative components of d^i. In our preliminary tests, the suggested modified Camerini-Fratta-Maffioli rule worked well, while the results were not promising when we projected the direction in equation (5.21). Therefore, in our final implementation, we use the modified Camerini-Fratta-Maffioli rule without projecting the direction.

The step size is computed using the following formula:

$$t^i = \lambda^i \left(Z^e_{DL} - Z_{LR}\left(\mu^i\right) \right) / g^i d^i, \tag{5.23}$$

where Z^e_{DL} is an estimation of the value of Z_{DL}, and λ^i is a scaling factor.

The performance of a subgradient method depends on the values of its parameters, in particular, λ^i and Z^e_{DL} [26]. The parameter λ^i is initialized to λ^0 and is divided by a parameter γ_1 (typically, 2), whenever LB^{best} (the best known lower bound) is not improved for γ_2 consecutive iterations. The minimum value of λ^i is restricted to λ_{min}. In our algorithm, Z^e_{DL} is initially set to the upper bound provided by the slope scaling step of the initialization phase and is updated to the best upper bound found so far by the

primal heuristic, UB^{best}. The subgradient procedure is limited to a maximum number of iterations γ_3. For more details on the subgradient method see [26].

The subgradient method follows a *three-phase approach* when handling the design subproblem. During the first phase, performed for a limited number of subgradient iterations, it_{max}^{ph1}, the LP relaxation of $\mathscr{P}_D(\alpha,\beta)$ is solved by a cutting-plane method, where the upward design linking constraints (5.6) are added iteratively. During this phase, at the beginning of any given iteration of the subgradient method, the design subproblem contains the current set of added cuts, and only the costs are modified. Therefore, we run the primal simplex because the initial primal basis is feasible and only the optimality needs to be achieved. When we enter the cutting-plane procedure, we switch to the dual simplex, because the basis is optimal or dual feasible and only primal feasibility must be reached. During the second phase, performed for a limited number of subgradient iterations, it_{max}^{ph2}, the root node of the branch-and-bound tree of the IP design subproblem is solved. At the beginning of phase 2, the final LP formulation of the first phase (including all the cuts added so far) is converted to an IP model. Finally, during the third phase, performed for a limited number of subgradient iterations, it_{max}^{ph3}, the IP design subproblem is solved. The solution time of each IP problem is restricted to a time limit, t^{Des}. Since we care about the quality of the lower bound, we force the branch-and-bound tree to emphasize the search on finding the best lower bound using the parameters provided in the solver. After solving the Lagrangian subproblems, a lower bound is computed for the original problem, and the best lower bound found so far, LB^{best}, is updated, if a better lower bound is found.

At each iteration of the subgradient method, when a new design solution is obtained from the design subproblem, the primal heuristic step is triggered using as input the solution $\widehat{y}$ of the design subproblem if: 1) the frequency of total number of subgradient iterations is equal to a predefined number, fr^{prim}, or 2) the lower bound has improved significantly since the last time the primal heuristic was called. The improvement is considered to be significant if $(LB_c - LB_l)/LB_l > \delta^d$, where δ^d is a parameter (set to 1%) and LB_c and LB_l are, respectively, the lower bound computed at the current iteration and the lower bound obtained the last time the primal heuristic was triggered.

At the end of each execution of the primal heuristic procedure, in addition to the intensification step described in Section 5.3.2, a diversification step is performed if the algorithm is not able to improve the upper bound for a predefined maximum number of primal heuristic steps, st^p_{max}, where a *primal heuristic step* is a complete slope scaling procedure with intensification. The idea of diversification is to avoid selecting the arcs that frequently appeared in the obtained feasible solutions. The design solution of the best upper bound found so far is considered to initialize the linearized cost of formula (5.16) where a huge linearized cost is assigned not only to the closed arcs but also to a small set of arcs (set to 10) selected randomly from the most frequently opened ones in a percentage (set to 90%) of the obtained feasible solutions so far. Then, a new slope scaling procedure is triggered using the new initial linearized costs. The selected arcs are marked to avoid choosing them again in the next diversification step. Therefore, at each diversification step, a new set of the most frequently opened arcs are penalized. We restrict the total number of slope scaling steps in each diversification procedure and the total number of diversification steps to it^d_{max} and st^d_{max}, respectively.

Three different termination conditions are considered to stop the Lagrangian phase: 1) if the procedure elapsed time reaches a predefined time limit t^{Lag}, 2) if the subgradient norm is less than ε, and 3) if the total number of the subgradient iterations reaches a predefined number it^{Lag}. At the end of the subgradient procedure, a slope scaling step is called using the design solution of the best obtained lower bound found so far in the hope of finding a better upper bound.

5.4.3 Post-Optimization Phase

At the end of the algorithm, a *post-optimization* phase is performed by solving a restricted MIP problem, $\mathscr{P}_{PO}$, for a limited time $t^{PostOpt}$. To build the restricted MIP, we use a number of elite solutions, the feasible solutions selected from the long term memory, F, with less than $\delta_p\%$ gap from the best known feasible solution. The design variables that are closed in all the elite solutions are fixed to 0. A simple iterative procedure is performed to set $\delta_p\%$ in such a way that at least $f\%$ of the design variables are set to 0 in the restricted MIP. At the beginning of the procedure, all the solutions in F are

selected as elite. If the percentage of fixed variables is more than $f\%$, then the desired percentage is obtained; otherwise, a new set of elite solutions is built by removing the worst solution from the elite solutions until the percentage falls into the desired range. Since there is a huge number of arcs in large-scale MSMCFND instances, in our tests, we set f to a large number (96%) to make the restricted MIP problem tractable.

5.4.4 Summary

The algorithm is presented in Algorithm 4. It starts with an initialization phase where $\mathscr{P}_{LP}$ is solved, and the Lagrange multipliers are initialized to the obtained dual values. In this phase, a graph traversal algorithm reduces the size of the shortest-path subproblems. At the end of initialization phase, a slope scaling procedure is triggered using the final design solution of the cutting plane $\mathscr{P}_{LP}$ as the input design solution $\widehat{y}$.

In the Lagrangian phase, we use a subgradient method to solve the Lagrangian dual. At each iteration, Dijkstra algorithm is used to solve the shortest path subproblems. The subgradient method follows a three-phase approach when handling the design subproblem. The algorithm finds a lower bound at each iteration of the subgradient method. Using the obtained solution of the design subproblem, the slope scaling heuristic, with the intensification and diversification procedures, finds feasible solutions. At each subgradient iteration, the best obtained upper bound is used in the subgradient procedure to define a better search direction.

At the end of the subgradient procedure, another slope scaling step is performed using the design solution of the best known lower bound. At the end of the algorithm, the post-optimization phase improves the upper bound using the elite solutions stored in a long-term memory during the algorithm.

5.5 Experimental Results

In this section, we present the set of randomly generated instances used to test the performance of our algorithm. Then, we describe how we have adjusted the parameters of the algorithm. Finally, we analyze the results of the experiments.

Algorithm 4: Lagrangian-based matheuristic for MSMCFND

Solve $\mathscr{P}_{LP} \rightarrow \widehat{y}$
Initialize Lagrange multipliers
Run primal heuristic using $\widehat{y}$
$it = 0$
repeat
  if $it < it^{ph1}_{max}$ **then**
    solve LP relaxation of $\mathscr{P}_D \rightarrow \widehat{y}$
  else if $it < it^{ph2}_{max}$ **then**
    solve root node of IP $\mathscr{P}_D \rightarrow \widehat{y}$
  else
    solve IP $\mathscr{P}_D$ for a limited time $\rightarrow \widehat{y}$
  end if
  Solve $\mathscr{P}^k_{SP}$ for each commodity $k \in K$
  Compute lower bound
  Update $LB^{best} \rightarrow \underline{y}$
  if $\widehat{y}$ is new **and** the frequency of the number of iterations is equal to fr^{prim} **then**
    Run primal heuristic using $\widehat{y}$
    if number of non-improvement iterations in upper bound is greater than st^p_{max} **then**
      Perform diversification
    end if
  end if
  Update Lagrange multipliers
  $it = it + 1$
until one of the termination conditions is met
Run primal heuristic using $\underline{y}$
Run post-optimization by solving $\mathscr{P}_{PO}$

5.5.1 Instances

To generate the instances, we use a *time-space network* structure in which the physical nodes (terminals) are repeated for each time period to represent the time dependency. A node in such a network represents a terminal in a specific time period, and each arc represents a transfer from a terminal in a time period to either the same or a different terminal in another time period, see [88] for more details.

To generate the instances, let $S = \{1,2,...,|S|\}$ and $T = \{1,2,...,|T|\}$ be a set of terminals and a set of time periods, respectively. Then, the set of nodes is $N = \{1,2,...,|S| \times |T|\}$ where each node $i \in N$ corresponds to terminal $s = i/|T|$ and time period $t = ((i-1) \mod |T|) + 1$. For layer $|L|$, an arc is generated between each pair of nodes $(i \in N, j \in N)$

if the corresponding time period of i is less than the corresponding time period of j. In other words, we generate an arc from a terminal in a time period to either the same terminal or another terminal in a future time period. The next layers (layer $|L|-1$ to 1) are built by finding at most 50 paths between each pair of nodes of the upper layer. The values corresponding to flow capacity, design capacity, and costs of the arcs are uniformly distributed. Different parameters are used to generate instances with different characteristics: tight/loose capacity and with different fixed cost to variable cost ratio values. The generated instances are tested to be feasible. An infeasible instance is replaced by a new one to have a final feasible set of instances. The origins, the destinations and the demands of the commodities are also selected randomly. A random origin-destination pair $(o \in N, d \in N)$ is selected as the origin and the destination of a commodity if the corresponding time period of o is less than the one of d.

Two different sets of instances are generated randomly using the above instance generator procedure: MLTS1 (MLTS stands for MultiLayer Time-Space) and MLTS2 with two and three layers, respectively. Table 5.I summarizes the characteristics of the instances where $|L|$, $|A|$, $|S|$, $|T|$, $|N|$, and $|K|$ are the number of layers, number of arcs, number of terminals, number of time periods, number of nodes, and number of commodities, respectively. Note that in our instances, the flow variable costs are the same for all commodities on a particular arc. Therefore, it is possible to aggregate the commodities whose origins are the same into a single commodity to accelerate the solution of the MMCNF problem of the primal heuristic.

5.5.2 Parameter Setting

We use a two-phase parameter setting strategy to calibrate the parameters. In the first and second phases, we set the subgradient and the primal heuristic parameters, respec-

Table 5.I – Instances

	# of Instances	$\|L\|$	$\|A\|$	$\|S\|$	$\|T\|$	$\|N\|$	$\|K\|$
MLTS1	32	2	20612-30664	8,10	7	56,70	100,200
MLTS2	32	3	42768-64162	8,10	7	56,70	100,200

tively. Tables 5.II and 5.III display the tested values of the parameters for each phase. We choose 20% of the instances randomly for the parameter setting process.

As mentioned before, the performance of the subgradient method depends on its parameters, the most critical being λ^i, which is adjusted based on the parameters γ_1, γ_2 and λ^0. To find the best values of the important parameters (Table 5.II), we fix all the parameters, change one parameter at a time, and select the value that produces better results on average for all the test instances. In Table 5.II, the selected values are shown in bold. We set the values of the less-important parameters, as suggested in [26] to: $\varepsilon = 10^{-7}$ and $\lambda_{min} = 10^{-3}$. We set the maximum number of total subgradient iterations to a large value ($\gamma_3 = 3000$). The value of it_{max}^{ph3} is then equal to $\gamma_3 - (it_{max}^{ph2} + it_{max}^{ph3})$.

Table 5.II – Values of parameters tested for subgradient method

Name	Description	Tested Values
t^{LP}	time limit for solving cutting plane $\mathscr{P}_{LP}$ (minutes)	3, 4, **5**, 6, 7, 8, 9
t^{Des}	time limit for solving design subproblem (minutes)	0.5, **1**, 5, 10
γ_1	parameter to change λ^i	1.5, **2**, 2.5, 3, 3.5
γ_2	maximum number of non improvement iterations in lower bound	20, 40, 60, 80, **100**, 120
λ^0	initial value of subgradient scaling factor (λ^i)	1, 1.5, **2**
it_{max}^{ph1}	number of subgradient iterations to solve LP $\mathscr{P}_D$	5%, **10%**, 20%, 30% of γ_3
it_{max}^{ph2}	number of subgradient iterations to solve $\mathscr{P}_D$ in root node	5%, **10%**, 20%, 30% of γ_3

Table 5.III shows the values that are tested to set the parameters of the primal heuristic. We use the same strategy as for the subgradient procedure: we fix all the parameters, change one of them at a time, and select the best value. The table shows the selected values in bold. Some of the parameters of the primal heuristic can be set initially without testing different values. $t^{PostOpt}$ is determined based on two previous timing parameters, t^{LP} and t^{Lag}. The big M parameter has to be large enough to ensure not to select the design variables that have value zero in the design subproblem solution, and can be obtained using the following formula:

$$M = 10 \times \frac{\max_{l \in L, a \in A_l} \{f_{al}\}}{\min_{l \in L, a \in A_l} \{f_{al}\}} \tag{5.24}$$

Table 5.III – Values of parameters tested for the primal heuristic

Name	Description	Tested Values
fr^{prim}	primal heuristic frequency	5, 10, 15, **20**, 25
t^{Lag}	time limit on Lagrangian procedure including subgradient and primal heuristic (hours)	5, 6, **7**, 8
it^{s}_{max}	maximum number of slope scaling iterations in each primal heuristic step	10, **15**, 20
n	number of intensification iterations	1, **4**, 8
st^{p}_{max}	maximum number of primal heuristic steps with no improvement in upper bound	3, **5**, 8, 10
it^{d}_{max}	total number of iterations in each diversification procedure	5, **10**, 15, 20
st^{d}_{max}	total number of diversification steps	1, **3**, 4, 5

5.5.3 Upper Bound Analysis

The proposed Lagrangian heuristic is evaluated using MLTS1 and MLTS2 instances. We compare the results of the proposed algorithm to those obtained by CPLEX 12.6. with default setting. We add our preprocessing procedure before starting the CPLEX solver. We also consider different scenarios to determine whether it is suitable to add the strong linking constraints (5.4) and the upward design linking constraints (5.6) to the model of CPLEX or not. The tested scenarios includes adding each set of constraints to the model of CPLEX: 1) a priori, 2) as user cuts, 3) as lazy constraints, and 4) using cut callback functions of CPLEX. The tests show that the best case for CPLEX is not to add any of the upward design linking constraints but to add strong linking constraints as user cuts to the model.

The results are summarized in Tables 5.IV and 5.V for MLTS1 and MLTS2 instances. We fix the total time limit to 10 hours for CPLEX and for the proposed algorithm. The second column in these tables shows the characteristics of the instances:

— $|A_l|$: number of arcs in layer l;

— $|N|$: number of nodes;

— $|K|$: number of commodities;

— FCTI: Flow Capacity Tightness Index defined as the flow capacity average:

$$(\sum_{l \in L} \sum_{a \in A_l} u_{al})/|A|,$$

divided by the total demand, $\sum_{k \in K} d^k$, L stands for loose flow capacity (FCTI ≥ 0.05), while T refers to tight flow capacity (FCTI < 0.05);

Table 5.IV – CPLEX and the proposed algorithm comparison for MLTS1 (2-layer instances)

#	($\|A1\|$, $\|A2\|$, $\|N\|$, $\|K\|$, FCTI, DCTI, CDI)	T(sec)	Optimality Gap %		LB/UB Gap %	
			Gap^{CUB}_{CLB}	Gap^{LUB}_{LLB}	Gap^{CLB}_{LLB}	Gap^{CUB}_{LUB}
1	(19952,660, 56, 100, L, L, F)	13055.5	6.82	17.27	10.79	0.48
2	(19952,660, 56, 100, L, L, V)	26895.6	15.90	18.76	7.57	**-4.32**
3	(19952,660, 56, 100, L, T, F)	14321.6	13.93	20.84	9.53	**-1.64**
4	(19952,660, 56, 100, L, T, V)	Limit	11.22	15.86	4.67	0.59
5	(19952,660, 56, 100, T, L, F)	Limit	16.01	19.32	6.94	**-3.12**
6	(19952,660, 56, 100, T, L, V)	Limit	14.27	16.05	6.30	**-4.31**
7	(19952,660, 56, 100, T, T, F)	Limit	16.40	21.73	9.84	**-3.69**
8	(19952,660, 56, 100, T, T, V)	Limit	16.22	16.36	3.51	**-3.35**
9	(19952,660, 56, 200, L, L, F)	34545.7	26.55	**25.02**	9.12	**-10.98**
10	(19952,660, 56, 200, L, L, V)	Limit	75.52	**19.70**	7.44	**-71.78**
11	(19952,660, 56, 200, L, T, F)	Limit	83.65	**32.25**	13.78	**-79.20**
12	(19952,660, 56, 200, L, T, V)	Limit	20.32	21.22	6.74	**-5.68**
13	(19952,660, 56, 200, T, L, F)	Limit	22.48	**19.34**	4.85	**-8.55**
14	(19952,660, 56, 200, T, L, V)	Limit	26.34	**18.82**	4.91	**-13.72**
15	(19952,660, 56, 200, T, T, F)	Limit	73.67	**20.22**	3.57	**-68.17**
16	(19952,660, 56, 200, T, T, V)	Limit	76.87	**23.52**	6.21	**-71.63**
17	(29794,870, 70, 100, L, L, F)	31989.3	9.83	16.90	9.10	**-1.36**
18	(29794,870, 70, 100, L, L, V)	32681.7	6.97	12.46	7.35	**-1.53**
19	(29794,870, 70, 100, L, T, F)	Limit	5.94	13.78	8.29	0.05
20	(29794,870, 70, 100, L, T, V)	25351.6	5.97	10.00	5.56	**-1.34**
21	(29794,870, 70, 100, T, L, F)	Limit	8.05	16.22	8.36	0.57
22	(29794,870, 70, 100, T, L, V)	Limit	15.59	18.32	7.01	**-3.90**
23	(29794,870, 70, 100, T, T, F)	Limit	10.49	15.45	6.68	**-1.21**
24	(29794,870, 70, 100, T, T, V)	Limit	14.27	17.18	6.91	**-3.65**
25	(29794,870, 70, 200, L, L, F)	Limit	95.63	**27.31**	13.07	**-94.78**
26	(29794,870, 70, 200, L, L, V)	Limit	94.85	**15.23**	6.61	**-94.33**
27	(29794,870, 70, 200, L, T, F)	Limit	95.86	**21.33**	8.04	**-95.17**
28	(29794,870, 70, 200, L, T, V)	Limit	94.59	**20.43**	5.36	**-93.57**
29	(29794,870, 70, 200, T, L, F)	Limit	29.50	**24.91**	11.16	**-16.59**
30	(29794,870, 70, 200, T, L, V)	Limit	94.35	**14.90**	6.15	**-93.77**
31	(29794,870, 70, 200, T, T, F)	Limit	79.41	**22.93**	8.71	**-75.61**
32	(29794,870, 70, 200, T, T, V)	Limit	68.40	**17.30**	6.67	**-64.34**
		Average	38.93	19.09	7.52	-30.93
		Minimum	5.94	10.00	3.51	-95.17
		Maximum	95.86	32.25	13.78	0.59

— DCTI: Design Capacity Tightness Index defined as the design capacity average, $(\sum_{l\in L, l\geq 2}\sum_{a\in A_l} v_{al})/(|A| - |A_1|)$, divided by $B^{al}_{ave} = \sum_{l\in L, l\geq 2}\sum_{a\in A_l} |B^{al}|/(|A| - |A_1|)$, L stands for loose design capacity (DCTI ≥ 0.3), while T refers to tight design capacity (DCTI < 0.3);

— CDI: Cost Dominance Index defined as the variable flow cost average:

$$(\sum_{k\in K}\sum_{a\in A_1} c^k_{a1})/|A_1|,$$

divided by the design cost average, $(\sum_{l\in L}\sum_{a\in A_l} f_{al})/|A|$, F refers to the cases where the fixed costs are predominant relatively to the variable costs (CDI $<$ 0.01), and V means the reverse (CDI ≥ 0.01).

In Tables 5.IV and 5.V, column T(sec) is the total run time of the proposed algorithm. CPLEX was not able to converge to optimality on any of the instances in 10 hours (time limit). Therefore, the run times of CPLEX are not reported in these tables (CPLEX run time is 10h). Let CUB and CLB be the upper and lower bounds obtained by CPLEX, respectively. Let also LUB and LLB be the upper and lower bounds obtained by the proposed algorithm, respectively. Columns $\text{Gap}^{\text{CUB}}_{\text{CLB}} = 100 \times (\text{CUB} - \text{CLB})/\text{CUB}$ and $\text{Gap}^{\text{LUB}}_{\text{LLB}} = 100 \times (\text{LUB} - \text{LLB})/\text{LUB}$ are the optimality gaps for CPLEX and the proposed algorithm, respectively. Column $\text{Gap}^{\text{CLB}}_{\text{LLB}} = 100 \times (\text{CLB} - \text{LLB})/\text{LLB}$ is the gap between the lower bound of CPLEX and the lower bound obtained by our algorithm. Column $\text{Gap}^{\text{LUB}}_{\text{CUB}} = 100 \times (\text{LUB} - \text{CUB})/\text{CUB}$ is the gap between the upper bound of CPLEX and the upper bound obtained by our algorithm. A negative value indicates that the Lagrangian-based matheuristic method found a better feasible solution than CPLEX.

The results show that the algorithm is better than CPLEX in almost all instances (90%) in terms of the upper bounds (last column of Tables 5.IV and 5.V). The averages of the improvements to the upper bounds are 30.93% and 46.08% for MLTS1 and MLTS2, respectively. The algorithm improves the upper bounds by more than 10% in 57% of the instances.

The proposed algorithm finds better optimality gaps than CPLEX in 60% of the instances (bold optimality gaps in column $\text{Gap}^{\text{LUB}}_{\text{LLB}}$). The goal of the proposed algorithm, as a matheuristic, is to find high-quality upper bounds. However, one of the advantages of this algorithm is that it also computes lower bounds. The obtained lower bounds enable us to evaluate the quality of the obtained upper bounds, particularly for the instances where CPLEX ends up with an optimality gap of more than 90%.

Table 5.VI shows the analysis of the performance of the algorithm depending on the number of layers, number of arcs and number of commodities. The table shows that the difficulty of the instances increases with the number of layers, the number of arcs, the number of nodes and, even more significantly, the number of commodities. The algorithm improves the upper bound up to 71.66% on average for the instances with 200 commodities, while the average is 5.34% for the instances with 100 commodities.

5.5.4 Lower Bound Analysis

Tables 5.VII and 5.VIII show the lower bound analysis of the proposed Lagrangian relaxation for MLTS1 and MLTS2, respectively. Column Lag Time shows the computation time of the bounding procedure including the initialization phase and the subgradient method without the primal heuristic and the post optimization phase. Column Strong LP shows 1) the computation time of the strong LP relaxation solved by CPLEX where the strong linking constraints (5.4) and the upward design linking constraints (5.6) are added a priori to the model, and 2) the gap between the Lagrangian LB and the strong LP lower bound. Column Root Node presents 1) the computation time at the root node of a branch-and-bound algorithm solved by CPLEX, and 2) the gap between the Lagrangian LB and the root node lower bound.

For the Strong LP case, we also tested a cutting plane approach where inequalities (5.4) and (5.6) are added iteratively. The computation time of the cutting plane approach is much worse than the a priori approach. For the Root Node case, we turned off the heuristic of CPLEX but allowed other possible valid inequalities to be added to the model. We introduced strong inequalities as user cuts to the root node of CPLEX. We did not add the upward design linking constraints because it takes a long time (more than 10 hours) to add them as user cuts. When CPLEX adds some other cuts at the root note, the results can be better than the strong LP. However, since CPLEX does not add all the strong inequalities, and neither the upward linking constraints, it is possible that the root lower bound of CPLEX is worse than the strong LP lower bound.

Although the lower bounds of the subgradient procedure are around 6% away, on average, from the lower bounds of the strong LP, its average computation time is much

less. The average computation time of the subgradient process for MLTS2 instances, for example, is about one hour while it is 10 hours and 2 hours for the strong LP and the root node, respectively. The standard deviation of the computation time is also much less than those of the strong LP and the root node. For MLTS2 instances, the computation time of the subgradient procedure varies from 36 to 120 minutes, with a standard deviation of 26 minutes, while it varies from 1 hour to 25 hours for CPLEX to solve the strong LP, with a standard deviation of 7 hours. Although finding high-quality solutions is the primary goal of this paper, the proposed algorithm efficiently generates effective lower bounds that allow to evaluate the obtained feasible solutions.

5.6 Conclusions

We have proposed a general formulation for the MSMCFND as well as an effective algorithm to solve large-scale MSMCFND instances. The algorithm has been enhanced by a preprocessing procedure, intensification and diversification steps, and a post-optimization phase. We can summarize the obtained results as follows:

- The algorithm produces better upper bounds than CPLEX in 90% of the instances.
- The algorithm is better than CPLEX in 60% of the instances in terms of the optimality gap.
- The algorithm improves the upper bound by more than 10% in 57% of the instances.
- The complexity of the instances increases with the number of layers, the number of arcs and, even more significantly, the number of commodities.
- The performance of the Lagrangian relaxation is robust on either easy or difficult instances in terms of computational time, i.e., the standard deviation of the computation time is also much less than those of the strong LP and root node solved by CPLEX.

Four main interesting research avenues are: 1) improving the algorithm using more effective non-differentiable optimization approaches such as the bundle method, and

embedding the algorithm into a branch-and-bound structure to derive an exact solution methodology, 2) exploring new solution methods particularly those that have been successfully applied before to the MCFND, 3) proposing branch-and-cut approaches by developing new valid inequalities based on the multilayer structure of the problem, and 4) considering problems where there are commodities to be routed in all layers.

Table 5.V – CPLEX and the proposed algorithm comparison for MLTS2 (3-layer instances)

			Optimality Gap %		LB/UB Gap %	
#	($\|A1\|$, $\|A2\|$, $\|A3\|$, $\|N\|$, $\|K\|$, FCTI, DCTI, CDI)	T(sec)	Gap^{CUB}_{CLB}	Gap^{LUB}_{LLB}	Gap^{CLB}_{LLB}	Gap^{CUB}_{LUB}
33	(22156,19952, 660, 56, 100, L, L, F)	Limit	19.57	20.00	6.09	**-5.58**
34	(22156,19952, 660, 56, 100, L, L, V)	Limit	29.57	**24.57**	7.75	**-13.87**
35	(22156,19952, 660, 56, 100, L, T, F)	Limit	26.84	**22.68**	5.53	**-10.62**
36	(22156,19952, 660, 56, 100, L, T, V)	Limit	12.53	18.34	6.27	0.40
37	(22156,19952, 660, 56, 100, T, L, F)	Limit	21.73	**15.95**	5.49	**-11.98**
38	(22156,19952, 660, 56, 100, T, L, V)	Limit	20.92	**17.17**	4.62	**-8.94**
39	(22156,19952, 660, 56, 100, T, T, F)	Limit	23.52	**16.76**	5.27	**-12.96**
40	(22156,19952, 660, 56, 100, T, T, V)	Limit	21.24	**12.87**	4.66	**-13.81**
41	(22156,19952, 660, 56, 200, L, L, F)	Limit	98.32	**22.76**	4.30	**-97.91**
42	(22156,19952, 660, 56, 200, L, L, V)	Limit	97.96	**21.03**	2.86	**-97.49**
43	(22156,19952, 660, 56, 200, L, T, F)	Limit	98.13	**22.12**	4.65	**-97.71**
44	(22156,19952, 660, 56, 200, L, T, V)	Limit	97.90	**21.92**	3.41	**-97.41**
45	(22156,19952, 660, 56, 200, T, L, F)	Limit	97.66	**18.15**	3.64	**-97.24**
46	(22156,19952, 660, 56, 200, T, L, V)	Limit	72.96	**17.85**	3.62	**-68.28**
47	(22156,19952, 660, 56, 200, T, T, F)	Limit	97.60	**20.17**	4.81	**-97.13**
48	(22156,19952, 660, 56, 200, T, T, V)	Limit	66.11	**18.94**	4.06	**-59.88**
49	(33498,29794, 870, 70, 100, L, L, F)	Limit	22.30	26.30	11.40	**-6.59**
50	(33498,29794, 870, 70, 100, L, L, V)	Limit	15.19	22.67	11.58	**-3.02**
51	(33498,29794, 870, 70, 100, L, T, F)	Limit	21.70	31.26	11.66	0.64
52	(33498,29794, 870, 70, 100, L, T, V)	Limit	18.46	20.16	7.89	**-5.93**
53	(33498,29794, 870, 70, 100, T, L, F)	Limit	19.92	20.06	7.61	**-7.45**
54	(33498,29794, 870, 70, 100, T, L, V)	Limit	9.57	13.52	6.00	**-1.71**
55	(33498,29794, 870, 70, 100, T, T, F)	Limit	22.48	**17.54**	6.44	**-12.05**
56	(33498,29794, 870, 70, 100, T, T, V)	Limit	34.98	**18.45**	6.72	**-25.63**
57	(33498,29794, 870, 70, 200, L, L, F)	Limit	98.53	**36.89**	16.18	**-98.05**
58	(33498,29794, 870, 70, 200, L, L, V)	Limit	98.32	**27.39**	10.97	**-97.94**
59	(33498,29794, 870, 70, 200, L, T, F)	Limit	98.32	**28.85**	13.06	**-97.95**
60	(33498,29794, 870, 70, 200, L, T, V)	Limit	98.33	**29.68**	13.99	**-97.95**
61	(33498,29794, 870, 70, 200, T, L, F)	Limit	61.31	**22.58**	6.87	**-53.46**
62	(33498,29794, 870, 70, 200, T, L, V)	Limit	64.62	**20.52**	6.36	**-58.32**
63	(33498,29794, 870, 70, 200, T, T, F)	Limit	61.60	**24.01**	6.28	**-52.64**
64	(33498,29794, 870, 70, 200, T, T, V)	Limit	71.23	**23.47**	9.51	**-65.98**
		Average	53.73	21.71	7.17	-46.08
		Minimum	9.57	12.87	2.86	-98.05
		Maximum	98.53	36.89	16.18	0.64

Table 5.VI – Performance analysis of the proposed algorithm in terms of $|K|$, $|L|$, $|A|$ and $|N|$.

	$\|K\|$		$\|L\|$		$\|A\|$				$\|N\|$	
	100	200	2	3	20612	30664	42768	64162	56	70
Gap^{LUB}_{CUB} Average	-5.34	-71.66	-30.93	-46.08	-21.82	-40.03	-49.40	-42.75	-35.61	-41.39

Table 5.VII – Lower bound analysis: comparing proposed Lagrangian to CPLEX root node and strong LP for MLTS1.

			Strong LP		Root Node	
#	($\|A1\|$, $\|A2\|$, $\|N\|$, $\|K\|$, FCTI, DCTI, CDI)	Lag Time	Lag gap	Time	Lag gap	Time
1	(19952,660, 56, 100, L, L, F)	0h 14m	9.08	0h 35m	8.92	0h 11m
2	(19952,660, 56, 100, L, L, V)	0h 14m	8.38	0h 31m	8.28	0h 30m
3	(19952,660, 56, 100, L, T, F)	0h 14m	8.59	0h 21m	8.53	0h 11m
4	(19952,660, 56, 100, L, T, V)	0h 14m	4.00	0h 20m	3.83	0h 21m
5	(19952,660, 56, 100, T, L, F)	0h 14m	5.12	0h 40m	5.07	0h 7m
6	(19952,660, 56, 100, T, L, V)	0h 14m	5.61	0h 25m	5.62	0h 8m
7	(19952,660, 56, 100, T, T, F)	0h 20m	7.79	0h 28m	7.77	0h 17m
8	(19952,660, 56, 100, T, T, V)	0h 20m	2.82	0h 21m	2.88	0h 11m
9	(19952,660, 56, 200, L, L, F)	0h 32m	6.00	5h 35m	5.91	0h 37m
10	(19952,660, 56, 200, L, L, V)	0h 31m	6.48	3h 38m	6.26	0h 37m
11	(19952,660, 56, 200, L, T, F)	0h 29m	6.99	5h 20m	6.83	1h 46m
12	(19952,660, 56, 200, L, T, V)	0h 19m	5.58	2h 27m	5.51	1h 5m
13	(19952,660, 56, 200, T, L, F)	0h 19m	4.48	3h 49m	4.49	0h 47m
14	(19952,660, 56, 200, T, L, V)	0h 19m	4.50	2h 31m	4.51	0h 43m
15	(19952,660, 56, 200, T, T, F)	0h 30m	3.13	4h 3m	3.19	0h 18m
16	(19952,660, 56, 200, T, T, V)	0h 19m	4.47	2h 13m	4.47	0h 25m
17	(29794,870, 70, 100, L, L, F)	0h 18m	7.91	0h 53m	7.75	0h 40m
18	(29794,870, 70, 100, L, L, V)	0h 29m	6.71	0h 33m	6.63	0h 47m
19	(29794,870, 70, 100, L, T, F)	0h 28m	5.42	0h 27m	5.21	0h 20m
20	(29794,870, 70, 100, L, T, V)	0h 29m	4.65	0h 22m	4.53	1h 24m
21	(29794,870, 70, 100, T, L, F)	0h 19m	7.87	0h 56m	7.81	0h 38m
22	(29794,870, 70, 100, T, L, V)	0h 20m	3.42	0h 31m	3.35	1h 12m
23	(29794,870, 70, 100, T, T, F)	0h 19m	5.41	0h 51m	5.29	0h 29m
24	(29794,870, 70, 100, T, T, V)	0h 19m	3.76	0h 32m	3.66	0h 38m
25	(29794,870, 70, 200, L, L, F)	0h 28m	8.41	3h 23m	8.23	0h 50m
26	(29794,870, 70, 200, L, L, V)	0h 27m	4.78	3h 57m	4.69	0h 48m
27	(29794,870, 70, 200, L, T, F)	0h 28m	5.54	4h 13m	5.40	1h 6m
28	(29794,870, 70, 200, L, T, V)	0h 28m	3.14	2h 47m	3.02	0h 59m
29	(29794,870, 70, 200, T, L, F)	0h 29m	8.64	7h 57m	8.62	1h 15m
30	(29794,870, 70, 200, T, L, V)	0h 26m	4.96	1h 33m	4.92	0h 56m
31	(29794,870, 70, 200, T, T, F)	0h 26m	7.53	3h 35m	7.54	0h 44m
32	(29794,870, 70, 200, T, T, V)	0h 27m	5.30	2h 13m	5.31	0h 57m
	Average	0h 23m	5.83	2h 7m	5.75	0h 41m
	Minimum	0h 14m	2.82	0h 20m	2.88	0h 7m
	Maximum	0h 32m	9.08	7h 57m	8.92	1h 46m
	Standard Deviation	0h 6m		1h 57m		0h 24m

Table 5.VIII – Lower bound analysis: comparing proposed Lagrangian to CPLEX root node and strong LP for MLTS2.

			Strong LP		Root Node	
#	($\|A1\|$, $\|A2\|$, $\|N\|$, $\|K\|$, FCTI, DCTI, CDI)	Lag Time	Lag gap	Time	Lag gap	Time
33	(19952,660, 56, 100, L, L, F)	0h 36m	5.23	6h 27m	5.64	0h 32m
34	(19952,660, 56, 100, L, L, V)	0h 37m	7.24	6h 43m	7.35	0h 52m
35	(19952,660, 56, 100, L, T, F)	0h 40m	4.69	3h 25m	5.06	0h 58m
36	(19952,660, 56, 100, L, T, V)	0h 37m	7.15	1h 0m	7.50	0h 56m
37	(19952,660, 56, 100, T, L, F)	0h 36m	4.65	3h 35m	5.29	1h 30m
38	(19952,660, 56, 100, T, L, V)	0h 38m	3.59	4h 26m	4.31	1h 3m
39	(19952,660, 56, 100, T, T, F)	0h 43m	3.57	2h 41m	4.62	0h 26m
40	(19952,660, 56, 100, T, T, V)	0h 39m	3.52	2h 3m	4.22	0h 34m
41	(19952,660, 56, 200, L, L, F)	0h 48m	3.96	21h 38m	4.28	1h 19m
42	(19952,660, 56, 200, L, L, V)	0h 46m	2.25	12h 31m	2.53	2h 11m
43	(19952,660, 56, 200, L, T, F)	0h 56m	3.41	15h 37m	3.62	1h 34m
44	(19952,660, 56, 200, L, T, V)	0h 45m	2.86	15h 13m	3.15	1h 24m
45	(19952,660, 56, 200, T, L, F)	0h 43m	2.56	16h 33m	3.43	1h 18m
46	(19952,660, 56, 200, T, L, V)	0h 43m	2.43	9h 57m	3.26	1h 11m
47	(19952,660, 56, 200, T, T, F)	1h 1m	3.42	21h 40m	3.96	2h 8m
48	(19952,660, 56, 200, T, T, V)	0h 45m	3.04	15h 11m	3.88	0h 52m
49	(29794,870, 70, 100, L, L, F)	0h 57m	6.90	8h 20m	7.17	3h 15m
50	(29794,870, 70, 100, L, L, V)	0h 58m	4.47	2h 42m	4.45	3h 6m
51	(29794,870, 70, 100, L, T, F)	1h 34m	3.89	6h 17m	3.98	2h 52m
52	(29794,870, 70, 100, L, T, V)	1h 35m	4.79	3h 25m	4.85	2h 36m
53	(29794,870, 70, 100, T, L, F)	0h 50m	6.49	2h 42m	7.19	1h 55m
54	(29794,870, 70, 100, T, L, V)	0h 47m	4.22	2h 30m	4.94	1h 13m
55	(29794,870, 70, 100, T, T, F)	1h 34m	4.49	1h 30m	5.14	1h 59m
56	(29794,870, 70, 100, T, T, V)	1h 29m	5.20	3h 35m	6.04	2h 49m
57	(29794,870, 70, 200, L, L, F)	1h 57m	4.91	15h 55m	5.13	2h 38m
58	(29794,870, 70, 200, L, L, V)	1h 9m	4.22	17h 4m	4.57	2h 42m
59	(29794,870, 70, 200, L, T, F)	1h 7m	5.21	25h 1m	5.55	3h 37m
60	(29794,870, 70, 200, L, T, V)	1h 59m	3.10	21h 55m	3.40	5h 24m
61	(29794,870, 70, 200, T, L, F)	1h 5m	3.04	18h 10m	3.79	6h 40m
62	(29794,870, 70, 200, T, L, V)	1h 12m	3.15	18h 37m	4.52	4h 9m
63	(29794,870, 70, 200, T, T, F)	2h 0m	3.15	19h 35m	4.15	3h 3m
64	(29794,870, 70, 200, T, T, V)	1h 9m	4.00	8h 58m	4.58	1h 59m
	Average	1h 2m	4.21	10h 28m	4.74	2h 9m
	Minimum	0h 36m	2.24	1h 0m	2.53	0h 26m
	Maximum	2h 0m	7.24	25h 1m	7.50	6h 40m
	Standard Deviation	0h 26m		7h 32m		1h 25m

CHAPTER 6

CONCLUSION

Lagrangian-based algorithms are one of the most effective solution methods to solve network design problems, in particular the single-layer MCFND. The usual Lagrangian relaxations for the formulation are the so-called shortest path (commodity-based) and knapsack (arc-based) relaxations. For the first one, the resulting Lagrangian subproblem decomposes into a collection of shortest path subproblems, one for each commodity, while the second one allows solving the Lagrangian subproblem as a series of continuous knapsack subproblems, one for each arc. The nodes of a network are the other entities that can be considered as decomposition components. We have proposed three new node-based Lagrangian relaxation-reformulations for the multicommodity capacitated fixed-charge network design problem. A Lagrangian-based matheuristic has also been proposed to find upper bounds. We have conducted significant computational experiments on the benchmark instances. The Lagrangian dual bound of the new node-based Lagrangian relaxations improve significantly upon the so-called strong LP bound (known to be equal to the Lagrangian dual bounds of the flow and knapsack relaxations). The node-based proposed Lagrangian heuristic based on the Location relaxation outperforms traditional flow and knapsack based heuristics. The proposed node-based Lagrangian heuristic algorithm outperforms almost all the previously proposed heuristics in the literature in average.

The applications of network design models and their solution techniques have been surveyed in the literature. However, in recent years, interesting applications of multilayer network design have appeared, that are not covered in these surveys. We have presented a state-of-the-art review on multilayer network design modeling and methodological approaches in telecommunications and transportation applications. The proposed review sheds lights on different models, solution method challenges, and identifies several important research avenues. Moreover, we proposed a general formulation for the multilayer network design problem. This model can formulate most of the applications in

telecommunications and transportation.

Motivated by the success of Lagrangian-based heuristics for the single-layer MCFND, we have proposed a general formulation for the MSMCFND, as well as an effective algorithm to solve large-scale MSMCFND instances. The algorithm has been enhanced by a preprocessing procedure, intensification and diversification steps, and a post-optimization phase. The proposed algorithm produces better upper bounds than CPLEX in 92% of the instances. It is also better than CPLEX in 68% of the instances in terms of the optimality gap. It improves the upper bound by more than 10% in 56% of the instances. The performance of the proposed algorithm for the MSMCFND is robust in either the easy or the difficult instances.

The first future research avenue is to develop exact solution methods by embedding the proposed Lagrangian relaxation into a branch-and-bound procedure or other MIP schemes. To do so, we are developing a new algorithm called *Relax-Fix-and-Cut* (RF&C) for the MCFND. RF&C performs three main steps at each node of a branch-and-bound procedure: 1) relax, where a relaxation of the formulation is solved, 2) fix, where the design variables are fixed based on the solution of the relaxation to produce a restricted MIP, and 3) cut, where a pseudo cut is added to the formulation to remove the explored fixed solution space and produce new relaxation node. The restricted MIP is addressed by a state-of-the-art MIP solver. The RF&C algorithm is in the final stages of implementation and test. For the MCFND, the second future research avenue is to improve the solution time of the node-based subproblems by using specialized algorithms.

For the MSMCFND, future research avenues include: 1) considering new problems with flow-design connectivity and with even more layers; 2) improving the proposed algorithm for the MSMCFND by using more efficient non-differentiable optimization approaches such as the bundle method; 3) proposing new solution methods particularly those that have been successfully applied before to the MCFND and can be used for the MSMCFND; 4) proposing branch-and-cut approaches by developing new valid inequalities based on the multilayer structure of the problem; 5) considering a more complex problem where commodities must be routed in all layers.

BIBLIOGRAPHY

[1] Mohammad Rahim Akhavan Kazemzadeh, Teodor Gabriel Crainic, and Bernard Gendron. A survey of multilayer network design. *Publication CIRRELT 2018*, 2018.

[2] Ada M Alvarez, José Luis González-Velarde, and Karim De-Alba. Scatter search for network design problem. *Annals of Operations Research*, 138(1):159–178, 2005.

[3] Jardar Andersen, Teodor Gabriel Crainic, and Marielle Christiansen. Service network design with asset management: Formulations and comparative analyses. *Transportation Research Part C: Emerging Technologies*, 17(2):197–207, 2009.

[4] Jardar Andersen, Teodor Gabriel Crainic, and Marielle Christiansen. Service network design with management and coordination of multiple fleets. *European Journal of Operational Research*, 193(2):377–389, 2009.

[5] Claudia Archetti, M Grazia Speranza, and Martin WP Savelsbergh. An optimization-based heuristic for the split delivery vehicle routing problem. *Transportation Science*, 42(1):22–31, 2008.

[6] Anantaram Balakrishnan, Thomas L Magnanti, A Shulman, and Richard T Wong. Models for planning capacity expansion in local access telecommunication networks. *Annals of Operations research*, 33(4):237–284, 1991.

[7] Anantaram Balakrishnan, Thomas L Magnanti, and Prakash Mirchandani. A dual-based algorithm for multi-level network design. *Management Science*, 40(5):567–581, 1994.

[8] Pietro Belotti, Antonio Capone, Giuliana Carello, and Federico Malucelli. Multi-layer MPLS network design: The impact of statistical multiplexing. *Computer Networks*, 52(6):1291–1307, 2008.

[9] Jacques F Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4(1):238–252, 1962.

[10] Valentina Cacchiani and Juan-José Salazar-González. Optimal solutions to a real-world integrated airline scheduling problem. *Transportation Science*, 51(1):250–268, 2016.

[11] Paolo M Camerini, Luigi Fratta, and Francesco Maffioli. On improving relaxation methods by modified gradient techniques. In *Nondifferentiable Optimization*, pages 26–34. Springer, 1975.

[12] Antonio Capone, Giuliana Carello, and Riccardo Matera. Multi-layer network design with multicast traffic and statistical multiplexing. In *IEEE GLOBECOM 2007-IEEE Global Telecommunications Conference*, pages 2565–2570. IEEE Operations Center, 2007.

[13] Mervat Chouman and Teodor Crainic. A mip-tabu search hybrid framework for multicommodity capacitated fixed-charge network design. *Publication CIRRELT 2010-31*, 2010.

[14] Mervat Chouman, Teodor Gabriel Crainic, and Bernard Gendron. Commodity representations and cut-set-based inequalities for multicommodity capacitated fixed-charge network design. *Transportation Science*, 51(2):650–667, 2016.

[15] Mervat Chouman, Teodor Gabriel Crainic, and Bernard Gendron. The impact of filtering in a branch-and-cut algorithm for multicommodity capacitated fixed charge network design. *EURO Journal on Computational Optimization*, 6(2):1–42, 2017.

[16] Amy Mainville Cohn and Cynthia Barnhart. Improving crew scheduling by incorporating key maintenance routing decisions. *Operations Research*, 51(3):387–396, 2003.

[17] Jean-François Cordeau, Goran Stojković, François Soumis, and Jacques Desrosiers. Benders decomposition for simultaneous aircraft routing and crew scheduling. *Transportation Science*, 35(4):375–388, 2001.

[18] Jean-François Cordeau, Federico Pasin, and Marius M Solomon. An integrated model for logistics network design. *Annals of Operations Research*, 144(1):59–82, 2006.

[19] Alysson M Costa. A survey on Benders decomposition applied to fixed-charge network design problems. *Computers & Operations Research*, 32(6):1429–1450, 2005.

[20] Alysson M Costa, Jean-François Cordeau, and Bernard Gendron. Benders, metric and cutset inequalities for multicommodity capacitated network design. *Computational Optimization and Applications*, 42(3):371–392, 2009.

[21] Alysson M Costa, Paulo M França, and Christiano Lyra Filho. Two-level network design with intermediate facilities: An application to electrical distribution systems. *Omega*, 39(1):3–13, 2011.

[22] Alysson M Costa, Jean-François Cordeau, Bernard Gendron, and Gilbert Laporte. Accelerating Benders decomposition with heuristicmaster problem solutions. *Pesquisa Operacional*, 32(1):03–20, 2012.

[23] Teodor Gabriel Crainic. Service network design in freight transportation. *European Journal of Operational Research*, 122(2):272–288, 2000.

[24] Teodor Gabriel Crainic and Michel Gendreau. Cooperative parallel tabu search for capacitated network design. *Journal of Heuristics*, 8(6):601–627, 2002.

[25] Teodor Gabriel Crainic, Michel Gendreau, and Judith M Farvolden. A simplex-based tabu search method for capacitated network design. *INFORMS Journal on Computing*, 12(3):223–236, 2000.

[26] Teodor Gabriel Crainic, Antonio Frangioni, and Bernard Gendron. Bundle-based relaxation methods for multicommodity capacitated fixed charge network design. *Discrete Applied Mathematics*, 112(1):73–99, 2001.

[27] Teodor Gabriel Crainic, Bernard Gendron, and Geneviève Hernu. A slope scaling/lagrangean perturbation heuristic with long-term memory for multicommodity capacitated fixed-charge network design. *Journal of Heuristics*, 10(5):525–545, 2004.

[28] Teodor Gabriel Crainic, Ye Li, and Michel Toulouse. A first multilevel cooperative algorithm for capacitated multicommodity network design. *Computers & Operations Research*, 33(9):2602–2622, 2006.

[29] Teodor Gabriel Crainic, Nicoletta Ricciardi, and Giovanni Storchi. Models for evaluating and planning city logistics systems. *Transportation Science*, 43(4):432–454, 2009.

[30] Teodor Gabriel Crainic, Xiaorui Fu, Michel Gendreau, Walter Rei, and Stein W Wallace. Progressive hedging-based metaheuristics for stochastic network design. *Networks*, 58(2):114–124, 2011.

[31] Teodor Gabriel Crainic, Mike Hewitt, Michel Toulouse, and Duc Minh Vu. Service network design with resource constraints. *Transportation Science*, 50(4):1380–1393, 2014.

[32] Teodor Gabriel Crainic, Mike Hewitt, Michel Toulouse, and Duc Minh Vu. Scheduled service network design with resource acquisition and management. *EURO Journal on Transportation and Logistics*, 7(3):277–309, 2018.

[33] Geir Dahl, Alexander Martin, and Mechthild Stoer. Routing through virtual paths in layered telecommunication networks. *Operations Research*, 47(5):693–702, 1999.

[34] Roberto De Franceschi, Matteo Fischetti, and Paolo Toth. A new ilp-based refinement heuristic for vehicle routing problems. *Mathematical Programming*, 105 (2-3):471–499, 2006.

[35] Jack J Dongarra. Performance of various computers using standard linear equations software. Technical report, University of Manchester, 2014.

[36] Matteo Fischetti and Andrea Lodi. Local branching. *Mathematical Programming*, 98(1-3):23–47, 2003.

[37] Matteo Fischetti, Carlo Polo, and Massimo Scantamburlo. A local branching heuristic for mixed-integer programs with 2-level variables, with an application to a telecommunication network design problem. *Networks*, 44(2):61–72, 2004.

[38] Bernard Fortz and Michael Poss. An improved benders decomposition applied to a multi-layer network design problem. *Operations Research Letters*, 37(5):359–364, 2009.

[39] Antonio Frangioni and Giorgio Gallo. A bundle type dual-ascent approach to linear multicommodity min-cost flow problems. *INFORMS Journal on Computing*, 11 (4):370–393, 1999.

[40] Antonio Frangioni and Enrico Gorgone. Bundle methods for sum-functions with "easy" components: applications to multicommodity network design. *Mathematical Programming*, 145(1-2):133–161, 2014.

[41] Antonio Frangioni, Bernard Gendron, and Enrico Gorgone. On the computational efficiency of subgradient methods: a case study with lagrangian bounds. *Mathematical Programming Computation*, pages 1–32, 2017.

[42] Bernard Gendron and Teodor Gabriel Crainic. Relaxations for multicommodity capacitated network design problems. *Publication CRT-965*, 1994.

[43] Bernard Gendron and Mathieu Larose. Branch-and-price-and-cut for large-scale multicommodity capacitated fixed-charge network design. *EURO Journal on Computational Optimization*, 2(1-2):55–75, 2014.

[44] Bernard Gendron, Teodor Gabriel Crainic, and Antonio Frangioni. *Telecommunications Network Planning*, chapter Multicommodity capacitated network design. Centre for Research on Transportation. Springer, Boston, MA, 1999.

[45] Bernard Gendron, Saïd Hanafi, and Raca Todosijević. Matheuristics based on iterative linear programming and slope scaling for multicommodity capacitated fixed charge network design. *European Journal of Operational Research*, 268(1):70–81, 2018.

[46] Arthur M Geoffrion. Lagrangean relaxation for integer programming. In *Approaches to Integer Programming*, pages 82–114. Springer, 1974.

[47] Ilfat Ghamlouche, Teodor Gabriel Crainic, and Michel Gendreau. Cycle-based neighbourhoods for fixed-charge capacitated multicommodity network design. *Operations Research*, 51(4):655–667, 2003.

[48] Ilfat Ghamlouche, Teodor Gabriel Crainic, and Michel Gendreau. Path relinking, cycle-based neighbourhoods and capacitated multicommodity network design. *Annals of Operations Research*, 131(1):109–133, 2004.

[49] Fred Glover. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, 13(5):533–549, 1986.

[50] Fred Glover, Manuel Laguna, and Rafael Martí. Fundamentals of scatter search and path relinking. *Control and Cybernetics*, 29(3):653–684, 2000.

[51] Monique Guignard and Siwhan Kim. Lagrangean decomposition: A model yielding stronger lagrangean bounds. *Mathematical programming*, 39(2):215–228, 1987.

[52] Mike Hewitt, George L Nemhauser, and Martin WP Savelsbergh. Combining exact and heuristic approaches for the capacitated fixed-charge network flow problem. *INFORMS Journal on Computing*, 22(2):314–325, 2010.

[53] Kaj Holmberg and Di Yuan. A Lagrangian heuristic based branch-and-bound approach for the capacitated network design problem. *Operations Research*, 48(3): 461–481, 2000.

[54] Naoto Katayama. A combined capacity scaling and local branching approach for capacitated multi-commodity network design problem. *Far East Journal of Applied Mathematics*, 92(1):1, 2015.

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

[56] Dukwon Kim and Panos M Pardalos. A solution approach to the fixed charge network flow problem using a dynamic slope scaling procedure. *Operations Research Letters*, 24(4):195–203, 1999.

[57] Dukwon Kim, Xinyan Pan, and Panos M Pardalos. An enhanced dynamic slope scaling procedure with tabu scheme for fixed charge network flow problems. *Computational Economics*, 27(2-3):273–293, 2006.

[58] Scott Kirkpatrick, C Daniel Gelatt, Mario P Vecchi, et al. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[59] Mikko Kivelä, Alex Arenas, Marc Barthelemy, James P Gleeson, Yamir Moreno, and Mason A Porter. Multilayer networks. *Journal of Complex Networks*, 2(3): 203–271, 2014.

[60] Georg Kliewer and Larissa Timajev. Relax-and-cut for capacitated network design. In Gerth Stølting Brodal and Stefano Leonardi, editors, *Algorithms–ESA 2005, 13th Annual European Symposium*, pages 47–58. Springer, 2005.

[61] Arnaud Knippel and Benoit Lardeux. The multi-layered network design problem. *European Journal of Operational Research*, 183(1):87–99, 2007.

[62] Arie MCA Koster, Sebastian Orlowski, Christian Raack, Georg Baier, and Thomas Engel. Single-layer cuts for multi-layer network design problems. In Edward Wasil S. Raghavan, Bruce Golden, editor, *Telecommunications Modeling, Policy, and Technology*, chapter 1, pages 1–23. Springer Science and Business Media, 2008.

[63] Cheng-Chang Lin. The integrated secondary route network design model in the hierarchical hub-and-spoke network for dual express services. *International Journal of Production Economics*, 123(1):20–30, 2010.

[64] Thomas L Magnanti and Richard T Wong. Network design and transportation planning: Models and algorithms. *Transportation Science*, 18(1):1–55, 1984.

[65] Thomas L Magnanti, Prakash Mirchandani, and Rita Vachani. Modeling and solving the two-facility capacitated network loading problem. *Operations Research*, 43 (1):142–157, 1995.

[66] Sara Mattia. Solving survivable two-layer network design problems by metric inequalities. *Computational Optimization and Applications*, 51(2):809–834, 2012.

[67] Sara Mattia. A polyhedral study of the capacity formulation of the multilayer network design problem. *Networks*, 62(1):17–26, 2013.

[68] Anne Mercier and François Soumis. An integrated aircraft routing, crew scheduling and flight retiming model. *Computers & Operations Research*, 34(8):2251–2265, 2007.

[69] Anne Mercier, Jean-François Cordeau, and François Soumis. A computational study of Benders decomposition for the integrated aircraft routing and crew scheduling problem. *Computers & Operations Research*, 32(6):1451–1476, 2005.

[70] Michel Minoux. Networks synthesis and optimum network design problems: Models, solution methods and applications. *Networks*, 19(3):313–360, 1989.

[71] Michel Minoux. Discrete cost multicommodity network optimization problems and exact solution methods. *Annals of Operations Research*, 106(1-4):19–46, 2001.

[72] Carlos Obreque, Macarena Donoso, Gabriel Gutiérrez, and Vladimir Marianov. A branch and cut algorithm for the hierarchical network design problem. *European Journal of Operational Research*, 200(1):28–35, 2010.

[73] Sebastian Orlowski. *Optimal design of survivable multi-layer telecommunication networks*. PhD thesis, Technische Universität Berlin, 2009.

[74] Sebastian Orlowski and Roland Wessäly. An integer programming model for multi-layer network design. Technical Report 04-49, ZIB, Berlin, 2004.

[75] Sebastian Orlowski, Christian Raack, Arie MCA Koster, Georg Baier, Thomas Engel, and Pietro Belotti. Branch-and-cut techniques for solving realistic two-layer network design problems. In Xavier Muñoz Arie Koster, editor, *Graphs and Algorithms in Communication Networks*, chapter 3, pages 95–118. Springer Berlin Heidelberg, 2010.

[76] Manfred W Padberg. On the facial structure of set packing polyhedra. *Mathematical Programming*, 5(1):199–215, 1973.

[77] Dimitris C Paraskevopoulos, Tolga Bektaş, Teodor Gabriel Crainic, and Chris N Potts. A cycle-based evolutionary algorithm for the fixed-charge capacitated multi-commodity network design problem. 2013.

[78] Dimitris C Paraskevopoulos, Tolga Bektaş, Teodor Gabriel Crainic, and Chris N Potts. A cycle-based evolutionary algorithm for the fixed-charge capacitated multi-commodity network design problem. *European Journal of Operational Research*, 253(2):265–279, 2016.

[79] Christian Raack and Arie M.C.A. Koster. An integer set-packing problem arising in two-layer network design. In *Proceedings of INOC 2009*, 2009.

[80] Inmaculada Rodríguez-Martín and Juan José Salazar-González. A local branching heuristic for the capacitated fixed-charge network design problem. *Computers & Operations Research*, 37(3):575–581, 2010.

[81] Juan-José Salazar-González. Approaches to solve the fleet-assignment, aircraft-routing, crew-pairing and crew-rostering problems of a regional carrier. *Omega*, 43:71–82, 2014.

[82] Meinolf Sellmann, Georg Kliewe, et al. Lagrangian cardinality cuts and variable fixing for capacitated network design. In *Algorithms—ESA 2002*, pages 845–858. Springer, 2002.

[83] Shengzhi Shao, Hanif D Sherali, and Mohamed Haouari. A novel model and decomposition approach for the integrated airline fleet assignment, aircraft routing, and crew pairing problem. *Transportation Science*, 51(1):233–249, 2015.

[84] AL Soyster, Ben Lev, and W Slivka. Zero-one programming with many variables and few constraints. *European Journal of Operational Research*, 2(3):195–201, 1978.

[85] Duc Minh Vu, Teodor Gabriel Crainic, and Michel Toulouse. A three-phase matheuristic for capacitated multi-commodity fixed-cost network design with design-balance constraints. *Journal of heuristics*, 19(5):757–795, 2013.

[86] Masoud Yaghini, Mohadeseh Rahbar, and Mohammad Karimi. A hybrid simulated annealing and column generation approach for capacitated multicommodity network design. *Journal of the Operational Research Society*, 64(7):1010–1020, 2013.

[87] Vahid Zeighami and François Soumis. Combining Benders decomposition and column generation for integrated crew pairing and personalized crew assignment problems. Technical Report G–2017–41, Cahiers du GERAD, 2017.

[88] Endong Zhu, Teodor Gabriel Crainic, and Michel Gendreau. Scheduled service network design for freight rail transportation. *Operations Research*, 62(2):383 – 400, 2014.