Network Design with Facility Location

Approximation and Exact Techniques

vorgelegt von
Diplom-Mathematiker
Mohsen Rezapour
geboren in Mashhad

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

Doktor der Naturwissenschaften
– Dr. rer. nat. –

genehmigte Dissertation

Promotionsausschuss:

Vorsitzender:	Prof. Dr. Boris Springborn
Gutachter:	Prof. Dr. Andreas Bley
	Prof. Dr. Martin Skutella
	Prof. Dr. Mohammad R. Salavatipour

Tag der wissenschaftlichen Aussprache: 16. March 2015

Berlin 2015
D83

Abstract

In this thesis, we consider a family of problems that combine network design and facility location. Such problems arise in many practical applications in different fields such as telecommunications, transportation networks, logistic, and energy supply networks. In facility location problems, we want to decide which facilities to open and how to assign clients to the open facilities so as to minimize the sum of the facility opening costs and client connection costs. These problems typically do not involve decisions concerning the routing of the clients' demands to the open facilities; once we decided on the set of open facilities, each client is served by the closest open facility. In network design problems, on the other hand, we generally want to design and dimension a minimum-cost routing network providing sufficient capacities to route all clients' demands to their destinations. These problems involve deciding on the routing of each client's demand. But, in contrast to facility location problems, demands' destinations are predetermined. In many modern day applications, however, all these decisions are interdependent and affect each other. Hence, they should be taken simultaneously. A naive strategy that first decides which facilities to open, and then builds routing networks connecting clients to the open facilities, may lead to very poor solutions. This has motivated several new combined network design facility location problems which are studied in this thesis. We develop new algorithmic techniques and solution approaches for these problems, building on the known techniques for facility location and network design.

In a first line of work, we study problems that integrate buy-at-bulk network design into the classical facility location problem. More precisely, we consider generalizations of the facility location problem where multiple clients may share capacitated trees to connect to the open facilities instead of requiring direct links. The task is to open facilities (sinks) and route all clients's demands to the open facilities via a capacitated access network that is constructed by installing access cables of different costs and capacities. On the theoretical side, we present the first LP-based approximation algorithm for this problem and prove an upper bound of O(K) on the Integrality gap of the underlying LP. We also undertake the first computational study for this problem. To this end, we provide compact and exponential-sized formulations for the problem and develop a branch-cut-and-price algorithm allowing us to solve very large real-world instances of the problem.

In many real-world applications, particularly in telecommunications, there is the additional requirement to connect the open facilities via high bandwidth core cables. In

the simplest case, this leads to variants of the problem in which open facilities are connected via a tree-like core network that consists of infinite capacity cables. We analyze two fundamental versions of this problem: In the Buy-at-Bulk version of the problem, each access cable type has a fixed setup cost and a fixed capacity, whereas in the Deep-Discount problem version, each cable type has unlimited capacity but a traffic-dependent variable cost in addition to its fixed setup cost. The Buy-at-Bulk version arises in the planing of telecommunication networks, while the Deep-Discount problem version is motivated by applications in logistic and transportation networks. We develop the first constant-factor approximation algorithms for these connected versions. Using sampling techniques, we then improved the approximation factors. We were even able to prove a tighter bound of O(1) on the the Integrality gap of an LP formulation of the problem similar to one for the unconnected version.

The core network plays a primary role for the service availability and the service quality in telecommunications networks. Therefore it is common to require that the core networks are fault-tolerant and have short routing paths. In the final part of this thesis we focus on the core network design. To take these requirements into account, we introduce and study another network design facility location problem where the core network has to fulfill simultaneous survivability and hop-length restrictions between the chosen facilities. It is easy to see that it is already NP-hard to compute a constant-factor approximation for the problem. Hence, we focus our research on IP based techniques. We propose two strong extended formulations for the problem and devise a practically efficient branch-and-cut algorithm based on Benders decomposition for its solution.

Acknowledgements

This thesis would not have been possible without the support of many people. Most of all, I would like to thank my advisor, Andreas Bley, for his guidance, encouragement, and advice during the development of this work. I owe you many thanks for the confidence you put on me and your kind way of guidance. I am indebted to Martin Skutella for giving me the privilege of working in his working group. Thank you for all of your invaluable support and advice throughout my PhD. I am also very grateful to Mohammad Salavatipour for taking the third assessment of this thesis.

Going back a bit further in time, I am grateful to Mehdi Hashemi for his support and advice during my years at Amirkabir University of Technology and for the many things I learned from him.

My biggest thanks go to my coauthors Ashwin Arulselvan, Andreas Bley, Zachary Friggstad, Mehdi Hashemi, Mohammad Salavatipour, José Soto, and Wolfgang Welz, with whom I had the chance to work on different parts of this thesis. My work has greatly benefited from discussing with them. Moreover, I would like to thank Ashwin Arulselvan, Martin Groß, Majid Rezapour, Alexander Richter, and José Soto for carefully reading parts of this work and for giving comments that have been helpful to improve the presentation of this thesis.

The members of the COGA group have contributed immensely to my personal and professional time at Berlin. The group has been a source of friendships as well as good advice and collaboration. Thank you for the wonderful time I had in Berlin. My special thanks go to Martin Groß for being a great office-mate and friend during the past 3 years.

I would like to thank Mohammad Salavatipour, Zachary Friggstad, and Babak Behsaz for all helpful discussions we had during my stay in Edmonton. Working with you was an invaluable experience for me.

I want to thank Dorothea Kiefer for her kind support concerning all the administrative paperwork and also Ralf Hoffman for his technical support concerning all my computer system related problems.

Lastly, I would like to thank my parents for all their love, encouragement, and support. I could not have asked for better parents. And most of all I would like to thank my

loving, encouraging, and patient wife, Anna, whose faithful support during my Ph.D. is so appreciated. Thank you. Words cannot express the gratitude I owe you.

Mohsen Rezapour
Berlin, January 2015

This research was supported by the Deutsche Forschungsgemeinschaft within the research training group 'Methods for Discrete Structures' (GRK 1408)

Contents

List of Figures

Chapter 1

Introduction

A wide range of combinatorial optimization problems occur in the field of designing telecommunication networks. A typical telecommunication network, in its simplest form, consists of a *backbone network* with (almost) unlimited capacity on the links and several local *access networks*. In such a network, the traffic originating from the clients is sent through access networks to gateways or *core nodes*, which provide routing functionalities and access to the backbone network. The backbone then provides the *connectivity* among the core nodes, which is necessary to route the traffic further towards its destination. Designing such a network involves *locating* the core nodes, building a high bandwidth network connecting core nodes, and building an access network by installing access cables of different costs and capacities to route the traffic from the clients to the selected core nodes. This has motivated several new combined network design facility location problems we focus on in this thesis.

We study generalizations of the facility location problem where multiple clients may share capacitated trees to connect to the open facilities instead of requiring direct links. In addition, we study variants of the problem in which open facilities are connected via a tree-like core network that consists of infinite capacity cables. We notice that the performance measures for the effectiveness of the telecommunication network not only contain *cost*, but also contain *survivability* and *quality of service*. Therefore, to take these two requirements into account, we study another network design facility location problem where the core network has to fulfill simultaneous survivability and hop-length restrictions between the chosen facilities.

In this thesis, we obtain several new approximation and exact algorithms for the considered NP-hard problems, using a wide range of techniques from computer science and operations research.

We start this introductory chapter with reviewing some of the most common approaches for coping with NP-hard optimization problems in practice, which is indeed the main purpose of this thesis. We review related facility location and network design problems in Section 1.2. We then introduce several new network design problems with applications in telecommunication networks in Section 1.3. We end this chapter by summarizing our contributions in this thesis.

1.1 Solution Approaches Used in This Thesis

There are many important optimization problems in practice that are difficult to solve optimally. In fact, many of those problems are known as **NP-hard** problems. Notice that no polynomial-time algorithm exists that solves any NP-hard problem optimally, assuming $\mathbf{P} \neq \mathbf{NP}$; we refer the reader to book by Garey and Johnson (1979) for a thorough introduction to complexity theory. Therefore, we cannot expect to find polynomial-time algorithms for a large number of the real world optimization problems (e.g., see Section 1.3) that are required to be solved. To cope with such hard problems, in this thesis we use some of the most common approaches, namely *approximation algorithms*, *heuristics*, and *Integer Programming*.

Approximation Algorithms

For some of the NP-hard problems, one may devise polynomial-time algorithms to solve them efficiently at the cost of providing solutions whose costs are guaranteed to be within a constant factor of the optimal solutions' costs. This leads to the notion of approximation algorithms.

We call algorithm A an α-approximation algorithm for a minimization problem, if A runs in polynomial time and returns a solution of cost no more than α times of the optimum. The value $\alpha > 1$ is known as the *approximation ratio* of the algorithm. There are several powerful techniques (e.g., *greedy procedure*, *primal-dual*, *dual-fitting*, *LP-rounding*, *sampling*) that can be used in the design of approximation algorithms; see books by Vazirani (2001); Williamson and Shmoys (2011) for an introduction to these approximation techniques. The class **APX** is the set of NP problems that allow *constant-factor approximation algorithms* (or, more precisely, those that allow approximation algorithms with an approximation ratio bounded by a constant). We will apply some of these techniques to design new approximation algorithms for the problems under consideration in Chapters 3 and 4.

Of course not all the NP-hard problems allow constant-factor approximation algorithms. In fact, there are problems (e.g. *traveling salesman*, *set cover*) which are so hard that even finding constant-factor approximation for them can be shown to be NP-hard; e.g., the network design problem considered in Chapter 5. This is where heuristics, for example, can come into play to solve such complex problems.

Heuristics

Algorithms that run in polynomial time and provide reasonably good solutions are called heuristics. We should remark that heuristics in contrast to approximation algorithms do not come always with a guarantee on the quality of their solution. In fact, heuristic algorithms often work well on most of the instances (in particular on those instances at hand), but perhaps not on all possible instances of a problem.

Heuristics can be classified into those which gradually build a feasible solution by a sequence of decisions, called *constructive algorithms* (e.g., greedy algorithm; see Section 2.3.1), and those which take a solution as input and produce a new improved solution by performing a sequence of operations, called *improvement algorithms* (e.g., local search algorithm). We refer the reader to the book by Hromkovic (2010) for an introduction to heuristic techniques.

One may still solve the NP-hard problems exactly, but of course not in a polynomial time. This leads us to the field of Integer Programming and many techniques there.

Integer Programming

Integer Programming (IP) is the natural way of modeling many real-world problems, including numerous NP-hard problems. Most of the techniques used to solve IPs are based on solving LP relaxations. In fact, this is because solving linear programs is much easier than solving integer programs. More precisely, solving integer programs is NP-hard (one can model some NP-hard problems as integer programs), whereas linear programs can be solved in polynomial time (see Karmarkar, 1984).

Integer linear programs are typically solved by using *Branch-and-Bound*, a widely known exact solution technique which creates a tree of nodes, called the *Branch-and-Bound tree*, as follows. The LP relaxation of the original problem is at the root node. The other nodes of the tree represent subproblems that each guarantee the integrality of a subset of variables. In fact, *Branch-and-Bound* handles integrality by exploring and branching this tree. The *cutting plane* method is another exact technique one can use to solve an IP. It works by iteratively solving the LP relaxation of the given IP which is gradually

refined by adding more linear constraints called *cuts*. The *Branch-and-Bound* technique when used together with cutting plane methods is called *Branch-and-Cut*. This is a very successful technique for solving IP problems, in particular for IP programs of moderate size. We refer the reader to book by Wolsey (1998) for an introduction to the subject.

There exist successful techniques (e.g. *column generation* (see Dantzig and Wolfe, 1960; Lübbecke and Desrosiers, 2005) and *Benders decomposition* (see Benders, 1962)) that may be used to attack even very large scale problems by exploiting some specific structures of the problems; e.g., see Chapters 2 and 5. The Branch-and-Bound technique, when used together with column generation and cut separation is called *Branch-Cut-and-Price*.

1.2 Facility Location and Network Design Problems

In this section we review some realted results on facility location and network design problems.

1.2.1 Facility Location

One of the most well-studied problems in the operations research and computer science literature is the *facility location* problem. In this problem, in its simplest form, we are given a set of clients and facilities, an opening cost associated with each facility, and a nonnegative distance between any pair of elements. The task is to open a subset of the facilities and assign each client to an open facility, such that the sum of opening costs and the distance between each client and the facility it is assigned to is minimized. This class of problems has a wide range of applications such as deciding placement of factories, warehouses, libraries, fire stations, hospitals, and base station for mobile phone service.

Uncapacitated Facility Location

The most widely studied model in discrete facility location is the *metric Uncapacitated Facility Location* (**UFL**) problem.

Problem 1.1. **Metric Uncapacitated Facility Location (UFL)**

Input

- A finite set of locations V
- Potential facilities $F \subseteq V$ with opening costs $\mu_i \in \mathbb{Z}_{\geq 0}$, $i \in F$

- Clients $D \subseteq V$
- Metric cost $l_{ij} \in \mathbb{Z}_{\geq 0}$, for assigning client $j \in D$ to facility $i \in F$

Solution

- Open facilities $F^* \subseteq F$
- Assignment $\sigma^*(j) : D \to F^*$

Goal

$$\min \sum_{i \in F^*} \mu_i + \sum_{j \in D} l_{\sigma^*(j)j}$$

The UFL problem is widely studied in the computer science literature. A greedy algorithm with $O(\log(n))$-approximation guarantee for the UFL problem (similar to one for the set cover problem (see Chvatal, 1979)) was given by Hochbaum (1982), where n is the number of clients. The first constant factor approximation algorithm for UFL was given by Shmoys et al. (1997), and was based on LP rounding and a filtering technique due to Lin and Vitter (1992). Since then this factor has been gradually reduced to 1.488 (Li, 2013) by a long series of papers (we point the reader to a survey by Vygen (2005)). A number of elegant and powerful techniques have been used in the design of these approximation algorithms, e.g. *LP-rounding* (Sviridenko, 2002; Chudak and Shmoys, 2003), *greedy procedure* (Charikar and Guha, 1999; Guha and Khuller, 1999), *primal-dual* (Jain and Vazirani, 2001), and *dual-fitting* (Jain et al., 2003). There are also results that combine the above techniques. For example, Jain et al. (2002) presented a greedy algorithm that uses the LP-relaxation implicitly to obtain a lower bound for a primal-dual analysis, Mahdian et al. (2006) who use Jain's algorithm (Jain et al., 2002) and a greedy procedure to get an approximation factor of 1.52, Byrka and Aardal (2010) who combine an LP-rounding based algorithm and Jain's algorithm (Jain et al., 2002) to obtain a 1.5-approximation algorithm, and finally Li (2013) who combines the algorithm presented by Byrka and Aardal (2010) and Jain's algorithm (Jain et al., 2002) to achieve an approximation guarantee of 1.488. He uses a randomized selection in Byrka's algorithm.

On the hardness side, Guha and Khuller (1999) showed (by a reduction from the set cover problem) that it is hard to approximate UFL within a factor of 1.463, assuming $\mathbf{NP} \not\subseteq \mathbf{DTIME}(n^{\log \log n})$. Later, this was generalized by Jain et al. (2002) who show that the existence of a (λ_f, λ_c)-approximation algorithm with $\lambda_c < 1 + 2e^{-\lambda_f}$ implies $\mathbf{NP} \subseteq \mathbf{DTIME}(n^{\log \log n})$. An algorithm is a (λ_f, λ_c)-approximation algorithm if the solution the algorithm delivers has total cost at most $\lambda_f \cdot F^* + \lambda_c \cdot C^*$, where F^* and C^* are the facility and the assignment cost of an optimal solution, respectively.

Connected Facility Location

An interesting variant of UFL occurs in communication networks (in particular in telecommunications) where facilities want to communicate with each other, and hence a connectivity among facilities (via high bandwidth links) is required. This leads to a variant of UFL that is called *connected facility location*.

Problem 1.2. **Connected Facility Location (ConFL)**

Input

- Undirected graph $G = (V, E)$
- Metric edge lengths $c_e \in \mathbb{Z}_{\geq 0}$, $e \in E$
- Potential facilities $F \subseteq V$ with opening costs $\mu_i \in \mathbb{Z}_{\geq 0}$, $i \in F$
- Clients $D \subseteq V$ with demands $d_j \in \mathbb{Z}_{>0}$, $j \in D$
- Core cable type with infinite capacity and setup cost (per unit length) $M > 1$

Solution

- Open facilities $F^* \subseteq F$
- Steiner tree $T^* \subseteq E$ of core cables spanning F^*
- Assignment $\sigma^*(j) : D \to F^*$

Goal

$$\min \sum_{i \in F^*} \mu_i + \sum_{e \in T^*} M c_e + \sum_{j \in D} d_j \cdot l(j, \sigma^*(j))$$

Where $l(u, v)$ is the shortest path distance between vertices u and v in G.

Several approximation algorithms for this problem have been proposed in the computer science literature. Gupta et al. (2001) obtain a 10.66-approximation for this problem, based on LP rounding. This later was improved by Swamy and Kumar (2004) who obtain an approximation ratio of 8.55, using a primal-dual algorithm. Then, using LP rounding techniques, the approximation factor was improved to 8.29 in the general case and to 7 in case all opening costs are equal by Hasan et al. (2008). A 6.55-approximation primal-dual algorithm for the ConFL problem was proposed by Jung et al. (2009) which can be viewed as a refinement of the algorithm given by Swamy and Kumar (2004). Finally, using *sampling techniques*, the guarantee was reduced to 4 by Eisenbrand et al. (2010), and to 3.19 by Grandoni and Rothvoß (2011). Both algorithms by Eisenbrand et al. (2010) and Grandoni and Rothvoß (2011) can be viewed as a randomized decomposition of the ConFL problem into the the facility location and the Steiner tree problems. The

analysis (of both algorithms) exploits the core detouring scheme (see Eisenbrand et al., 2010) to bound the cost of assigning the clients to open facilities. Eisenbrand et al. (2010) first relax the requirement to connect the open facilities and solve a corresponding UFL instance. They then randomly sample clients and thereby they open a subset of facilities from among those used by the UFL solution. Finally, they construct a Steiner tree over sampled clients and augment it so as to connect the opened facilities; whereas the algorithm by Grandoni and Rothvoß (2011) first randomly samples clients and constructs a Steiner tree connecting them. Then it solves a manipulated instance of UFL in which the opening cost of each facility is increased by the cost of connecting that facility to the Steiner tree on sampled clients. Applying this modification to the algorithm by Eisenbrand et al. (2010), they could provide an improved approximation guarantee of 3.19 for the problem. In Chapter 3 we extend this framework to the Buy-at-Bulk version of ConFL.

On the hardness side, the results by Guha and Khuller (1999) for the facility location problem can be adapted to prove that ConFL is hard to approximate within 1.463 (unless $\mathbf{NP} \subseteq \mathbf{DTIME}(n^{\log \log n})$), as observed by Grandoni and Rothvoß (2011).

The ConFL problem is also widely studied in the operation research community. The first heuristic algorithm for the ConFL problem was given by Ljubić (2007) who combines the variable neighborhood search with a reactive tabu-search to obtain her heuristic algorithm. A greedy randomized adaptive search with a multi-start iterative construction was also proposed by Tomazic and Ljubić (2008). Later, the ConFL problem was formulated as a directed Steiner tree problem with a unit degree constraint by Bardossy and Raghavan (2010) who propose a dual-based heuristic for the problem. It is worth noting that their dual-based algorithm is able to provide both upper and lower bounds (by returning a primal feasible solution together with a dual feasible solution) for a given instance. This can be used to assess the quality of the solutions. We refer the reader to the paper by Gollowitzer and Ljubić (2011) for an overview of formulations and exact approaches for ConFL.

A special case of the ConFL problem where all opening costs are 0 and facilities may be opened anywhere ($F = V$) is called the *Single-Sink Rent-or-Buy* problem.

Problem 1.3. **Single-Sink Rent-or-Buy (SSRoB)**

Input

- Undirected graph $G = (V, E)$
- Metric edge lengths $c_e \in \mathbb{Z}_{\geq 0}$, $e \in E$
- Clients $D \subseteq V$ with demands $d_j \in \mathbb{Z}_{>0}$, $j \in D$
- Integer $M > 1$

Solution

- Open facilities $F^* \subseteq V$
- Steiner tree $T^* \subseteq E$ of bought edges spanning F^*
- Assignment $\sigma^*(j) : D \to F^*$

Goal

$$\min \sum_{e \in T^*} M c_e + \sum_{j \in D} d_j \cdot l(j, \sigma^*(j))$$

The SSRoB problem is well-studied in the literature (see Karget and Minkoff, 2000; Gupta et al., 2001; Swamy and Kumar, 2004; Gupta et al., 2003; Eisenbrand et al., 2010). The approximation algorithms proposed for the ConFL problem obviously work for SSRoB too, however some of them may come with improvements in their approximation guarantees. For example, the algorithms by Gupta et al. (2001), Swamy and Kumar (2004), and Eisenbrand et al. (2010) guarantee improved approximation ratios of 9.002, 4.55, and 2.92, respectively, for this special case.

On the hardness side, Grandoni and Rothvoß (2011) obtained a 1.278-inapproximability bound for SSRoB.

1.2.2 Network Design

Network Design is one of the central topics in both computer science and operations research literature. The network design problems, in their simplest forms, only deal with building minimum-cost networks which satisfy a certain connectivity requirement between a set of terminals. This class of network design problems, known as *Connectivity*, has a large number of practical applications; e.g., in the design of communication networks.

Another important class of network design problems arise, for example, in telecommunication networks where one has to design a network by installing cables of different costs and capacities to route traffic of a set of demand sources to a (or, multiple) sink(s); while high-capacity cables are more expensive than low-capacity cables and they satisfy economies of scale (that is, the cost per unit capacity decreases from small to large cables). We refer to this class of the problems as *Buy-at-Bulk Network Design* problems.

In this section we briefly discuss about problems and results related to these two important classes of network design problems.

Connectivity

The most general version of the connectivity problems is called the *survivable network design* problem.

Problem 1.4. **Survivable Network Design (SND)**

Input

- Undirected graph $G = (V, E)$
- Edge lengths $c_e \in \mathbb{Z}_{\geq 0}$, $e \in E$
- A set of demand pairs $D \subseteq V \times V$
- An integer connectivity requirement $r_{uv} > 0$ for each pair of $(u, v) \in D$

Solution

- Edge set $E^* \subseteq E$ containing r_{uv} edge-disjoint (u, v)-paths for each $(u, v) \in D$

Goal

$$\min \sum_{e \in E^*} c_e$$

We note that if these paths are required to be vertex-disjoint, the problem is referred to as Vertex-Disjoint SND (**VD-SND**); and when all demand pairs D have a common vertex, say r, the problem is referred to as *rooted* SND (**rSND**).

These problems are well studied in the literature. The first non-trivial approximation algorithm for SND was given by Williamson et al. (1993, 1995) who gave a $2K$-approximation, where $K = max_{(u,v) \in D}\ r_{u,v}$. The factor was later improved to $2H_K$ by Goemans et al. (1994), where $H_K = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{K}$. Finally a 2-approximation for this problem was obtained by Jain (2001) who introduced the influential iterated rounding technique to design his algorithm. Then, this result has been generalized to the case of VD-SND when $r_{uv} \in \{0, 1, 2\}$ by Fleischer et al. (2006).

We refer the reader to the paper by Schrijver (2003) for exact algorithms.

The *Steiner tree* problem is one of the most fundamental network design problems.

Problem 1.5. **Steiner Tree (ST)**

Input

- Undirected graph $G = (V, E)$
- Edge lengths $c_e \in \mathbb{Z}_{\geq 0}$, $e \in E$
- Terminals $T \subseteq V$

Solution

- Tree $S^* \subseteq E$ spanning terminals T

Goal

$$\min \sum_{e \in S^*} c_e$$

This problem can be viewed as the special case of the SND problem when $r_{uv} = 1$ iff $u, v \in T$. The Steiner tree problem is **NP**-hard, even when edge costs are either 1 or 2 (see Bern and Plassmann, 1989). The minimum cost terminal spanning tree on the fully connected graph of the metric closure containing only the terminals as vertices and the edges between them is well-known to be a 2-approximation for the Steiner tree problem; see the paper by Takahashi and Matsuyama (1980). More specifically, it is a $(2 - \frac{2}{|R|})$-approximation. This factor was later improved to $(1 + \frac{\ln 3}{2}) \approx 1.55$ by Robins and Zelikovsky (2000), and then to $(\ln 4 + \epsilon) \approx 1.39$ by Byrka et al. (2010) who use the iterative rounding technique to obtain the currently best approximation ratio for the ST problem.

On the negative side, Chlebík and Chlebíková (2008) show that there is no $(\frac{96}{95} - \epsilon)$-approximation algorithm for the Steiner tree problem, unless **P** = **NP**. Note that the same inapproximability result extends to the SND problem, too.

In some real world networks (e.g., telecommunications), to guarantee a desired level of quality of service, one has to pose a limit on the number of edges (hops) of the routing paths. This leads to an interesting variant of the SND problem, called the *Survivable Hop-Constrained Network Design* problem.

Problem 1.6. **Survivable Hop-Constrained Network Design (SHND)**

Input

- Undirected graph $G = (V, E)$
- Edge lengths $c_e \in \mathbb{Z}_{\geq 0}$, $e \in E$
- A set a of demand pairs $D \subseteq V \times V$
- Integer connectivity requirement $r_{uv} > 0$ for each pair of $(u, v) \in D$
- Integer hop limit $H > 0$

Solution

- Edge set $E^* \subseteq E$ containing r_{uv} edge-disjoint (u, v)-paths with at most H edges for each $(u, v) \in D$

Goal

$$\min \sum_{e \in E^*} c_e$$

We call the rooted case when all demand pairs D have a common vertex *rooted Survivable Hop-Constrained Network Design* (**rSHND**). Given a set of terminals $T \subseteq V$; we call a special case of the SHND problem where $r_{uv} = 1$ iff $u, v \in T$ as *Hop-Constrained Steiner Tree* (**HST**).

The HST problem is not in **APX**, even if the edge weights satisfy the triangle inequality (see Manyem, 2009). Recall that **APX** is the set of NP optimization problems that allow constant-factor approximation algorithms. Therefore this inapproximability result can be extended to SHND and rSHND, too.

The first IP formulation for SHND has been presented by Huygens et al. (2007) who only consider the case with $H \leq 4$ and $r_{uv} = 2$ for all (u, v) in D. Later, a more general (but rooted) version of this problem, with uniform connectivity requirement $K > 1$ and $H > 1$, has been considered by Botton et al. (2013) who present a branch-and-cut algorithm to solve the problem. The formulation given by Botton et al. (2013) then has been strengthened by Mahjoub et al. (2013) who presented an extended formulation for the rSHND by introducing additional variables which indicate the distance of each demand node to the root.

Buy-at-Bulk Network Design

The most general form of the buy-at-bulk problem is called *Non-uniform Buy-at-Bulk Network Design* problem, and is defined as follows.

Problem 1.7. **Non-uniform Buy-at-Bulk Network Design (Non-uniform BBND)**

Input

- Undirected graph $G = (V, E)$
- Edge lengths $c_e \in \mathbb{Z}_{\geq 0}$, $e \in E$
- A set a of source-sink pairs $D \subseteq V \times V$ with demands $d_{(u,v)} \in \mathbb{Z}_{>0}$, $(u, v) \in D$
- A sub-additive monotone function $f_e : \mathbb{Z}_{\geq 0} \to \mathbb{R}_{\geq 0}$ which gives the cost (per unit length) of transporting demand along edge e

Solution

- Edges set $E^* \subseteq E$ such that, all pairs (u, v) are connected in $G[E^*]$

Goal

$$\min \sum_{e \in E^*} f_e(\hat{x}_e) \cdot c_e$$

Where $\hat{x}_e$ denotes the total units of demand routed along edge e.

We refer to the problem as *single-sink* case when all source-sink pairs share the same sink terminal. When multiple sinks are allowed and the sink terminals can be any vertices in the graph, we refer to the problem as *multi-sink* case. We call the case when $f_e = f$ for all $e \in E$ as the *uniform* case.

Buy-at-bulk network design problems have been considered in both operations research and computer science literature. The first non-trivial approximation algorithm to the non-uniform BBND problem was given by Charikar and Karagiozova (2005) who obtained an approximation ratio of $e^{O(\sqrt{\log |D| \log\log |D|})} \cdot \log \bar{d}$, where $\bar{d} = \sum_{(u,v)\in D} d_{(u,v)}$. They also obtained a $O(\log^2 |D|)$-approximation for the single-sink case. Their algorithm is a simple randomized greedy algorithm based on shortest-path approach. In Section 2.3.1, we will use an adapted version of this greedy algorithm to obtain promising initial solutions for our Branch-Cut-and-Price algorithm. The first poly-logarithmic approximation for non-uniform BBND was given by Chekuri et al. (2010) who obtained an approximation ration of $O(\min\{\log^3 |D| \cdot \log \bar{d}, \log^5 |D| \log\log |D|\})$, which was then improved to $O(\log^3 |D|)$ for the case when demand values can be polynomially bounded with respect to $|D|$ by Kortsarz and Nutov (2009).

For the uniform case, a randomized $O(\log^2 n)$-approximation was obtained by Awerbuch and Azar (1997), where n is the number of vertices in the graph. Their algorithm is based on the tree-embeddings (Bartal, 1996). Thus the approximation ratio naturally can be improved to $O(\log n \log\log n)$ and then to $O(\log n)$ using the improved results on approximation of metrics by trees; see the papers by Bartal (1998); Fakcharoenphol et al. (2003).

Regarding the single-sink case, the first result is an $O(\log |D|)$ randomized approximation algorithm due to Meyerson et al. (2000). They referred to the problem as *Cost-Distance*. The algorithm by Meyerson et al. (2000) was then derandomized by Chekuri et al. (2001) who use an LP rounding approach, establishing an integrality gap of $O(\log |D|)$.

On the hardness side, Andrews (2004) obtained a hardness result of $\Omega(\log^{\frac{1}{2}} n)$ and $\Omega(\log^{\frac{1}{4}} n)$ for the non-uniform and uniform cases, respectively. Moreover, a hardness result of $\Omega(\log\log n)$ is obtained by Chuzhoy et al. (2008) for the single-sink case.

The uniform single-sink case of the BBND problem under a *cable capacity* cost model is called *Single-Sink Buy-at-Bulk Network Design*.

Problem 1.8. **Single-Sink Buy-at-Bulk Network Design (SSBB)**

Input

- Undirected graph $G = (V, E)$
- Edge lengths $c_e \in \mathbb{Z}_{\geq 0}$, $e \in E$

- A set a of demands $D \subseteq V$ with demands $d_j \in \mathbb{Z}_{>0}$, $j \in D$
- A sink vertex $t \in V$
- Cable types K with
  - Capacity $u_k \in \mathbb{Z}_{>0}$, $k \in K$
  - Setup cost (per unit length) $\sigma_k \in \mathbb{Z}_{\geq 0}$, $k \in K$

  $\sigma_1 < ... < \sigma_K \leq M$ and $\frac{\sigma_1}{u_1} > ... > \frac{\sigma_K}{u_K}$

Solution

- Edges set $E^* \subseteq E$ with an cable installation $\alpha : E^* \times K \to \mathbb{Z}_{\geq 0}$ such that, all demands in D can be sent to t via the resulting capacitated network

Goal

$$\min \sum_{e \in E^*} \sum_{l \in K} \sigma_k c_e \alpha_{e,l}$$

We refer to the problem as *splittable* when the demand of each client is allowed to be routed along several paths. When the entire demand of each client must be routed along a single path, we refer to the problem as *unsplittable*.

Several approximation algorithms for this problem have been proposed in the computer science literature. For the unsplittable case, Garg et al. (2001) developed an $O(K)$ approximation, using LP rounding techniques. The first constant factor approximation for this problem is due to Guha et al. (2001, 2009). Talwar (2002) showed that an LP formulation of this problem has a constant integrality gap and provided a 216 approximation algorithm. Using sampling techniques, this factor was reduced to 145.6 by Jothi and Raghavachari (2004), and later to 40.82 by Grandoni and Rothvoß (2010).

For the splittable case, Gupta et al. (2003) presented a simple 76.8-approximation algorithm using random-sampling techniques. Unlike the algorithms mentioned above, their algorithm does not guarantee that the solution is a tree. Modifying Gupta's algorithm, the approximation for the splittable case was later reduced to 65.49 by Jothi and Raghavachari (2004), and then to 24.92 by Grandoni and Rothvoß (2010).

On the negative side, a 1.278-inapproximability bound for SSBB may be obtained trivially from inapproximability of the Single-Sink Rent-or-Buy problem (Grandoni and Rothvoß, 2011).

The SSBB problem is also well studied in the operation research literature. However, in the operation research literature, it is mostly known as *single-source network loading problem* (e.g. Ljubić et al., 2012), or (in the case of telecommunication network planning) as *Local Access Network Design Problem (LAN)* (e.g. Salman et al., 2008).
The LAN problem with only two cable types under the assumption that the solution

must be a tree (with unsplittable flow) was considered by Randazzo et al. (2001) who provide a multicommodity flow based formulation for the problem and solve it by applying Benders' decomposition. The LAN problem with multiple cable types was then considered by Salman et al. (2008). They apply flow-based MIP formulations and work with relaxations obtained by approximating the capacity step cost function by its lower convex envelope to provide a special branch-and-bound algorithm for LAN design. Their technique was later reformulated as a stylized branch-and-bound algorithm by Raghavan and Stanojević (2006). Finally, a stronger multicommodity flow formulation for the problem was considered by Ljubić et al. (2012) who applied a branch-and-cut algorithm based on Benders decomposition for solving the problem.

1.3 Problems Considered in This Thesis

We consider two classes of the network design facility location problems.

1.3.1 Facility Location in Network Design

In this section we introduce several new network design facility location problems that integrate buy-at-bulk network design and facility location aspects in order to aptly model the practical needs when planning certain types of networks in telecommunications, transportation networks, logistic, and energy supply networks.

A Motivating Example

Consider the following example in the planning of point-to-point optical access networks in telecommunications. In optical access networks, demand nodes (clients) are connected to their respective central offices (facilities) in a tree-like fashion, where a combination of different optical cable types are installed on the edges of these trees. This allows for multiple fibers emanating from different clients to share a single, larger cable and the same trunk on their common path towards their common central office. In fact the obvious advantage of using trees, compared to using an individual cable and trunk for each client, is a considerable reduction in cable and trunk cost. The central offices provide switching and routing functionalities and are connected amongst each other or to some higher network level. This so-called core (or metro) network consists of connections of (almost) unlimited capacity, which may follow the same trunks as the access network connections, but use dedicated core cables for technical and management simplicity; see Figure 1.1.

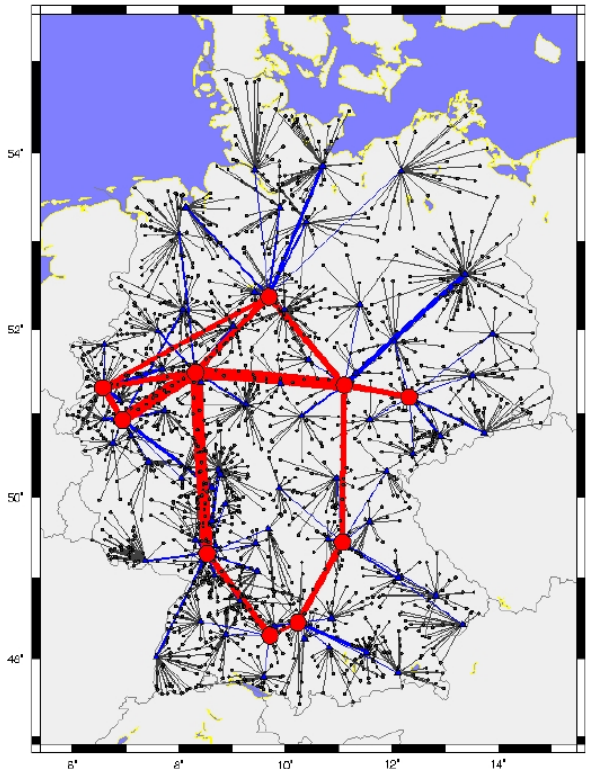


Figure 1.1: A fiber optic network

When planning the deployment of such a network, one generally has to decide where to set up central offices, how to lay out the access trees connecting the clients to these offices and core network among the offices, and which cable types to use on the edges of the network. One of the main objectives in this planning is to minimize the total network cost, which is comprised by cost for setting up the central offices and the cost for laying out the cables. More precisely a network planner is expected to decide which facilities (central offices) to open, where to install core cables to obtain a Steiner tree connecting the open facilities, and where to install which access cable type to obtain a forest-like access network whose capacities suffice to simultaneously route all client demands to open facilities. We will model this as a *Buy-at-Bulk Connected Facility Location* problem denoted by **BBConFL**.

Definitions

Formally, in this problem we are given a complete graph $G = (V, E)$ with nonnegative edge lengths $c_e \in \mathbb{Z}_{\geq 0}$, $e \in E$ satisfying triangle inequality; a set $F \subseteq V$ of facilities with opening costs $\mu_i \in \mathbb{Z}_{\geq 0}$, $i \in F$; and a set of clients $D \subseteq V$ with demands $d_j \in \mathbb{Z}_{>0}$, $j \in D$. We are also given K *types* of *access cables* that may be used to connect clients to open facilities. A cable of type i has capacity $u_i \in \mathbb{Z}_{>0}$ and cost (per unit length) $\sigma_i \in \mathbb{Z}_{\geq 0}$.

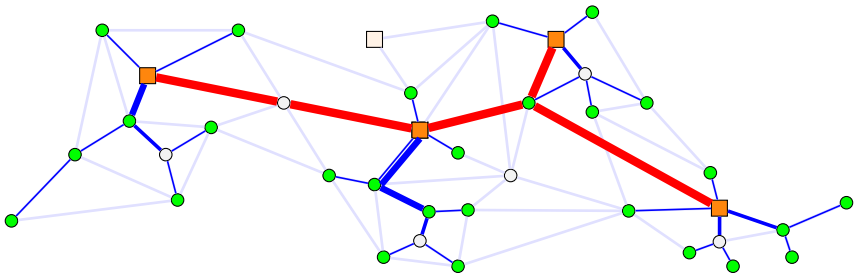


Figure 1.2: A feasible solution for the BBConFL problem, where the square nodes (in orange) represent (open) facilities and the circle nodes in green represent clients.

Furthermore, we are given an extra type of cable, called *core cable*, having a cost (per unit length) of $M > \sigma_K$ and infinite capacity, which may be used to connect the open facilities with each other. We assume that access cable types obey economies of scale. That is, $\sigma_1 < \sigma_2 < \cdots < \sigma_K$ and $\frac{\sigma_1}{u_1} > \frac{\sigma_2}{u_2} > \cdots > \frac{\sigma_K}{u_K}$.

A feasible solution (Figure 1.2) or BBConFL consists of (1) A subset $F_0 \subseteq F$ of facilities to open; (2) a Steiner tree of G (core network) connecting all open facilities via core cables; and (3) a forest (access network) connecting all clients to the open facilities. Furthermore, on each edge of this forest we have to specify a list of possibly multiple copies and types of access cables to install, in such a way that the entire demand of each client can be routed along a single path to an open facility. The objective of BBConFL is to minimize the total cost of opening facilities, and constructing core and access networks; where the cost for using edge e in the core network is Mc_e, and the cost for installing a single copy of access cable of type i on an edge e is $\sigma_i c_e$. It is worth noting that we are allowed to install core cables on edges incident to closed facilities, to clients, or even to nodes in $V \setminus (F \cup D)$. Nevertheless, the demand from a client to an open facility *is not allowed to* use core cables. The rationality for this constraint is that in real-life situations the core network and the access network are run independently. The only way to access from the access network to the core network is through an open facility. The problem definitions can be summarized as follows.

Problem 1.9. **Buy-at-bulk Connected Facility Location (BBConFL)**

Input

- Undirected graph $G = (V, E)$
- Metric edge lengths $c_e \in \mathbb{Z}_{\geq 0}$, $e \in E$
- Potential facilities $F \subseteq V$ with opening costs $\mu_i \in \mathbb{Z}_{\geq 0}$, $i \in F$
- Clients $D \subseteq V$ with demands $d_j \in \mathbb{Z}_{>0}$, $j \in D$
- Access cable types K with
  - Capacity $u_k \in \mathbb{Z}_{>0}$, $k \in K$

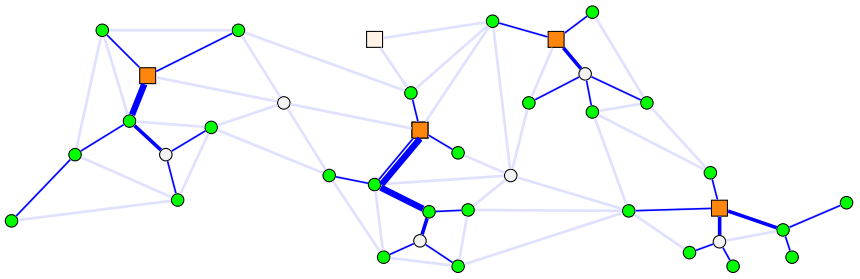


Figure 1.3: A feasible solution for the BBFL problem.

  - Setup cost (per unit length) $\sigma_k \in \mathbb{Z}_{\geq 0}$, $k \in K$

  $\sigma_1 < ... < \sigma_K \leq M$ and $\frac{\sigma_1}{u_1} > ... > \frac{\sigma_K}{u_K}$

- Core cable type with
  - Capacity ∞ and
  - Setup cost (per unit length) $M > \sigma_K$

Solution

- Open facilities $F^* \subseteq F$
- Steiner tree $T^* \subseteq E$ spanning F^*
- Forest $A^* \subseteq E$ such that, for each client $j \in D$, A^* contains exactly one path P_j from j to some open facility $i_j \in F^*$
- Access cable installation $x : A^* \times K \to \mathbb{Z}_{\geq 0}$ of sufficient capacity, i.e., $\sum_{j:\ e \in P_j} d_j \leq \sum_k u_k x_{e,k}$

Goal

$$\min \sum_{i \in F^*} \mu_i + \sum_{e \in T^*} M c_e + \sum_{e \in A^*} \sum_{k \in K} \sigma_k c_e x_{e,k}$$

We will study this problem in Chapter 3 of this thesis where we only focus on obtaining approximation algorithms for this problem.

An interesting variant of BBConFL occurs in logistic networks where the connectivity among facilities is not required. We refer to this variant of BBConFL as the Buy-at-Bulk Facility Location problem (**BBFL**); see Figure 1.3. The reader is referred to Chapter 2 and Chapter 4 of this thesis for exact and approximation algorithms for this problem.

Similar problems also arise in the planning of water and energy supply networks or transportation networks. In some of those applications, however, the consideration of different connection types on the edges of the access network is not motivated by the different capacities but by the different per unit shipping cost of alternative technologies or operational modes, while the maximum capacity is seemingly unlimited. In transportation logistics, for example, the per unit shipping cost on a connection typically is

strongly dependent on the chosen transportation mode, while the maximum capacity is (seemingly) unlimited. This naturally leads to another interesting combined facility location network design problem where each cable type, instead of having a fixed cost and a fixed capacity, has unlimited capacity but a traffic-dependent variable cost in addition to its fixed cost. We call this problem the *Deep-Discount Connected Facility Location* problem (**DDConFL**). More precisely, in the *deep-discount* problem version, an access cable of type k has a fixed *setup cost* (per unit length) $\sigma_k \in \mathbb{Z}_{\geq 0}$ and a flow dependent *incremental cost* (per unit length and per flow unit) of $r_k \in \mathbb{Z}_{\geq 0}$. The problem can be stated as the following.

Problem 1.10. **Deep-Discount Connected Facility Location (DDConFL)**

Input

- Undirected graph $G = (V, E)$
- Metric edge lengths $c_e \in \mathbb{Z}_{\geq 0}$, $e \in E$
- Potential facilities $F \subseteq V$ with opening costs $\mu_i \in \mathbb{Z}_{\geq 0}$, $i \in F$
- Clients $D \subseteq V$ with demands $d_j \in \mathbb{Z}_{>0}$, $j \in D$
- Access cable types K with (infinite capacities and)
  - Setup cost (per unit length) $\sigma_k \in \mathbb{Z}_{\geq 0}$, $k \in K$
  - Flow cost (per unit length and flow unit) $r_k \in \mathbb{Z}_{\geq 0}$, $k \in K$

  $\sigma_1 < ... < \sigma_K \leq M$ and $r_1 > ... > r_K$
- Core cable type with (infinite capacity)
  - Setup cost (per unit length) $M > \sigma_K$

Solution

- Open facilities $F^* \subseteq F$
- Steiner tree $T^* \subseteq E$ spanning F^*
- Forest $A^* \subseteq E$ such that, for each client $j \in D$, A^* contains exactly one path P_j from j to some open facility $i_j \in F^*$
- Access cable type $k_a \in K$ installed for each $a \in A^*$

Goal

$$\min \sum_{i \in F^*} \mu_i + \sum_{e \in T^*} M c_e + \sum_{e \in A^*} c_e(\sigma_{k_e} + r_{k_e} D_e) \quad \text{where } D_e := \sum_{j:\ e \in P_j} d_j$$

We call the unconnected variant of this problem when connectivity among facilities is not required the *Deep-Discount Facility Location* problem (DDFL). We will consider and

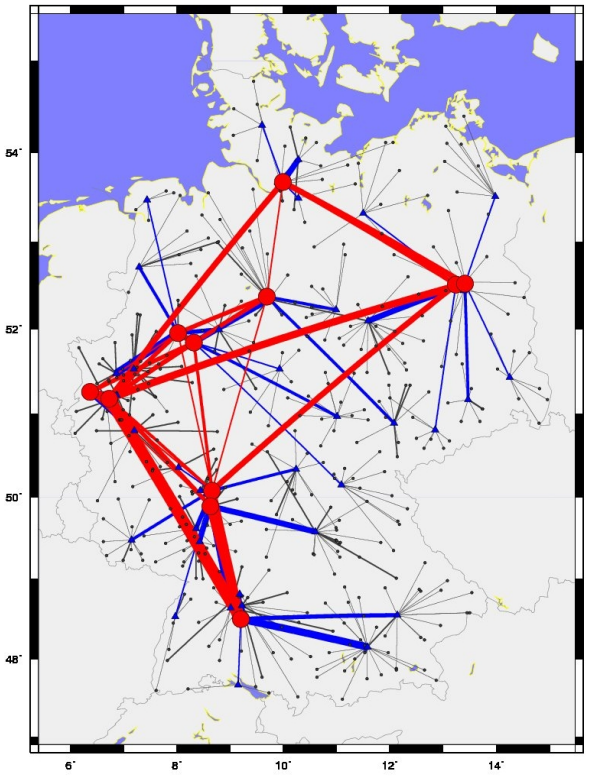


Figure 1.4: A fiber optic network with more focus on the core network and less on the local access networks

analyze these two problems in Chapter 3 (by presenting combinatorial approximation algorithms for DDConFL) and Chapter 4 (by proving an integrality gap upper bound for DDFL).

1.3.2 Facility Location with Complex Connectivity

We recall that the performance measures for the effectiveness of a telecommunication network contain not only cost, but also contain survivability and quality of service. In this section we focus more on the aspects of the core network and less on the access networks. We introduce a generalized version of the rooted connected facility location problem which occurs in planning of telecommunication networks with both survivability and hop-length constraints. Notice that hop-length constraints can also guarantee a required level of quality of service, as long routing paths may lead to unacceptable delays in the network.

A Motivating Example

A typical metropolitan telecommunication network consists of several local access networks, that are connected by a (regional) core network to a central hub node, that provides connectivity to the national or international backbone. The traffic originating at the clients is sent through the access networks to the (regional) core nodes. From there, it traverses the core network(s) to reach the national core or the access network of its destination. Routing functionalities are typically only available at the regional or central core nodes. In fact, the core networks play a primary role for the service availability and the service quality in such networks. This implies that the failure of a core link can represent an unacceptable breakdown of service, as each link in the core network may carry enormous amounts of traffic from a large number of clients. To avoid this, it is common to require that the core network is fault-tolerant and lead to short (hop-limited) routing paths. In the design of optical core networks, for example (see Figure 1.4), this can be addressed by increasing the number of reserved edge-disjoint (hop-limited) routing paths between each pair of core nodes so as to provide a certain level of survivability (and quality of service).

To model such a network planning scenario, we introduce a generalized version of the rooted *connected facility location* problem considering both *survivability* and *hop-length* constraints, called the *Survivable Hop Constrained Connected Facility Location* problem (**SHConFL**).

Definitions

In the SHConFL problem the set of clients, potential facilities, a predetermined root facility, and (optional) interconnection nodes together with their possible connections is modeled as an undirected weighted graph. Given a hop limit (i.e., the maximum allowable path length) $H \geq 1$, and a connectivity requirement (i.e., the minimum required disjointness) $\lambda \geq 1$. The task is to decide which facilities to open, how to assign the clients to the open facilities, and how to interconnect the open facilities in such a way, that the resulting network (core network) contains at least λ edge-disjoint paths, each containing at most H edges, between the root and each open facility; see Figure 1.5. The objective is to minimize the total cost for opening facilities, assigning clients to open facilities, and installing core connections.

The problem definition can be summarized as follows.

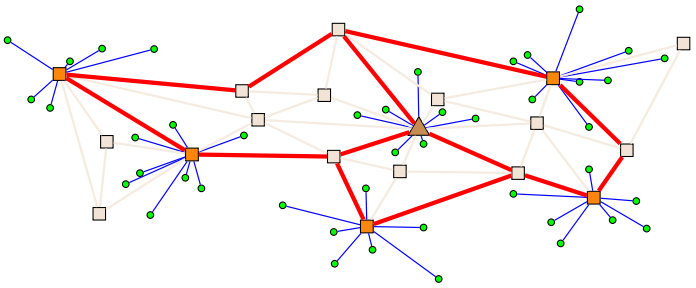


Figure 1.5: A feasible solution for SHConFL with $H = 4$ & $\lambda = 2$, where the square nodes (in orange) represent (open) facilities, the triangle node represents the root facility, and the circle nodes represent clients.

Problem 1.11. **Survivable Hop Constrained Connected Facility Location problem (SHConFL)**

Input

- Undirected graph $G = (D \dot{\cup} S, E)$ containing clients D and core nodes S
- Core edge lengths $c_e \in \mathbb{Z}_{\geq 0}$, $e \in E_S$, where $E_S := \{uv \in E : u, v \in S\}$
- Potential facilities $F \subseteq S$ with opening costs $\mu_i \in \mathbb{Z}_{\geq 0}$, $i \in F$
- Root $r \in S \setminus F$
- Assignment costs $a_{ij} \in \mathbb{Z}_{\geq 0}$ for assigning client j to facility i
- Integer hop limit $H \geq 1$
- Integer connectivity requirement $\lambda \geq 1$

Solution

- Open facilities $I^* \subseteq F$
- Assignment $\sigma^*(j) : D \to I \cup \{r\}$
- Edge set $E^* \subseteq E_S$ containing λ edge-disjoint H-bounded paths between r and each facility $i \in I^*$

Goal

$$\sum_{i \in I^*} \mu_i + \sum_{j \in D} a_{\sigma^*(j)j} + \sum_{e \in E^*} c_e$$

We will study this problem in Chapter 5.

1.4 Contributions and Outline

In this thesis, we introduce and study several new network design facility location problems and develop algorithms for their solution, using a wide range of techniques from computer science and operations research. An overview of the thesis's contributions can be found in the following.

Chapter 2: Facility Location in Buy-at-Bulk Network Design. In this chapter, we study an integrated buy-at-bulk network design facility location problem arising in the design and planning of transportation networks. In contrast to standard buy-at-bulk network design, client demand has to be routed to an open facility; opening a facility incurs an opening cost. In this problem, called Buy-at-Bulk Facility Location, we need to open facilities, build a routing network, and route every client demand to an open facility. Furthermore, capacities of the edges can be purchased in discrete units from a set of different cable types with costs that satisfy economies of scale.

The focus of our work in this chapter is on IP based techniques. We provide a path-based formulation and we compare it with the natural compact formulation for this problem. We then design an exact branch-cut-and-price algorithm for solving the path based formulation. We study the effect of two families of valid inequalities. In addition to this, we present three different types of primal heuristics and employ a hybrid approach to effectively combine these heuristics in order to improve the primal bounds. We finally report the results of our approach that were tested on a set of real world instances and two sets of benchmark instances and evaluate the effects of our valid inequalities and primal heuristics.

Chapter 3: Connected Facility Location in Buy-at-Bulk Network Design. In this chapter, we consider a variant of the buy-at-bulk facility location problem in which open facilities have to be connected on a core network. We introduce and analyze two fundamental versions of this problem. In the Buy-at-Bulk version of the problem, each access cable type has a setup cost and a fixed capacity, whereas in the Deep-Discount problem version, each cable type has unlimited capacity but a traffic-dependent variable cost in addition to its setup cost.

The focus of our work in this chapter is on combinatorial approximation techniques. We derive the first constant-factor approximation algorithms for each variation of the problem, using different algorithmic and analytical techniques.

Chapter 4: LP-based Approximations and Integrality Gaps. The focus of our work in this chapter is on LP based techniques. Namely, we establish a framework for LP-based approximations for buy-at-bulk variants.

For the unconnected variant we present the first LP-based approximation algorithm and prove an upper bound of $O(K)$ on the Integrality gap of the underlying LP. For the connected variant we prove a tighter bound of $O(1)$ on the the Integrality gap of an LP formulation of the problem similar to one for the unconnected version.

Chapter 5: Complex Connected Facility Location. In this chapter, we introduce and study another network design facility location problem where the core network has to fulfill simultaneous survivability and hop-length restrictions between the chosen facilities. In this problem, we need to decide which facilities to open, how to assign clients to the open facilities, and how to interconnect the open facilities on a core network which is fault-tolerant and has short routing paths.

We see that it is NP-hard to compute a constant-factor approximation for the problem. Hence, we focus our research on IP based techniques. We propose two strong extended formulations for the problem and devise a practically efficient branch-and-cut algorithm based on Benders decomposition for finding its solution. We evaluate the algorithm on a set of benchmark instances. The computational results show that most of the algorithm's solutions are within 2 % of optimality.

Chapter 2

Facility Location in Buy-at-Bulk Network Design

The focus of our work in this chapter is on IP based techniques. We consider the Buy-at-Bulk Facility Location problem (**BBFL**). We provide a path-based formulation and we compare it with the natural compact formulation for this problem. We then design an exact branch-cut-and-price algorithm for solving the path based formulation. We study the effect of two families of valid inequalities. In addition to this, we present three different types of primal heuristics and employ a hybrid approach to effectively combine these heuristics in order to improve the primal bounds. We finally report the results of our approach that were tested on a set of real world instances and two sets of benchmark instances and evaluate the effects of our valid inequalities and primal heuristics.

Some results presented in this chapter have been developed in joint work with Ashwin Arulselvan (University of Strathclyde) and Wolfgang A. Welz (TU Berlin). A preliminary version of the results has been published online (Arulselvan et al., 2014).

Previous Work

The BBFL problem was first considered by Meyerson et al. (2000). They showed that BBFL can be seen as a special case of the *Cost-Distance* problem, and thereby provide the first approximation algorithm with approximation guarantee $O(\log(|D|))$ for this problem, where D is the set of clients. Ravi and Sinha (2006) later developed an $O(K)$ approximation, where K is the number of cable types, for this problem and called it *Integrated logistics*. To the best of our knowledge there is still no $O(1)$ approximation nor an exact algorithm for the Buy-at-Bulk Facility Location problem in the literature. We remark that a simplified variation of our BBFL problem, where only a single cable type

(instead of a set of cable types K) is available, has already been considered; see Melkote and Daskin (2001) and Ravi and Sinha (2006) for computational and approximability results for this special case.

The SSBB problem, as another simplification of the BBFL, has been widely studied in *operations research* as well as *computer science* communities. Several approximation algorithms for this problem have been proposed in the computer science literature; see Section 1.2.2 for an overview on the results.

In the operation research literature this problem is also known as *single-source network loading problem* (see Ljubić et al., 2012), or (in the case of telecommunication network planning) as *Local Access Network Design Problem (LAN)* (see Salman et al., 2008). We should recall that the unsplittable variant of the SSBB problem is a special case of our problem.

Randazzo et al. (2001) considered the LAN problem with only two cable types under the assumption that the solution must be a tree (and therefore the flows are unsplittable). They provided a multicommodity flow formulation for the problem and solved it by applying Benders' decomposition. Salman et al. (2008) considered the LAN problem with multiple cable types where the cable types obey economies of scale. They applied a flow-based MIP formulation and worked with the relaxation obtained by approximating the step cost function on the capacities by a lower convex envelope to provide a special branch-and-bound algorithm for LAN design. Raghavan and Stanojević (2006) later reformulated this as a stylized branch-and-bound algorithm. Working with the approximate step cost function, as defined by Salman et al. (2008), Ljubić et al. (2012) considered a stronger multicommodity flow formulation for the problem by disaggregating the commodities, and applied a branch-and-cut algorithm based on Benders decomposition for solving the problem.

Contributions

In this chapter, we undertook the first computational study for the BBFL problem, which so far has been only addressed from the perspective of designing approximation algorithms. We provide the integer programming formulations–both compact and exponential-sized–for the problem. In particular, we model the problem as a path-based formulation and compare it with the natural compact formulation for this problem (see Section 2.1). We study two classes of valid inequalities to improve the lower bounds (see Section 2.2). In addition to this, we present three different types of primal heuristics and employ a hybrid approach to effectively make use of these heuristics in order to improve the primal bounds. We finally develop a branch-cut-and-price algorithm for

solving the path-based IP formulation (see Section 2.3) which allows us to solve very large real-world instances of the problem (see Section 2.4).

2.1 Integer Programming Formulations

Recall that in BBFL, we are given a graph $G = (V, E)$ with a set of facilities F with an opening cost μ_i for each facility i, a set of clients D with demand d_j for each client j, a weight function $c_e \in \mathbb{Z}_{\geq 0}$ for each edge e, and K different access cables. A cable of type i has capacity $u_i \in \mathbb{Z}_{>0}$ and setup cost (per unit length) $\sigma_i \in \mathbb{Z}_{\geq 0}$. We assume that the cable types obey economies of scale, i.e., we have $\sigma_1 < ... < \sigma_K$ and $\frac{\sigma_1}{u_1} > ... > \frac{\sigma_K}{u_K}$. The task is to decide which facilities to open and where to install which access cable type to obtain an access network whose capacities suffice to simultaneously route all client demands to open facilities. Recall that the entire demand of each client must be routed via the access network to an open facility using exactly one path. The objective is to minimize the total cost of opening facilities and installing access cables.

In the following we propose a natural compact IP formulation for the BBFL problem. Relying on (approximate) step cost functions that one can precompute for each edge using dynamic programming, we then propose some new IP formulations for the problem.

A natural IP Formulation

We write a natural flow-based integer linear program for the BBFL problem. For each edge we create a pair of anti-parallel directed arcs, with same length as the original one. Let $\vec{E}$ be the set of these arcs. We denote to the undirected version of an arc $\bar{e} \in \vec{E}$ by e. For an edge $e = lm$ between nodes l and m, we will use a notation (l, m) or (m, l) to explicitly specify the orientation of the corresponding arc $\bar{e} \in \vec{E}$. We will use the notation e and $\bar{e}$, when it is clear from the context, for the sake of compactness.

In our model we will use the following variables: For every $\bar{e} \in \vec{E}$ and client $j \in D$, the variable $f_{\bar{e}}^j$ indicates if flow from client j uses arc $\bar{e}$; for $e \in E$ and access cable type k, x_e^k indicates if access cable k is installed on edge e; and z_i indicates if facility i is open or not. We use the notation $\delta^+(u) = \{(u, v) \in \vec{E}\}$ and $\delta^-(v) = \{(u, v) \in \vec{E}\}$. We are now ready to provide our integer program.

$$\text{(IP-2-1)} \quad \min \sum_{i \in F} \mu_i z_i + \sum_{e \in E} c_e \sum_{k=1}^{K} \sigma_k x_e^k$$

$$\sum_{\bar{e} \in \delta^+(j)} f_{\bar{e}}^j \geq 1 \qquad \forall j \in D \tag{2.1}$$

$$\sum_{\bar{e} \in \delta^+(v)} f_{\bar{e}}^j = \sum_{\bar{e} \in \delta^-(v)} f_{\bar{e}}^j \qquad \forall j \in D, v \in V \backslash F, v \neq j \tag{2.2}$$

$$\sum_{\bar{e} \in \delta^-(i)} f_{\bar{e}}^j - \sum_{\bar{e} \in \delta^+(i)} f_{\bar{e}}^j \leq z_i \qquad \forall j \in D, i \in F \tag{2.3}$$

$$\sum_{j \in D} d_j (f_{(l,m)}^j + f_{(m,l)}^j) \leq \sum_{k=1}^{K} u_k x_{lm}^k \qquad \forall\ lm \in E \tag{2.4}$$

$$f_{\bar{e}}^j, z_i \in \{0, 1\}$$

$$x_e^k \quad \text{non-negative integers}$$

Constraints (2.1) impose that at least one unit of flow leaves each client. Constraints (2.2) are flow conservation constraints at non-facility nodes. Constraints (2.3) state that the flow only terminates at open facilities. Constraints (2.4) ensure that we install sufficient capacity to support the flow.

It is not hard to see that an optimal fractional solution for the LP relaxation of IP-2-1 only uses the last cable type with the lowest cost per capacity ratio. Consider, for example, the following small instance. Given are a single client with unit demand; a single facility with zero cost; an edge between them with unit cost; and two types of access cables: $\sigma_1 = 1$, $u_1 = 1$, $\sigma_2 = 2$, $u_2 = l$. While the cost of the optimal solution is 1, the cost of the optimal fractional solution is $\frac{2}{l}$. Hence, the integrality gap is proportional to l and so it can be made arbitrarily large.

Remark 2.1. The integrality gap of (IP-2-1) can be arbitrarily large.

An alternative IP modeling of BBFL

We now provide a slightly better formulation. Notice that if the locations of the open facilities as well as the subgraph (edges supporting the access network) connecting clients to open facilities are given, then the problem reduces to the *integer minimum knapsack* problem for each edge; wherein one needs to choose the optimal combination of the cables for that edge so as to support the demand flowing through it. Inspired by the results in Salman et al. (2008) for the single-sink buy-at-bulk network design problem, we compute the optimal combination of cable types for all possible flow levels on every edge (using dynamic programming). This provides a monotonically increasing step cost

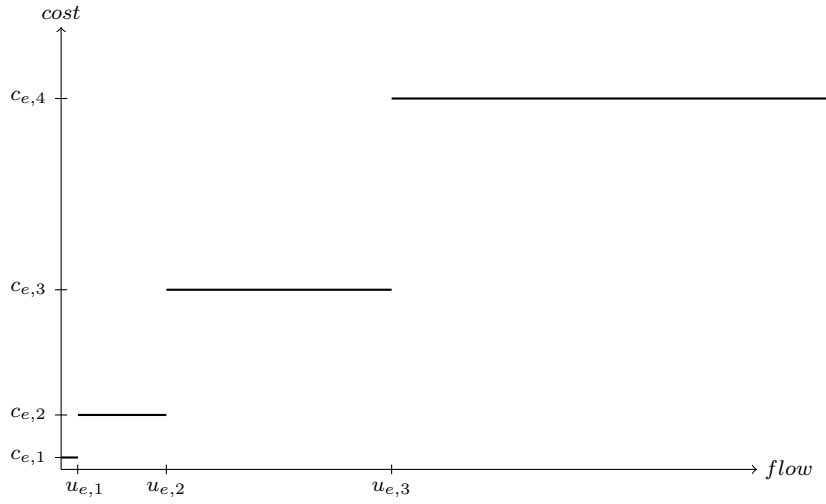


Figure 2.1: An illustration for the cost function g_e

function for each edge e in the network, which we denote by g_e. It is worth noting that this property drives our primal heuristic to provide solutions with reasonable quality (see Section 2.3.1). As a result, this idea allows the following alternative IP formulations for BBFL.

Flow based Formulation

We first present an IP model with a nonlinear objective function as follows.

$$\min \sum_{i \in F} \mu_i z_i + \sum_{e \in E} g_e(\bar{f}_e)$$

$$\sum_{\bar{e} \in \delta^+(j)} f^j_{\bar{e}} \geq 1 \qquad \forall j \in D \tag{2.1}$$

$$\sum_{\bar{e} \in \delta^+(v)} f^j_{\bar{e}} = \sum_{\bar{e} \in \delta^-(v)} f^j_{\bar{e}} \qquad \forall j \in D, v \in V \backslash F, v \neq j \tag{2.2}$$

$$\sum_{\bar{e} \in \delta^-(i)} f^j_{\bar{e}} - \sum_{\bar{e} \in \delta^+(i)} f^j_{\bar{e}} \leq z_i \qquad \forall j \in D, i \in F \tag{2.3}$$

$$\sum_{j \in D} d_j (f^j_{(l,m)} + f^j_{(m,l)}) \leq \bar{f}_{lm} \qquad \forall lm \in E \tag{2.5}$$

$$f^j_{\bar{e}}, z_i \in \{0,1\}$$

$$\bar{f}_e \quad \text{non-negative integers}$$

In this formulation we use variable $\bar{f}_e$ associated with each edge e to indicate the total flow crossing that edge.

Next, we obtain a corresponding linear IP modeling of the above model. We consider each piece of the step cost function, for each edge, as a module with a specified cost and a specified capacity available for that edge; see Figure 2.1. More precisely, we assume that for each edge e a set $\mathcal{N}_e = \{n_1, n_2, \ldots, n_{\mathcal{N}_e}\}$ of modules (obtained by finding the optimal combination of cable types for all flow levels on e) is given, and at most one of these modules can be installed to support the corresponding flow along that edge. Each module n has a cost of $c_{e,n}$ and a capacity of $u_{e,n}$. Finally, to model this, we introduce, for each edge e and each module $n \in \mathcal{N}_e$, a variable $x_{e,n}$ which indicates whether module n has been installed on edge e or not. Intuitively speaking, this indicates whether piece n of the step cost function determines the optimal cable cost for edge e.

Now, we reformulate the problem by replacing constraints (2.4) by constraints (2.6) and (2.7) as follows.

$$
\begin{aligned}
\text{(IP-2-2)} \quad \min & \sum_{i \in F} \mu_i z_i + \sum_{e \in E} \sum_{n \in \mathcal{N}_e} c_{e,n} x_{e,n} & & \\
& \sum_{\bar{e} \in \delta^+(j)} f^j_{\bar{e}} \geq 1 & \forall j \in D & \quad (2.1)\\
& \sum_{\bar{e} \in \delta^+(v)} f^j_{\bar{e}} = \sum_{\bar{e} \in \delta^-(v)} f^j_{\bar{e}} & \forall j \in D, v \in V \backslash F, v \neq j & \quad (2.2)\\
\sum_{\bar{e} \in \delta^-(i)} f^j_{\bar{e}} - & \sum_{\bar{e} \in \delta^+(i)} f^j_{\bar{e}} \leq z_i & \forall j \in D, i \in F & \quad (2.3)\\
\sum_{j \in D} d_j (f^j_{(l,m)} + f^j_{(m,l)}) \leq & \sum_{n \in \mathcal{N}_{lm}} u_{lm,n} x_{lm,n} & \forall lm \in E & \quad (2.6)\\
& \sum_{n \in \mathcal{N}_{lm}} x_{lm,n} \leq 1 & \forall lm \in E & \quad (2.7)\\
& x_{e,n}, f^j_{\bar{e}}, z_i \in \{0,1\} &&
\end{aligned}
$$

We denote by $\text{proj}_{f,z}(P) = \{f, z \in [0,1]^{|\vec{E}| \times |D| \times |F|} \mid \exists (x, f, z) \in P\}$ the projection of polyhedra P on the space of f and z variables. Let P_1 and P_2 denote the feasible space of the LP relaxation corresponding to formulations IP-2-1 and IP-2-2, respectively. We give an intuitive explanation for the fact that $\text{proj}_{f,z}(P_2) \subseteq \text{proj}_{f,z}(P_1)$. This could be shown by the claim that for every solution in P_2, we could construct an equivalent solution $(\bar{\mathbf{f}}, \bar{\mathbf{z}}, \bar{\mathbf{x}})$ in P_1. Assume we have a solution (possibly fractional) $(\mathbf{f}^*, \mathbf{z}^*, \mathbf{x}^*) \in P_2$, we take the vectors $\bar{\mathbf{f}} = \mathbf{f}^*$ and $\bar{\mathbf{z}} = \mathbf{z}^*$ the same for the newly constructed solution. But for the cable variables we do as follows

$$
\bar{x}^k_{ij} = \sum_{n \in \mathcal{N}_{ij}} \mathbb{L}_{n(k)} x^*_{ij,n},
$$

where $\mathbb{L}_{n(k)}$ is the number of cables of type k used by module n and this is obtained by our dynamic program. Note that the capacity constraints were the only set of different constraints in the two formulations. Therefore, based on the way the modules were built, the newly constructed solution is obviously feasible to P_1. This implies the following result.

Lemma 2.2. *IP-2-2 is at least as strong as IP-2-1 in terms of the lower bounds provided by their linear relaxations.*

The IP-2-2 formulation has $O(|D|\cdot|E|)$ variables and $O(|E|)$ constraints which may lead to quite large IP formulations with respect to the size of real-world applications; see Section 2.4. To get around solving such a model with huge number of variables, we will propose a path-based formulation for the problem and solve it using column generation.

Path based Formulation

We present a path-based formulation for the problem with an exponential number of variables. For the sake of modeling paths, we first create a dummy root node r and connect all facilities with the root node. Let $\vec{E}' = \vec{E} \cup \{\bigcup_{i\in F}(i,r)\}$. Let $P(j)$ denote the set of all possible paths in $G' = (V \cup \{r\}, \vec{E}')$ starting from client j and terminating at the root node r. Remember that the demand of each client must be routed to an open facility, and so to the root node, via a single path. For each $j \in D$ and for each $p \in P(j)$, we introduce a binary variable y_p which indicates if flow from j is routed along p. Then the problem can be formulated as follows:

$$\text{(IP-2-3)} \quad \min \sum_{i\in F} \mu_i z_i + \sum_{e\in E}\sum_{n\in\mathcal{N}_e} c_{e,n}\cdot x_{e,n}$$

$$\sum_{p\in P(j)} y_p = 1, \qquad \forall j \in D \tag{2.8}$$

$$\sum_{j\in D} \sum_{\substack{p\in P(j):\\ \{(l,m),(m,l)\}\cap p\neq\emptyset}} d_j y_p \leq \sum_{n\in\mathcal{N}_{lm}} u_{lm,n} x_{lm,n}, \qquad \forall lm \in E \tag{2.9}$$

$$\sum_{n\in\mathcal{N}_{lm}} x_{lm,n} \leq 1, \qquad \forall lm \in E \tag{2.10}$$

$$\sum_{p\in P(j):(i,r)\in p} y_p \leq z_i, \qquad \forall i \in F, \forall j \in D \tag{2.11}$$

$$y_p, x_{e,n}, z_i \in \{0,1\}$$

Constraints (2.8) force each client to be connected to a routing path. Constraints (2.9) ensure that we install sufficient capacity to support the flow along routing paths, Constraints (2.10) guarantee that at most one module is installed along each edge and Constraints (2.11) ensure that a serving facility is open.

It is worth noting that one could improve the lower bound provided by the linear relaxation of IP-2-3 by adding the following set of strengthening inequalities to the model:

$$\sum_{\substack{p\in P(j):\\ \{(l,m),(m,l)\}\cap p\neq\emptyset}} y_p \leq \sum_{n\in\mathcal{N}_{lm}} x_{lm,n}, \qquad \forall lm \in E, j \in D \tag{2.12}$$

This guarantees that there always is some module installed along any edge used by routing paths. However, adding the above set of valid inequalities leads to a model with $O(|D| \cdot |E|)$ constraints, increasing the number of constraints by a factor of $|D|$. Therefore, instead of adding them directly to the model, we will propose a family of strong valid inequalities (see Section 2.2) for IP-2-3 which contains Inequalities (2.12) as special cases.

2.2 Valid Inequalities

In this section we propose two families of valid inequalities that naturally emerge from the IP-2-3 model.

Cover Inequalities

In order to derive the cover inequalities corresponding to each constraint in set (2.9), we obtain a knapsack structure by complementing the x variables (replacing x by $1 - x$) in the constraint. Consider the constraint in set (2.9) corresponding to edge $lm \in E$. Let $U_{lm} = \sum_{n\in\mathcal{N}_{lm}} u_{lm,n}$. We define $\theta_{lm} = (D_\theta, M_\theta)$ to be a cover with respect to edge lm, where $D_\theta \subseteq D$ and $M_\theta \subseteq \mathcal{N}_{lm}$, if

$$\sum_{j\in D_\theta} d_j + \sum_{n\in M_\theta} u_{lm,n} > U_{lm}\,.$$

We say that a cover is minimal when removing any item either from D_θ or M_θ results in a cover for which the above inequality does not hold. It is not hard to show that if

θ_{lm} is a minimal cover, then the following inequalities are valid:

$$\sum_{j\in D_\theta} \sum_{\substack{p\in P(j):\\ \{(l,m),(m,l)\}\cap p\neq\emptyset}} y_p + \sum_{n\in M_\theta}(1-x_{lm,n}) \leq |M_\theta| + |D_\theta| - 1 \iff$$

$$\sum_{j\in D_\theta} \sum_{\substack{p\in P(j):\\ \{(l,m),(m,l)\}\cap p\neq\emptyset}} y_p \leq \sum_{n\in M_\theta} x_{lm,n} + |D_\theta| - 1 \tag{2.13}$$

Note that Inequalities (2.12) are dominated by Inequalities (2.13) with D_θ containing only a single client j.

Let (x^*, y^*, z^*) be the optimal fractional solution of the LP relaxation of model IP-2-3. Now, we present how to find a cover inequality corresponding to edge lm violated by (x^*, y^*, z^*). For each $j \in D$, we let

$$w_j^* = \sum_{\substack{p\in P(j):\\ \{(l,m),(m,l)\}\cap p\neq\emptyset}} y_p^*$$

And let $F_{lm} \subseteq \mathcal{N}_{lm}$ be the set of modules for the edge lm such that $x^*_{lm,n} > 0$. A most violated cover inequality is obtained by solving the following knapsack problem:

$$\begin{aligned}
\min\ \gamma = \sum_{n\in F_{lm}} x^*_{lm,n}x_{lm,n} + \sum_{j\in D}(1-w_j^*)w_j \\
\sum_{j\in D} d_j w_j + \sum_{n\in F_{lm}} u_{lm,n}x_{lm,n} \geq \sum_{n\in F_{lm}} u_{lm,n} + 1 \\
x_{lm,n} \in \{0,1\}, \quad \forall n \in F_{lm} \\
w_j \in \{0,1\}, \quad \forall j \in D
\end{aligned}$$

Which is equivalent to the following standard knapsack problem in maximization form by replacing x by $1-\bar{x}$, and w by $1-\bar{w}$.

$$\begin{aligned}
\sum_{n\in F_{lm}} x^*_{lm,n} + \sum_{j\in D}(1-w_j^*) - \max \sum_{n\in F_{lm}} x^*_{lm,n}\bar{x}_{lm,n} + \sum_{j\in D}(1-w_j^*)\bar{w}_j \\
\sum_{j\in D} d_j \bar{w}_j + \sum_{n\in F_{lm}} u_{lm,n}\bar{x}_{lm,n} \leq \sum_{j\in D} d_j - 1 \\
\bar{x}_{lm,n} \in \{0,1\}, \quad \forall n \in F_{lm} \\
\bar{w}_j \in \{0,1\}, \quad \forall j \in D
\end{aligned}$$

Let $D' \subseteq D$ and $F'_{lm} \subseteq F_{lm}$ be the optimal subsets of the minimizing knapsack problem. Then, it is easy to show that (x^*, y^*, z^*) violates Inequality (2.14) if $\gamma < 1$.

$$\sum_{j \in D'} \sum_{p \in P(j):(l,m) \in p} y_p \leq \sum_{n \in F'_{lm} \cup \{N_{lm} \setminus F_{lm}\}} x_{lm,n} + |D'| - 1 \tag{2.14}$$

Cut Inequalities

The modules generated follow economies of scale and hence the optimal solution of the LP relaxation ends up fractionally picking the last module that has the lowest cost per capacity ratio. In this section, we will introduce a set of valid inequalities, called *cut inequalities*, to somewhat remedy this problem. Given a fractional optimal solution (x^*, y^*, z^*) of IP-2-3. For the graph $\bar{G} = (V \cup \{r\}, E \cup \{\bigcup_{i \in F} ir\})$, we take the edge capacities to be $u_{lm} = \sum_{n \in \mathcal{N}_{lm}} x^*_{lm,n}$, for all $lm \in E$, and z^*_i for all ir edges. For every client $j \in D$, we solve the maximum flow problem with source j and sink r. If the flow value is less than one, then we obtain the following violated cut:

$$\sum_{lm:k \in \bar{S}} \sum_{n \in \mathcal{N}^j_{lm}} x_{lm,n} + \sum_{i \in \bar{S}} z_i \geq 1 \tag{2.15}$$

where $\bar{S}$ (containing j; not r) indicates the corresponding minimum cut set, and $N^j_{lm} = \{n \in \mathcal{N}_{lm} : u_{lm,n} \geq d_j\}$ indicates the modules available for edge lm with capacities greater than the demand of the client j. The validity of the cut follows from the fact that every client j needs to be connected to some facility along a path with every edge in the path having at least one module, with capacity greater than the demand d_j, installed.

2.3 Solution Procedure

Since the path based formulation presented above contains an exponential number of variables, our solution procedure is based on the column generation technique. We consider as the restricted master problem the continuous relaxation of the IP-2-3 model including all the constraints and the x and z variables, but only the y variables corresponding to a subset $P'(j) \subseteq P(j)$ of paths for each $j \in D$.

2.3.1 Initialization

We enrich the restricted master problem with solutions obtained from a few runs of a randomized greedy algorithm. Our algorithm (`GreedyAlgorithm`) works as follows: We

Algorithm `GreedyAlgorithm`

1. Pick a random permutation of clients in D;
   Let $\Pi = (j_1, j_2, ..., j_{|D|})$ be the picked permutation.
2. **For** $t = 1, 2, \cdots, |D|$ **do**
   - Find the cheapest cost routing path p_t, using the partial solution constructed by the previous $t-1$ clients.
   - Let i_t be the facility for which $(i_t, r) \in p_t$. Open facility i_t, if it has not been opened by the previous $t-1$ clients.
   - Route d_{j_t} units of demand from j_t to facility i_t via path p_t.

pick a random permutation $\Pi = (j_1, j_2, ..., j_{|D|})$ of clients. Then we construct a network in a greedy fashion, examining clients one by one in that order and greedily route the demand of each client to some open facility, while the cost of routing of each client depends on the routing of the previous clients.

More specifically, we start with an empty network, i.e. no modules installed and no facilities opened. We go down the list of clients according to Π, iteratively pick a client j_t that has not been routed yet and route its demand to some facility via a routing path with the minimum total cost, using the network constructed by the previous $t-1$ clients. This can be done as follows. Recall step cost function g_e obtained by finding the optimal combination of cable types for all flow levels on e. Let $\bar{f}_{\bar{e}}$ be the amount of flow which has already been routed along arc $\bar{e}$ by the previous $t-1$ clients and let $\bar{I}$ be the set of facilities opened so far. Consider graph $G' = (V \cup \{r\}, \vec{E}')$. For each arc $\bar{e} \in \vec{E}$ of this graph, we set its weight to be $g_e(\bar{f}_{\bar{e}} + d_{j_t}) - g_e(\bar{f}_{\bar{e}})$ (which is at most $g_e(d_{j_t})$ due to the fact that the cost function g_e is monotonically increasing); or, in other words, to be the marginal increase in the cost of that edge for transporting additional d_{j_t} units of demand. The weights for arcs $(i, r) \in \vec{E}$ are also defined to be μ_i if $i \notin \bar{I}$; otherwise 0. Now, to route the demand from client j_t, we find the shortest path from j_t to r in this graph. Let p_t be the resulting path and let i_t be the facility for which $(i_t, r) \in p_t$. Then we open facility i_t, if $i_t \notin \bar{I}$, and route the entire demand of client j_t to facility i_t via path p_t. Finally, we set $\bar{I} = \bar{I} \cup \{i_t\}$ and update $\bar{f}_{\bar{e}}$ accordingly for all $\bar{e}$. We continue this process for all clients (one bye one in the picked order) until all the demands are routed to open facilities. A brief description of the algorithm is given in the figure.

It should be noticed that such shortest-path based approaches can also be theoretically interesting; e.g. see the paper by Charikar and Karagiozova (2005) where an $O(log^2(|D|))$ approximation algorithm for the non-uniform buy-at-bulk network design problem is devised.

2.3.2 Column Generation

We iteratively solve the restricted master problem and search for new columns having negative reduced cost that is computed using the optimal dual solution. Let the dual variables corresponding to Constraints (2.8) be ρ_j, for all $j \in D$. We will refer to the dual variables corresponding to Constraints (2.9) with the notation as π_{lm}, for all $lm \in E$ and the dual variables corresponding to Constraints (2.11) by γ_i^j, for all $i \in F, j \in D$. For each $j \in D$, we determine if a path p in $P(j) \setminus P'(j)$ could improve the current (fractional) solution. The pricing problem associated with client j is:

$$\min_{p \in P(j)} \quad - \left(\rho_j + \sum_{\substack{p \in P(j): \\ \{(l,m),(m,l)\} \cap p \neq \emptyset, \\ l \neq r}} d_j \pi_{lm} + \sum_{i \in F} \mathbb{I}_i^p \gamma_i^j \right)$$

where $\mathbb{I}_i^p$ is an indicator variable denoting whether edge ir is in the path p or not (r being the root node).

Consider a graph $\bar{G} = (V \cup \{r\}, E \cup \{\bigcup_{i \in F} ir\})$, we take the weight of each edge lm to be $-d_j \pi_{lm}$, for all $lm \in E$ and weights $-\gamma_i^j$, for all ir edges, $i \in F$. We now find the shortest path in $\bar{G}$ from j to the root node r. Note that the dual vectors $\boldsymbol{\pi}, \boldsymbol{\gamma} \leq 0$ and so Dijkstra's algorithm can be used to find the shortest path. If the solution to this shortest path problem has length less than ρ_j, then the solution is not optimal for the master problem and this path should be added into our restricted master problem. The new restricted master problem is re-solved and the process is iterated as long as the pricing problems corresponding to the clients generate new columns.

We notice that the feasible solutions space of the restricted master problem may be empty during the loop mentioned above, due to branching constraints (see Section 2.3.4) or in the beginning when no columns have been generated yet. In this case, we use Farkas' Lemma to add columns that gradually move the solutions space closer to the feasible region. Note that this is the same problem as the pricing problem above considering the so called dual Farkas values. This method has been called *Farkas pricing*, and provided in Achterberg (2009) within the SCIP framework (see Section 2.4).

2.3.3 Cut Generation

Once the column generation is over, we start searching for the cover inequalities violated by the current fractional solution. We search for such cuts as described in Section 2.2. The generated cover cuts (2.14) will not change the structure of the pricing problem,

however the weights associated with edges of the network may be changed. Hence the column generation process should be repeated considering the new pricing problems, once new cuts are added.

Each e has multiple cover inequalities associated with it. Let Θ_e be the set of covers associated with edge e. Let $\Theta = \bigcup_{e \in E} \Theta_{(e)}$. For a cover $\theta \in \Theta$, let D_θ be the set of clients involved in the cover and α_θ be the corresponding dual variable.

The new pricing problem associated with client j is:

$$\min_{p \in P(j)} \quad - \left(\rho_j + \sum_{\substack{p \in P(j): \\ \{(l,m),(m,l)\} \cap p \neq \emptyset, \\ l \neq r}} d_j \pi_{lm} + \sum_{i: i \in F} \mathbb{I}_i^p \gamma_i^j + \sum_{(l,m) \in p} \sum_{\substack{\theta \in \Theta_{lm}: \\ j \in D_\theta}} \alpha_\theta \right) \tag{2.16}$$

Note that cut inequalities (2.15) involving x and z variables improve the quality of the bound without affecting the pricing problem.

Such a loop is repeated until neither new columns nor cuts are added.

2.3.4 Branching Strategies

So far, we have described how we employ the column generation and cut separation methods for solving the master problem. However, the optimal solution to the master problem might not be integral at the end of the price-and-cut loop. Integer linear programs are typically solved by using *Branch-and-Bound*, a widely known technique, which uses branching to handle integrality. This technique, when used together with column generation and cut separation is called Branch-Cut-and-Price.

The most intuitive branching rule for variable y_p with a fractional value is to create two branches, the first one with $y_p = 0$ and other with $y_p = 1$. However, this cannot be done without introducing complications in the pricing subproblem of our column generation algorithm. If a path p is fixed to zero, the pricing subproblem will find this route again on the next iteration and will return it to the restricted master. An alternative can be to implement standard branching in the space of the compact formulation by adding a constraint that give us lower and upper bounds on the capacity of an edge that currently has a fractional flow value. However, this branching decision destroys the pricing subproblem structure. To get around these issues, similar to the one used by Barnhart et al. (2000) for multi-commodity flow, we impose implicit branching constraints as the following. Consider any client $j \in D$ whose demand is routed along more than one path, say two distinct paths p_1 and p_2, in the current (fractional) solution to the master

problem. Note that paths have at least node j in common. Consider the first node at which these two paths split. We partition the set of edges emanating from this node into two subsets E_1 and E_2 such that E_1 (E_2, respectively) intersects p_1 (p_2, respectively). Then, we create two branches with one imposing $\sum_{p \in P(j): p \cap E_2 \neq \emptyset} y_p = 0$ and the other imposing $\sum_{p \in P(j): p \cap E_1 \neq \emptyset} y_p = 0$.

More precisely, when we are given a fractional solution, we create the next branch as follows:

1. Out of those clients whose demand is split, choose the $j \in D$ with the highest demand.
2. Identify the two paths p_1 and p_2 with the most fractional y_{p_1} and y_{p_2} of client j. (If the fractionalities are identical, we use the objective value of the corresponding variables as a tiebreaker.)
3. By traversing the path starting from the client node, identify the last common node d in both paths and then create two subsets from the outgoing edges. In order to generate balanced branches, the sets E_1 and E_2 are selected in such a way that $|E_1|$ and $|E_2|$ differ by at most one. For an efficient implementation it is further important to only add arcs to the set, that are not already forbidden in the current node of the branch-and-bound tree.

We remark that these branching decisions requires no changes in the basic structure of the pricing problem, which remains a simple shortest path problem through all the enumeration process. Our subproblems will work with the subgraph with the corresponding forbidden edges deleted.

2.3.5 Primal Heuristic

For the overall performance of a Branch-and-Price approach it is crucial that good primal solutions of the problem are found, however it is usually unlikely for the branch-and-bound process alone to find good upper bounds fast. We therefore present two simple heuristic algorithms in the following.

LP based Greedy Heuristic

The following heuristic is invoked at each node of the (B&B) tree. Given the fractional optimal solution of the current node, resulting in fractional $x_{e,n}$ and z_i variables, run algorithm `GreedyAlgorithm`, described in Section 2.3.1, while the weight assignment to

the arcs of the graph G' is based on the LP solution: For i^{th} client according to the picked permutation, assign a weight $\left(g_e(\bar{f}_{\bar{e}} + d_{j_i}) - g_e(\bar{f}_{\bar{e}})\right) \cdot \left(1 - \sum_{n \in \mathcal{N}_e} x_{e,n}\right)$ to each arc $\bar{e} \in \vec{E}$; set weights for (i, r) arcs to be $\mu_i \cdot (1 - z_i)$ if $i \notin \bar{I}$, otherwise 0; then route its demand to r via a routing path with the minimum total cost in G'.

IP based Primal Heuristic

We treat the problem including all variables generated so far as a complete integer program and then solve that program using IBM ILOG CPLEX. The result gives us the best possible solution that can be achieved by only using the current paths from $P'(j)$ without adding any new variables. Since the number of variables is fixed (to those variables generated so far in our framework, i.e. $P'(j)$), this problem is easier to solve and even if the solution process of the resulting IP is canceled after a certain amount of time, the best solution found is still a feasible solution to our problem. But as the IPs considered are still rather big, the solution process is very time consuming. In order to implement this idea as efficient as possible, the CPLEX-problem is solved in the background so that the main CPU-thread can still continue to perform the regular branch-and-bound process using SCIP. This way only some minor coordination and communication overhead needs to be performed within the actual SCIP-plugin, while the expensive solution process can be performed in other threads.

If CPLEX finds a new improving solution the main thread is signaled and the solution is then copied into SCIP so that it can be immediately used as an upper bound in the branch-and-bound process. After either CPLEX reaches a certain node limit or after too many new variables have been generated since the last start of the heuristic, the new variables are added and the solver is restarted. By adding variables to the existing CPLEX-problem we assure that solutions found in previous runs are still available and automatically used as a primal bound in the new computation. To improve the performance even further we also add the valid inequalities (see Section 2.2) as CPLEX user-cuts. Those inequalities are problem dependent and thus improve the automatically generated cuts by CPLEX. As the branch-and-price algorithm is single threaded, this approach allows us to perform this IP based heuristic on modern multi-core processors with basically no additional computation time.

2.4 Computational Results

The Branch-Cut-and-Price approach has been implemented using the framework provided by SCIP Achterberg (2009). In this context SCIP handles all the underlying

Integer Programming specific aspects and has been extended with problem dependent plugins for the pricing and branching as well as the cut generation and primal heuristic. As explained in Section 2.3.2 the pricing problem corresponds to solving a shortest path problem for every client, which is solved via an implementation of Dijkstra's Algorithm. To avoid that this computation is performed for all the clients in every iteration, we implemented a two step approach: In the first step one single-source shortest paths problem is solved to find lower bounds for the shortest paths to every clients. If we take a look at the arc costs corresponding to the problem for client j, we notice that only the dual variables corresponding to the Constraints (2.11) and to the cover cuts depend on j. The ρ_j and the capacities d_j can be applied after the shortest paths have been calculated. However, different dual variables resulting from the Constraints (2.11) are selected for different clients. Here the smallest of the corresponding values is used as an arc weight. This leads to the following optimization problem, where the shortest r-j-paths in the resulting graph represent a lower bound to the shortest paths in the actual pricing problem:

$$-\rho_j + d_j \left(\min_{p \in P(j)} \left(\sum_{\substack{(l,m) \in p \\ l \neq r}} -\pi_{lm} + \sum_{i : i \in F} \mathbb{I}_i^p \cdot \min_{j \in D} \frac{-\gamma_i^j}{d_j} \right) \right)$$

As the dual variables α_c are all not positive, this gives us a lower bound of the pricing problem (2.16).
In the second step the graph with client dependent weights is then only solved for the clients j that had a non-negative path in the first step.

For the cover inequalities we use the solver provided as part of SCIP that uses dynamic programming to find an optimal Knapsack solution. The min-cut computations for the cut inequalities are performed using a push-relabel maximum flow algorithm.

2.4.1 Preprocessing

Some basic preprocessing techniques have been used to reduce the size of the network. As a first step, there are some very basic rules that can be used to identify edges that will never carry any flow. This is for example the case for edges that cannot be reached by any facility or for edges adjacent to a node with a degree of one that is not a facility or a client.

This idea can also be extended for clients with only one adjacent edge. In this case we know that in any optimal solution this one edge will carry a flow of exactly the demand d of that client. It is therefore possible to relocate the client to the other end of the edge by

adding a certain offset to the objective function which corresponds to the cost required to route d along the edge. This approach is especially useful if the clients are arranged in a star like fashion around one node which is a common scenario for certain network types. Since our problem is uncapacitated and all the modules follow economies of scale, it is now possible to aggregate the clients located in the same node into one single client corresponding to their total demand.

As our application represents real-world maps and networks, the graph G is usually sparse. And since there might also be certain bottlenecks in the network, the graph G often contains bridges. These bridges can then be used to identify upper and lower bounds of the demand routed across certain edges: As a first step we detect bridges using Tarjan's Bridge-finding algorithm Tarjan (1974). Deleting a bridge separates the graph into two components; we now let d_{bridge} denote the total demand of all clients in the component corresponding to one side of the bridge. If this side does not contain any facilities, we know that a value of exactly d_{bridge} has to be routed across the bridge and that further a value of at most d_{bridge} will flow through any edge on that side. This information can then be used to eliminate unnecessary modules on these edges. If an upper bound of zero on an edge is detected the edge can even be entirely eliminated.

All the preprocessing rules mentioned above are used in our implementation. For sparse graphs they help to considerably reduce the number of edges and hence the number of x variables. This helps in significantly speeding up the pricing process and strengthening the LP bound as well.

Instances Details

We used three different tests sets, namely RW, PA, and JMP instances.

- The *RW* (real-world) instances tested correspond to real world network planning problems. The networks were generated from the publicly available information obtained through geographic information systems[1], arising from the German research project FTTx-Plan (FTT). Each instance corresponds to a region in Germany and was constructed bearing in mind the potential client and facility locations. The street segments form the edges, while the street intersections and traffic circles provide the intermediate nodes . The information about the different cable types along with their costs and capacities were provided by our industry partners.

- The *JMP* instances were created as by Johnson et al. (2000). We modified the adaptation of this model carried out by Álvarez-Miranda et al. (2014) for a single

[1] See http://www.zib.de/projects/tools-planning-fttx-networks

commodity robust network design problem. 20% of the terminals were taken as facilities and the remaining were taken as clients. The largest demand is taken as an argument and demands were randomized between 0 and this maximum demand value. The cable types were taken to be the same as in RW instances.

- The *PA* instances were created based on the model designed by Barabási and Albert (1999) in order to create realistic networks. Cacchiani et al. (2014) adapted this model to solve a single commodity network design problem. We used this model to generate our instances. We give the maximum demand as an input and randomize our demands for all clients. We also take the cost of the facilities based on our real world instances (it is taken to be a constant for all facilities).

Computational Experiments

All computations were performed on Intel Xeon E5-2630, 2.3 GHz CPUs using one thread for SCIP and three threads for the CPLEX- Heuristic. We used CPLEX 12.6.0 (IBM, 2013) and SCIP version 3.1.0 (Achterberg, 2009).

Description of the tables: In Table 2.1, we report the performance of our branch-cut-and-price approach applied for the real-world instances after a run time of 36 000 s (ten hours). In order to evaluate the performance of our approach, we use the compact formulation (IP-2) to solve these real-world instances as well. For this end we let CPLEX solve the real-world instances using a standard branch-and-bound (B&B) algorithm based on the IP-2 formulation; and then report the results after a run time of 36 000 s in Table 2.2. Finally we report the performance of our approach applied for the much smaller generated instances after a run time of 7200 s (two hours) in Tables 2.3 and 2.4. A gap limit of 2.0 % has been used for these instances as well.

More precisely, in tables 2.1, 2.3 and 2.4 we first show the instance size as well as the number of path based variables and cuts that were added during the branch-cut-and-price process. Then, we give the time in seconds that was spent solving the root node. The final columns corresponds to the gaps at various stages of the process. All of them have been calculated with respect to the best primal solution found. The LP gap is the average percentage gap of the dual LP bound (before any cuts were added), while the root gab corresponds to the gap of the dual bound after all cuts were added. Finally, we also report the average final gap. In Table 2.3 and 2.4 each row of the table represents the average results for 10 different random instances and we also give the number of instances that could be solved to 2.0 % within two hours.

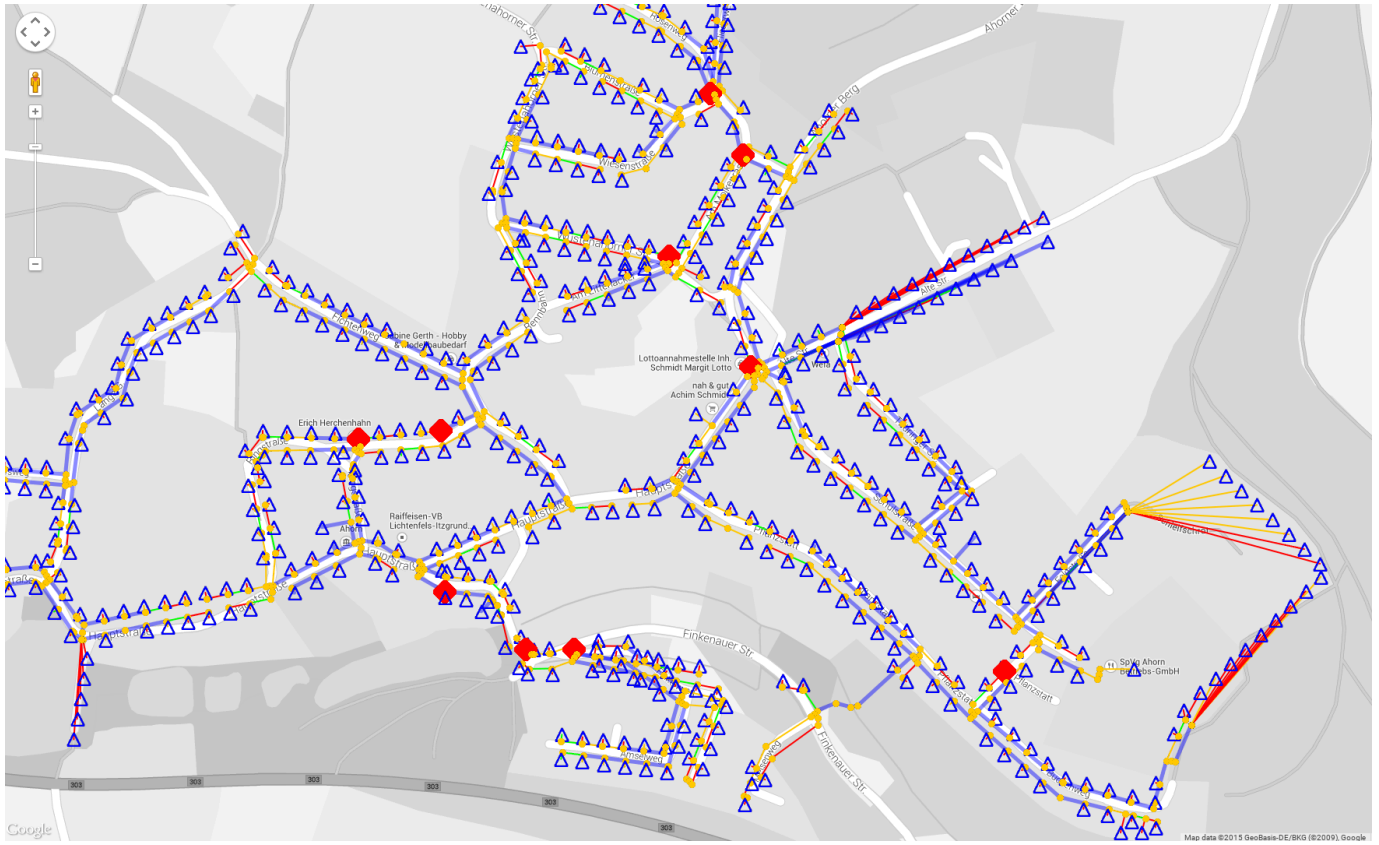


Figure 2.2: Illustration of some part of a solution obtained for input instance "a": squares, triangles and circles represent potential facilities, clients and nodes, respectively. This map has been created using the Google Maps API.

In Table 2.2, we show the number of flow based variables needed in the compact formulation of each (real-world) instance. We then report the time spent by CPLEX to solve the root node as well as the final gap we obtained after a run time of 36 000 s.

Results: Table 2.1 shows that our branch-cut-and-price algorithm together with implemented primal heuristics and problem specific valid inequalities can be used to solve very large real-world instances to roughly 20.0 %. Only instance "e" with over 12 000 edges and "c" with more than 7000 edges lead to slightly worse gaps. Table 2.2, however, shows that the CPLEX solver could not even solve the root node for most of these instances (those with ∞ as the final gap) within the time limit. In particular, comparing these two tables shows the success of our approach in reducing the number of used variables. In fact we believe this is the main reason why our approach based on the path based formulation is doing much better than the one based on the compact formulation.

Tables 2.3 and 2.4 show the performance of our approach for slightly smaller data. Instances JMP (Table 2.3) turn out to be much more challenging (with respect to their sizes). However, we were able to find solutions which are guaranteed to be far less than 5 % away from the optimal solution in most of the cases.

To take a closer look at the performance of the proposed heuristics we refer to Figure 2.3, which shows the progress of the upper bound during the solution process of instance "a". We observe that all approaches already reach good upper bounds within a few minutes. The LP based greedy heuristic is fast and can also return primal solutions that consist of variables currently not in the restricted master problem. The IP based heuristic basically uses the power of all the problem independent primal heuristics used bundled into CPLEX with respect to the currently available variables. This is computational expensive but it also helps to find very good primal solutions. We observe that the hybrid strategy, which uses both types of heuristics, leads to the best results. Here, the greedy heuristic helps to construct new promising paths that can then also be taken into account for the CPLEX heuristic.

Table 2.1: Results of the branch-cut-and-price algorithm on RW instances

inst	$\|V\|$	$\|E\|$	$\|F\|$	$\|D\|$	#path-vars	#cuts	root time [s]	lp-gap [%]	root-gap [%]	final-gap [%]
a	1,675	1,722	104	604	12,079	5,089	864	57.6	18.9	18.2
b	4,110	4,350	230	1,670	23,418	13,692	7,472	74.8	23.7	23.3
c	6,750	7,262	531	2,440	33,211	7,165	36,000	75.0	32.7	32.7
d	4,227	4,482	319	1,490	31,261	10,865	36,000	64.0	20.5	20.5
e	11,544	12,350	890	4,275	43,478	3,759	36,000	80.5	53.0	53.0
f	637	758	101	39	50,739	1,749	266	53.1	19.5	16.1
g	3,055	3,177	49	591	2,976	2,134	61.9	34.3	12.1	10.7
h	2,271	1,419	498	349	32,081	2,325	32,700	56.2	21.5	21.3
i	1,315	1,422	148	238	50,167	5,685	12,360	80.5	16.7	15.9

Table 2.2: Results of the B&B algorithm on RW instances

inst	#flow-vars	root time [s]	final-gap [%]
a	$2.08 \cdot 10^6$	36,000	27.2
b	$1.45 \cdot 10^7$	36,000	∞
c	$3.54 \cdot 10^7$	36,000	∞
d	$1.34 \cdot 10^7$	36,000	∞
e	$1.06 \cdot 10^8$	36,000	∞
f	59,124	347	12.7
g	$3.76 \cdot 10^6$	36,000	∞
h	$9.9 \cdot 10^5$	36,000	∞
i	$6.77 \cdot 10^5$	21,975	15.8

Table 2.3: Results of the branch-cut-and-price algorithm on JMP instances

$\|V\|$	$\|E\|$	$\|F\|$	$\|D\|$	solved	#path-vars	#cuts	lp-gap	root-gap	final-gap
25	48.6	2.4	5.7	10	660.3	68.6	16.1	10.6	2.0
30	60.3	2.4	6.4	10	414.2	88.9	9.6	4.1	2.0
35	70.1	2.4	6.8	9	888.2	94.5	16.8	11.3	2.0
40	88.4	2.4	9.5	7	4,847.1	170	25.1	17.6	2.1
45	99.3	2.7	8.7	10	1,112	135.8	10.1	2.8	2.0
50	114.7	3.8	9.3	8	3,796.2	163.7	17.4	10.2	2.2
55	117.9	3.4	11.6	6	7,568.2	225	17.0	9.4	2.8
60	133.2	3.1	13.7	3	8,866.3	297.2	13.6	6.2	4.1

Table 2.4: Results of the branch-cut-and-price algorithm on PA instances

$\|V\|$	$\|E\|$	$\|F\|$	$\|D\|$	solved	#path-vars	#cuts	lp-gap	root-gap	final-gap
50	97	2.5	9.5	10	511.2	79.4	27.1	4.4	2.0
50	143.7	2.8	9.2	9	4,098.1	130.8	19.3	4.1	2.0
75	147	5.2	12.8	10	1,470	156.2	32.5	4.5	2.0
75	218.7	4.1	13.9	7	7,066.8	261.9	26.6	4.5	2.3
100	197	6.1	18.9	7	5,850.1	271.7	44.5	6.5	2.2
100	293.8	5.1	19.9	2	15,982.4	311.5	37.7	7.2	3.6
125	247	6.3	24.7	7	4,244.8	281.9	52.6	6.2	2.4
125	368.8	6.8	24.2	5	11,709.9	403	41.9	5.2	2.9
150	297	8.2	28.8	5	6,308.5	380.3	59.4	6.5	2.8
150	443.4	7.3	29.7	1	16,461.1	471.8	47.7	6.9	4.0
175	347	9.2	33.8	2	8,858.3	427.6	62.4	7.9	4.1
175	518.8	8.1	34.9	2	12,142.2	607.7	56.2	6.5	4.2
200	397	9.9	40.1	0	8,750.7	515.3	72.0	10.2	4.9
200	593.6	10.5	39.5	0	14,564.7	739	59.8	7.9	5.2
250	497	13.2	48.8	0	12,154.8	687.2	74.6	8.3	5.3
250	743.7	13.2	48.8	0	19,729.3	867.9	61.8	8.0	5.6
300	597	16.1	58.9	0	13,482.4	827.4	79.0	9.3	6.7
300	893.5	17.1	57.9	0	18,309.4	993.6	70.3	9.3	6.3

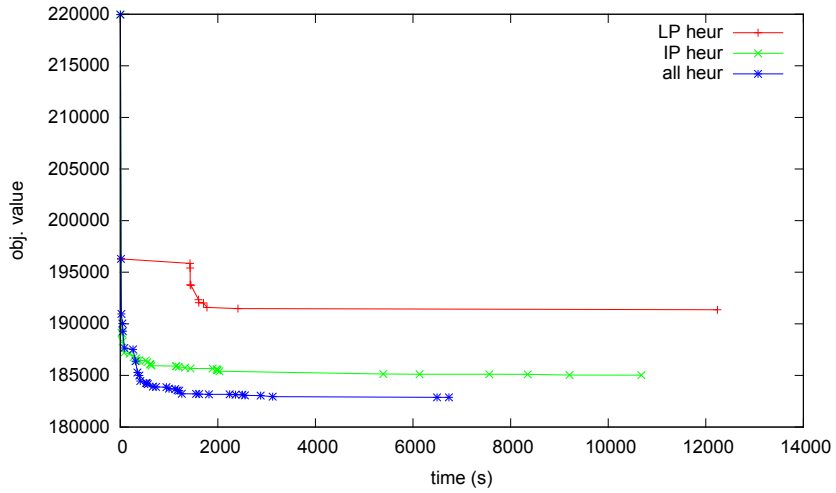


Figure 2.3: Progress of the upper bound for the instance "a" using only the LP based greedy heuristic, only the IP based primal heuristic and both heuristics.

2.5 Conclusion

In this chapter we considered the buy-at-bulk facility location problem. We proposed compact and exponential-sized Integer Programming (IP) formulations for the problem. We presented a branch-cut-and-price algorithm for solving the path-based IP formulation of the problem. We studied the effect of the two families of valid inequalities. Some pre-processing techniques were proposed, which helped in effectively handling sparse instances. We also proposed two efficient IP- and LP-based primal heuristics to obtain good integer solutions. We gave a parallel implementation for the IP heuristic and demonstrated that a hybrid approach to combine the two heuristics provided better primal bounds. We test our approach on a set of real world instances and two different sets of computer generated instances. Using the proposed branch-cut-and-price approaches in combination with the primal heuristics allowed us to solve most of the tested real-world instances with a gap of less than 20 %. We empirically observed that the number of variables generated by the column generation approach is much smaller than the variables in the corresponding compact formulation.

Facilities in real world applications are usually capacitated and this is not considered in the current model. As an extension of our current work, we could model the capacities of the facilities as modules. This could be quite easily incorporated in our current model. One can also consider models where clients need bi-connectivity with open facilities. This could also be solved using a branch-and-price framework with some alterations to the pricing problem.

Chapter 3

Connected Facility Location in Buy-at-Bulk Network Design

In this chapter we focus on obtaining approximation algorithms for two fundamental versions of the buy-at-bulk connected facility location problem, namely buy-at-bulk connected facility location (**BBConFL**) and deep-discount connected facility location (**DDConFL**) problems; see Problem 1.9 and Problem 1.10. Recall that in the Buy-at-Bulk version of the problem, each access cable type has a setup cost and a fixed capacity, whereas in the Deep-Discount problem version, each cable type has unlimited capacity but a traffic-dependent variable cost in addition to its setup cost. We derive the first constant-factor approximation algorithms for these problems using different algorithmic and analytical techniques.

Some results presented in this chapter have been developed in joint work with Andreas Bley (Universität Kassel) and appeared in (Bley et al., 2013a) and (Bley and Rezapour, 2013).

Previous Work

To the best of our knowledge, the problems under consideration in this chapter have not been studied in the literature. However, they are related to the following NP-hard problems.

The BBFL Problem. If we omit the requirement to connect the open facilities by a core network, then the BBConFL problem reduces to the BBFL problem. This combination of the facility location and the buy-at-bulk network problem was first considered by Meyerson et al. (2000). They show that BBFL can be seen as a special case of the Cost-Distance problem (see Section 1.2.2), and thereby provide an $O(\log(|D|))$-approximating

algorithm for this problem, where D is the set of clients. The approximation factor was then improved to $O(K)$ by Ravi and Sinha (2006), where K is the number of cable types. In Section 4.2 we develop a new LP-based approximation algorithm for this problem.

The Single-Sink Network Design Problems. The *single-sink buy-at-bulk* problem and the *Deep-Discount* problem can be viewed as a special case of the BBConFL and the DDConFL problem, respectively. In fact if the set of interconnected facilities is given in advance, then the BBConFL (DDConFL, respectively) problem reduces to the single-sink buy-at-bulk (Deep-Discount, respectively) problem. Recall that two variants of the single-sink buy-at-bulk problem (**SSBB**) exist in the literature, namely splittable SSBB (**sSSBB**) and unsplittable SSBB (**uSSBB**). We remark that, unlike SSBB, one can assume without loss of optimality that the support graph of an optimal solution to the Deep-Discount problem is indeed a tree and, hence, each client's demand is routed unsplit on a single path to its destination (see Garg et al., 2001).

Several approximation algorithms for the SSBB and Deep-Discount problems have been proposed in the literature. Garg et al. (2001) developed an $O(K)$ rounding based approximation for the Deep-Discount problem, and thereby for the uSSBB problem, where K is the number of cable types. The first constant factor approximation algorithm for the Deep-Discount problem is due to Guha et al. (2001, 2009). They also provide an $O(1)$-approximation for uSSBB loosing an extra factor of 2. Later, Talwar (2002) presented an LP based 108-approximation for Deep-Discount, that carries over to a factor 216 approximation for uSSBB. Using sampling techniques, the approximation ratio for uSSBB was improved to 145.6 by Jothi and Raghavachari (2004), and later to 40.82 by Grandoni and Rothvoß (2010). For the sSSBB problem, Gupta et al. (2003) presented a factor 76.8 approximation algorithm using random-sampling techniques. Unlike the previous algorithms, their algorithm does not guarantee that clients route their demands unsplit via a single paths. Modifying Gupta's algorithm, the approximation ratio was later improved to 65.49 by Jothi and Raghavachari (2004), and then to 24.92 by Grandoni and Rothvoß (2010) for sSSBB.

The Connected Facility Location Problem. The ConFL problem can be considered as a special case of BBConFL with only one cable type of unit capacity. The first constant factor approximation algorithm for the problem was given by Gupta et al. (2001) who obtained an LP based 10.66-approximation for this problem. Using the primal-dual technique, the factor was then improved to 8.55 by Swamy and Kumar (2004). Applying sampling techniques, the guarantee was later reduced to 4 by Eisenbrand et al. (2010), and to 3.19 by Grandoni and Rothvoß (2011).

Contributions

In this chapter, we develop the first constant factor approximation algorithms for the DDConFL and BBConFL problems. We extend the approximation for uSSBB in Guha et al. (2009) to DDConFL and BBConFL, thereby establishing a framework for approximations for buy-at-bulk connected facility location variants.

In Section 3.1, we see that the BBConFL and DDConFL problems are closely related, so that a ρ-approximation algorithm for one problem gives a 2ρ-approximation algorithm for the other.

In Section 3.2, we first show that there exist near optimal solutions to the DDConFL problem with a special layered structure. Exploiting this structural property, we then develop the first constant-factor approximation algorithm for the DDConFL problem. The main result of this section is the following:

Theorem 3.1. *There is a polynomial time constant-factor approximation algorithm for the DDConFL problem.*

With the results from Section 3.1, this algorithm also carries over to a constant-factor approximation algorithm for the BBConFL problem, implying the following:

Corollary 3.2. *There is a polynomial time constant-factor approximation algorithm for the BBConFL problem.*

In Section 3.3, we revise our general algorithmic approach. Incorporating and exploiting random sampling techniques, we develop a new algorithm with an improved approximation factor for BBConFL, which, again, carries over to an improved algorithm for DDConFL. The main results of this section are the following:

Theorem 3.3. *There is a polynomial time 192-approximation algorithm for the BBConFL problem.*

Corollary 3.4. *There is a polynomial time 384-approximation algorithm for the DDConFL problem.*

3.1 Preliminaries

In this chapter we study a generalization of the connected facility location problem in which clients are allowed to share connections to facilities. Clients are in fact connected

to open facilities via a network in which on each edge several cable types might be installed in order to accommodate the demand carried by the edge.

Recall that in this problem, we are given a graph $G = (V, E)$ with a set of facilities F with an opening cost μ_i for facility i, a set of clients D with demand d_j for each client j, a weight function $c_e \in \mathbb{Z}_{\geq 0}$, and K different access cables that obey economies of scale. Depending on the settings we consider for access cables, there are two variations of the problem: (i) Buy-at-Bulk Connected Facility Location (BBConFL) and (ii) Deep-Discount Connected Facility Location (DDConFL). In BBConFL, an access cable of type k has a fixed capacity $u_k \in \mathbb{Z}_{>0}$ and setup cost (per unit length) $\sigma_k \in \mathbb{Z}_{\geq 0}$. In DDConFL, an access cable of type k has a setup cost (per unit length) $\sigma_k \in \mathbb{Z}_{\geq 0}$, a flow dependent variable cost (per unit length and per flow unit) of $r_k \in \mathbb{Z}_{\geq 0}$, and infinite capacity. In both BBConFL and DDConFL variants, we are also given an extra type of cable, called core cable, having a cost (per unit length) of $M > \sigma_K$ and infinite capacity, which may be used to connect the open facilities with each other. The task is to open a set of facilities, connect the facility set via a Steiner tree using core cables, and connect clients to open facilities via a network using access cables. In BBConFL, unlike DDConFL, we have to ensure that the installed capacities suffice to route the entire demand of each client along a single path to an open facility via installed access cables. The objective is to minimize the total cost of opening facilities and installing core and access cables. We recall that, in case of DDConFL solutions, we have to add the extra flow dependent cost of the routing paths into account.

These problems are obviously NP-hard, as they contain the classical connected facility location problem (ConFL) as a special case. Therefore the current best lower bound of 1.463 for ConFL (see Grandoni and Rothvoß, 2011) extends to these problems, too.

As it has been observed in the earlier works, one can transform between buy-at-bulk and deep-discount variants of the problem with factor 2 loss. Given an instance of BBConFL with access cables having setup costs σ_k and capacities u_k as defined above, we consider its corresponding DDConFL problem instance, which is obtained by omitting the cable capacity u_k of each access cable k and setting its flow dependent cost to $r_k := \frac{\sigma_k}{u_k}$. It is not hard to see that

$$\left\lceil \frac{D_e}{u_k} \right\rceil \sigma_k c_e \leq \left(\sigma_k + D_e \frac{\sigma_k}{u_k} \right) c_e \leq 2 \left\lceil \frac{D_e}{u_k} \right\rceil \sigma_k c_e$$

holds for any edge e, where D_e is the total demand carried by edge e. Hence the total cost of the access cable installation of any solution of BBConFL is always within a factor of two of the cost of the same access cable installation as the solution to the corresponding modified instance of DDConFL. This immediately implies the following.

Remark 3.5. An ρ-approximation to DDConFL gives a 2ρ-approximation to BBConFL.

Similarly, one can show that an ρ-approximation to BBConFL is a 2ρ-approximation to DDConFL.

3.2 Approximation Algorithm for DDConFL

In this section, we develop the first constant-factor approximation algorithm for DDConFL, extending the algorithm for the uSSBB problem presented by Guha et al. (2009). Guha et al. (2009) proposed an algorithm that constructs a routing network (solution) for uSSBB in multiple stages. They use available cable types one by one, starting with the one with the lowest setup cost, to route the demand from clients to the sink node (in stages). At each stage their approach is to either route (already aggregated) demands to the sink node directly using the cable type considered in that stage, if they are sufficiently large, or gather sufficient demands to justify the use of the next larger cable type, which has a higher setup cost but a lower flow dependent cost. They continue this process for all the cable types until all the demands are routed to the sink. In this section, we extend their approach to our settings which will involve locating the facilities (sinks) and building the core network over facilities, in addition to designing the routing network. We assume that $2\sigma_K < M$. Note that in our and many other applications, it is natural to assume that $\sigma_K << M$.

Preprocessing

For the purpose of analysis it is essential to prune the set of discount cable types such that all cable types are considerably different. Using the following theorem from Guha et al. (2009), this can be done without increasing the cost of the optimal solution too much.

Theorem 3.6. *Given $\alpha \in (0, \frac{1}{2})$, we can prune the set of cables such that, for any i, we have $\sigma_{i+1} > \frac{1}{\alpha} \cdot \sigma_i$ and $r_{i+1} < \alpha \cdot r_i$ hold, and the total cable setup cost of any feasible solution increases by a factor of at most $\frac{1}{\alpha}$.*

Proof Sketch. Consider some feasible solution to DDConFL. Let $1 \leq i < K$ be the largest cable type for which we have $\sigma_i \geq \alpha \cdot \sigma_{i+1}$ holds. We delete cable type i from the set of cable types and replace cable type i by cable type $i+1$ in the solution. We repeat this process until all the remaining cable types satisfy $\sigma_i < \alpha \cdot \sigma_{i+1}$. This results a geometric increase in the setup cost of higher index cables. Notice that these cable

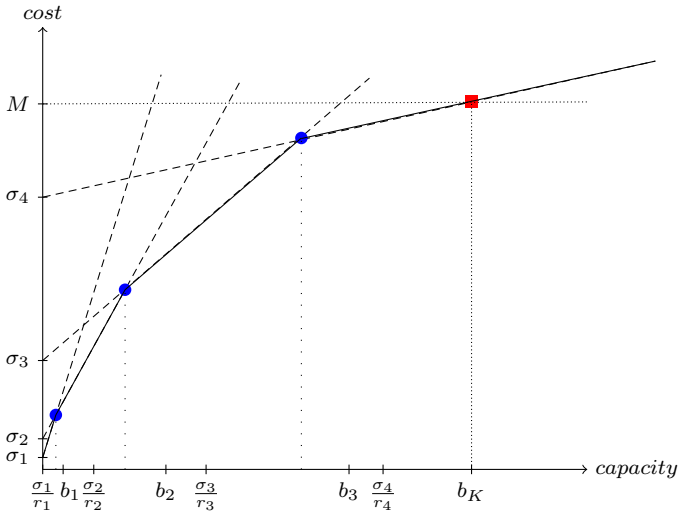


Figure 3.1: A cable setup cost function for DDConFL, where the circles represent the break-points, and the square represents flow at which the cost of using the largest discount cable and the core cable are the same.

replacements only increase the setup cost of the solution by a factor of at most $\frac{1}{\alpha}$, while decreasing the flow dependent cost.

In a similar manner, we can achieve also a geometric decrease in the flow dependent cost of higher index cables with a factor of at most $\frac{1}{\alpha}$ increase in the flow dependent cost. ■

In the proof of Theorem 3.1, we will see that these geometric increase (decrease) in the setup (flow dependent) cost are essential to get a constant approximation. For the ease of notation, let K be the number of cables left after the pruning stage.

In the following, we gain insights into the structure of an (near-)optimal solution to DDConFL by looking carefully at the cable cost structure.

Near-Optimum Layered Solution

We observe that, as demand along an edge increases, there are break-points at which it becomes cheaper to use the next larger cable type. This follows as the cable types obey economies of scale. For $1 \leq i < K$, we define b_i so that $\sigma_{i+1} + b_i r_{i+1} = 2\alpha(\sigma_i + b_i r_i)$. Recall that $\alpha \in (0, \frac{1}{2})$ is a constant and it will be fixed later. Intuitively, b_i is the demand at which it gets more economical to use a cable type $i+1$ rather than a cable type i; see Figure 3.1.

The following lemma is analogous to Lemmas 3.5 and 3.6 in Guha et al. (2009) and it can be proved using nothing more than the definition of break-points b_i.

Lemma 3.7. *For all i, we have $\frac{\sigma_i}{r_i} \leq b_i \leq \frac{\sigma_{i+1}}{r_{i+1}}$. For any i and $D \geq b_i$, we have $\sigma_{i+1} + Dr_{i+1} \leq 2\alpha(\sigma_i + Dr_i)$.*

Let $b_K = \frac{M-\sigma_K}{r_K}$ be the edge flow at which the cost of using cable type K and a core cable are the same (see Figure 3.1). In the following theorem, we show that there exists a solution to the problem with a special layered structure, and whose cost is within a constant factor of the cost of the optimal solution.

Theorem 3.8. *DDConFL has a solution with the following properties:*

(i) The incoming demand of each open facility is at least b_K.

(ii) Cable $i+1$ is used on edge e only if at least b_i demand is routed across e.

(iii) All demand which enters a node, except an open facility, using cable i, leaves that node using cables i or $i+1$.

(iv) The solution's cost is at most $(\frac{4}{\alpha} + 1)$ times the optimum cost.

Proof. Consider an optimum solution of the DDConFL problem. Let I^* be the set of open facilities and let T^* be the core tree connecting them in the optimum solution via core cables. Consider those open facilities whose incoming demand is less than b_K. To meet the first property, we close down these facilities and transfer their corresponding demands to some other open facilities using access cables. In the following we show that the cost of this transmission is at most twice the core network cost of the optimal solution. We double each edge in T^* in order to make the subgraph Eulerian; and take an Euler tour on I^* with shortcuts around any nodes not in I^* or those already visited. We can easily find an unsplittable flow on the edges of this tour sending the aggregated demand from these facilities to the other open facilities such that the resulting solution obeys property (i) and the total flow on any edge of the tour is at most b_K. Recall that for the flow less than b_K it gets cheaper to use access cables rather than core cables. So the cost of transferring this flow on the tour via access cables is at most the cost of transferring it on the tour via core cables, which in turn is at most twice the core network cost of the optimal solution.

Now identify the set of remaining open facilities to a single sink, and update the edge length metric appropriately. The resulting solution can be viewed as a feasible solution to the single-sink network design problem studied by Guha et al. (2009). Results by Guha et al. (2009) show how to convert this solution to a new solution for this single-sink instance which obeys properties (ii) and (iii) with a factor $\frac{2}{\alpha}$ loss in the total access

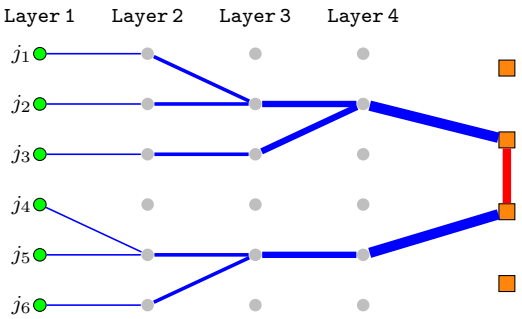


Figure 3.2: An illustration of a layered solution with $K = 4$ as described in Theorem 3.8. For the sake of illustration of the loop edges, we make $K - 1$ additional copies of each client, and put a copy on each layer. In the illustrated solution, for instance, the demand of client j_1 is routed to node j_2 via cable type 2, then from there it is routed to its facility directly, while it is sharing a cable of type 4 with client j_2.

network cost. Note that, as Guha et al. (2009), we assume that there are extra loop-edges such that property (iii) can be enforced for any solution. Hence, we can transform our DDConFL solution to a solution that satisfies properties (ii)–(iv) as well.

This yields a solution to DDConFL which satisfies properties (i)–(iv), and whose cost is at most $(\frac{4}{\alpha} + 1)$ times the cost of the optimum solution. ■

Theorem 3.8, in other words, shows that there exists a sub-optimal solution that uses access cable type i on an edge only if the flow routed along that edge is in the range $[b_{i-1}, b_i]$, $1 \leq i \leq K$, and that uses a facility only when the demand which is served by that facility is at least b_K. This implies that this solution can be viewed as a layered solution where its ith layer only consists of a network of cables of type i carrying a flow of at least b_{i-1} to the next layer or to open facilities if the amount flow is at least b_K; and so the last layer only consists of a network of core cables connecting open facilities (see Figure 3.2). In the next section, we present an algorithm which progressively constructs such a layered solution in a bottom-up fashion.

3.2.1 Algorithm

Our algorithm constructs a layered solution with the properties described in Theorem 3.8. We start from the first cable type and build a solution in $K+1$ stages. At stage i, we aggregate the (already aggregated) demands of value at least b_{i-1} into a smaller subset of nodes using cable type i such that the aggregated demand is at least b_i. Continuing this for all access cable types, all the demand eventually is routed to facilities at stage K where we open a subset of facilities each serving at least b_K demand. We finally construct a network of core cables to connect the open facilities.

More precisely, at stage i, we first aggregate the demands of value at least b_{i-1} to values of at least $\frac{\sigma_i}{r_i}$ using cable type i on the edges of an (approximate) Steiner tree connecting these demands. Then we further aggregate these aggregated demands to values of at least (a constant fraction of) b_i by solving a corresponding *Lower Bounded Facility Location* (LBFL) problem[1], where all nodes may serve as facilities with facility lower bound b_i except for the last phase where only real facilities are eligible. We finally install core cables on the edges of an (approximate) Steiner tree over open facilities to construct the core network.

To solve LBFL, we employ the bicriteria $(\nu \cdot \rho_{FL}, \frac{\nu-1}{\nu+1})$-approximation algorithm devised by Guha et al. (2000), which relaxes the lower bound on the minimum demand served by a facility by a factor $\beta = \frac{\nu-1}{\nu+1}$ and returns a solution within $\nu \cdot \rho_{FL}$ times the optimal cost, where ρ_{FL} is the best known approximation for the facility location problem.

Notice that when the flow is in the range $[b_{i-1}, \frac{\sigma_i}{r_i}]$, the setup cost for using cable type i on an edge is greater than the flow dependent cost of using cable type i on that edge. So by loosing a factor of at most 2, one can ignore the flow dependent cost of the routing and just relay on Steiner trees to aggregate the demands of value at least b_{i-1} to values of at least $\frac{\sigma_i}{r_i}$. With a similar argument, one can consider only the flow dependent cost and relay on shortest path trees to route the demands of values at least $\frac{\sigma_i}{r_i}$ to values b_i, by loosing a factor of at most 2. In fact this is why in the aggregation steps of the algorithm Steiner trees and shortest path trees are used alternatively (see Aggregate I and II steps of the algorithm).

A complete description of the stages of this algorithm (**AlgDD**) is given on Page 66.

One can easily verify that **AlgDD** computes a feasible solution in polynomial time. It remains to show that the returned solution is an approximate solution.

3.2.2 Analysis

Let C_i^*, S^*, and O^* be the amount paid for cables of type i, for the core Steiner tree, and for opening facilities in the near-optimal solution, respectively. We define C_i to be the total cost paid for cables of type i in the returned solution. Let D_j^i be the demand of node j at stage i of the algorithm. Let T_i, P_i and N_i be cost incurred in the Aggregate-I step, the Aggregate-II step, and the both Consolidate steps of iteration i, respectively. Also, let T_i^F and T_i^S denote the flow dependent and the setup cost components of the Aggregate-I step at iteration i. Analogously, P_i^F and P_i^S denote the flow dependent and

[1]The LBFL problem is a generalization of the facility location problem where each open facility is required to serve a certain minimum amount of demand (see Guha et al., 2000; Svitkina, 2010; Ahmadian and Swamy, 2012).

AlgDD– Approximation algorithm for **DDConFL**

1. Prune the set of cable types, as described in Theorem 3.6; let $D_1 = D$, and let D_i be the set of demand points we have at the i-th stage; guess a facility r from the optimum solution.

2. **For cable type** $i = 1, 2, ..., K-1$ **Do**

   - **Aggregate I:** Construct a ρ_{ST}-approximate Steiner tree T_i on demand points $D_i \cup \{r\}$ for edge costs σ_i per unit length. Install a cable of type i on each edge of this tree. Root this tree at r. Send the demands from D_i upwards along the tree. Walking upwards along this tree, identify edges whose demand is larger than $\frac{\sigma_i}{r_i}$ and cut the tree at these edges.

   - **Consolidate I:** For every tree in the forest created in the preceding step, transfer the total demand at the root of the tree, which is at least $\frac{\sigma_i}{r_i}$, back to one of its sources using a shortest path of cable type i. Choose this source with a probability proportional to the demand at the source.

   - **Aggregate II:** Solve the LBFL problem with (actual) demands D as clients, all nodes as facilities each having opening cost 0 and facility lower bound b_i, and edge costs r_i per unit length. The solution is a forest of shortest path trees. Then route the *current* demands (that is the demand aggregated on each node right after the Consolidate step) along these shortest path trees to their roots, by installing cables of type i.

   - **Consolidate II:** For every tree in the forest created in the preceding step, transfer the total demand at the root of the tree, which is at least b_i, back to one of its sources with a probability proportional to the demand at that source using a shortest path with cables of type i. Let D_{i+1} be the resulting set of demand locations.

3. **For cables type** K **Do**

   - Construct a ρ_{ST}-approximate Steiner tree T_K on demand points $D_K \cup \{r\}$ for edge costs σ_K per unit length. Install cables of type K on each edge of this tree. Root this tree at r. Send the demands from D_K upwards along the tree. Walking along this tree, identify edges whose demand is larger than $\frac{\sigma_K}{r_K}$ and cut the tree at these edges. For every tree in the created forest, send the total demand in the root of tree back to one of its sources with a probability proportional to the demand at that source via a shortest path using cables of type K.

   - Solve the LBFL problem with (actual) clients D, facility set F, opening costs μ_i, facility lower bound b_K, and edge costs r_K per unit length. We obtain a forest of shortest path trees. Then route the *current* demands along these trees to their roots, installing cables of type K. Let F' be the set of open facilities.

4. Compute a ρ_{ST}-approximate Steiner tree T_{core} on nodes $F' \cup \{r\}$ for edge costs M per unit length. Install the core cable on the edges of T_{core}.

the setup costs incurred in the Aggregate-II step. Recall that the set of discount cable types has been reduced depending on the constant parameter $\alpha \in (0, \frac{1}{2})$.

The following Lemma carries over from the uSSBB problem studied by Guha et al. (2009) to our problem in a straightforward way.

Lemma 3.9.

(i) At the end of each consolidation step every node has $E[D_j^i] = d_j$.

(ii) $E[N_i] \leq T_i + P_i$ *for each* i.

(iii) $P_i^S \leq P_i^F$ *and* $T_i^F \leq T_i^S$ *for each* i.

The following lemma bounds the setup costs of the cables installed in the Aggregate-I step of phase i of our algorithm.

Lemma 3.10. $E[T_i^S] \leq \rho_{ST}\big(\sum_{j=1}^{i-1} \frac{1}{\beta}(2\alpha)^{i-j}C_j^* + \sum_{j=i}^{K} \alpha^{j-i}C_j^* + \frac{1}{2}\alpha^{K-i}S^*\big)$

Proof. We prove the claim by constructing a Steiner tree of cables of type i on nodes $D_i \cup \{r\}$ whose expected cost is at most $\sum_{j=1}^{i-1} \frac{1}{\beta}(2\alpha)^{i-j}C_j^* + \sum_{j=i}^{K} \alpha^{j-i}C_j^* + \frac{1}{2}\alpha^{K-i}S^*$. Consider the near-optimum solution. We find an unsplittable flow sending the demand aggregated at points D_i to facility r via this network solution. Then, out of edges of the near-optimum solution, we keep only those edges carrying positive flow and replace their (access and/or core) cables with access cables of type i. This obviously results a tree of cables i on nodes $D_i \cup \{r\}$. In the following we bound the expected cost of this network.

Note that, being in stage i, the expected demand of each point in D_i is at least βb_i. Hence the expected flow on each cable $j < i$ is at least βb_i as well. Therefore, by Lemma 3.7, the expected cost of all replacement cables for cables of type $j < i$ is bounded by $\frac{1}{\beta}(2\alpha)^{i-j}C_j^*$. Furthermore, the expected cost of the replacement cables for the cables $j > i$ can be easily bounded by $\alpha^{j-i}C_j^*$, using the setup costs scale (see Theorem 3.6).

Finally, with a similar argument, the cost of replacement for the core cables is bounded by $\frac{1}{2}\alpha^{K-i}S^*$; recall that $2\sigma_K < M$.

Altogether, the expected setup cost of this Steiner tree using cables i is bounded by

$$\sum_{j=1}^{i-1} \frac{1}{\beta}(2\alpha)^{i-j}C_j^* + \sum_{j=i}^{K} \alpha^{j-i}C_j^* + \frac{1}{2}\alpha^{K-i}S^*$$

As we use a ρ_{ST}-approximation algorithm to solve this Steiner tree problem in our algorithm, the claim follows. ∎

In the following lemma, we bound the flow dependent costs of the cables installed in the Aggregate-II step of phase i of our algorithm.

Lemma 3.11. $E[P_i^F] \leq \nu \cdot \rho_{FL} \sum_{j=1}^{i} \alpha^{i-j} \cdot C_j^*$

Proof. To prove this, similar to that of Lemma 3.10, we construct a feasible solution to the LBFL problem solved in this stage by making use of the near-optimum solution. For this end, we only consider the forest defined by the edges with cable types 1 to i in the near-optimum solution and replace all cables of type less than i by cables of type i. This results a feasible solution for this LBFL problem as the root node of each tree of the resulting forest has incoming demand of at least b_i (recall property (ii) of Theorem 3.8). The flow dependent cost of the new solution on cables replaced by cables of type $j < i$ is bounded by $\alpha^{i-j} \cdot C_j^*$ using the flow dependent costs scale. Hence, the cost of the resulting solution is at most $\sum_{j=1}^{i} \alpha^{i-j} C_j^*$. As our algorithm applies a bicriteria $\nu \cdot \rho_{FL}$-approximation algorithm to solve the lower bounded facility location problem in this stage, the claim follows. ■

The opening costs and the flow dependent shortest path costs in the K-th stage of our algorithm can be bounded as follows.

Lemma 3.12. $E[P_K^F + \mu(F')] \leq \nu \cdot \rho_{FL}(\sum_{i=1}^{K} \alpha^{K-i} \cdot C_i^* + O^*)$

Proof. The proof is similar to the one for the previous lemma. We consider the access network of the near-optimum solution as a whole and replace all access cables (of type less than K) by cables of type K. This, combined with facilities opened in the near-optimum solution results a solution for this stage whose cost is at most $\sum_{i=1}^{K} \alpha^{K-i} C_i^* + O^*$. ■

Let S be the cost of the core Steiner tree returned by our algorithm. This can be bounded as follows.

Lemma 3.13. $E[S] \leq \rho_{ST}\big(S^* + \frac{2}{\beta} \sum_{j=1}^{K} (C_j^* + C_j)\big)$

Proof. Let F^*, T_{core}^* and T_{access}^* be the set of open facilities, the tree connecting them, and the forest connecting clients to open facilities in the near-optimum solution, respectively. Let T_{access} be the forest connecting clients to open facilities in the solution returned by our algorithm. We prove the claim by constructing a feasible Steiner tree of core cables on $F' \cup \{r\}$, whose expected cost is at most $S^* + \frac{2}{\beta} \sum_{j=1}^{K} (C_j^* + C_j)$.

Consider the near-optimum solution combined with the solution returned by the algorithm. In the algorithm's solution, each facility $l \in F'$ serves at least a total demand

of βb_K. This demand, on the other hand, is served by the set of facilities in the near-optimum solution. Therefore, at least βb_K flow can be routed (possibly along more than one path) between each facility $l \in F'$ and the set of facilities in F^* using the access cables on edges of $T^*_{access} \cup T_{access}$ (without violating the cables' capacities). By contracting the facilities F^* into a single node $\tilde{f}$, this guarantees that the access cables on edges of $T^*_{access} \cup T_{access}$ allow at least βb_K flow crossing each cut that contains at least one facility in F' and not containing $\tilde{f}$. Using the fact that it is cheaper to use core cables rather than access cables when the demand is at least b_K, we can deduce that the cost of a feasible (possibly fractional) solution to the core Steiner tree problem over $F' \cup \{\tilde{f}\}$ is at most $\frac{1}{\beta}\sum_{j=1}^{K}(C^*_j + C_j)$. Since the integrality gap of the cut formulation of Steiner tree is at most 2, one can then use any (LP-rounding) approximation algorithm to construct a core Steiner tree on $F' \cup \{\tilde{f}\}$ with cost at most $\frac{2}{\beta}\sum_{j=1}^{K}(C^*_j + C_j)$.

Finally, we uncontract node $\tilde{f}$ and augment the resulting forest by adding edges in T^*_{core} to get a core Steiner tree on $F' \cup \{r\}$. This completes the proof. ■

Together, Lemmas 3.9–3.13 imply our main result.

Proof of Theorem 3.1.

Proof. By Lemmas 3.9–3.12, the total expected cost of access cables is bounded as follows:

$$\sum_{i=1}^{K} C_i \le 4\sum_{i=1}^{K}\Big[\nu\cdot\rho_{FL}\sum_{j=1}^{i}\alpha^{i-j}C^*_j + \rho_{ST}\Big(\sum_{j=i}^{K}\alpha^{j-i}C^*_j + \sum_{j=1}^{i-1}\frac{1}{\beta}(2\alpha)^{i-j}C^*_j + \frac{1}{2}\alpha^{K-i}S^*\Big)\Big]$$
$$\le 4\Big(\frac{\nu\cdot\rho_{FL}+\rho_{ST}}{1-\alpha} + \frac{\rho_{ST}}{\beta(1-2\alpha)}\Big)\sum_{i=1}^{K}C^*_i + \frac{2\cdot\rho_{ST}}{1-\alpha}S^* \tag{3.1}$$

Additionally, using Lemmas 3.12 and 3.13, the total cost of installing core cables and opening facilities is bounded by

$$\nu\cdot\rho_{FL}\cdot O^* + \rho_{ST}\cdot S^* + \frac{2\rho_{ST}}{\beta}\Big(\sum_{i=1}^{K}C^*_i + \sum_{i=1}^{K}C_i\Big)\,.$$

Altogether, using Inequality (3.1) twice, we obtain a bound of

$$\nu\cdot\rho_{FL}\cdot O^* + \Big[\frac{2\rho_{ST}}{\beta} + 4\big(1+\frac{2\rho_{ST}}{\beta}\big)\big(\frac{\nu\cdot\rho_{FL}+\rho_{ST}}{1-\alpha} + \frac{\rho_{ST}}{\beta(1-2\alpha)}\big)\Big]\sum_{i=1}^{K}C^*_i$$
$$+\Big[\big(1+\frac{2\rho_{ST}}{\beta}\big)\big(\frac{2\rho_{ST}}{1-\alpha}\big) + \rho_{ST}\Big]S^*$$

for the worst case ratio between the algorithm's solution and a near optimal solution of DDConFL, according to Theorem 3.8.

With Theorems 3.6 and 3.8, this yields a worst case approximation guarantee of $\frac{1}{\alpha}(\frac{4}{\alpha}+1)$ times the above ratio against an unrestricted optimal solution of the DDConFL problem. Hence, the approximation guarantee of our algorithm can be bounded by

$$\frac{4+\alpha}{\alpha^2} \cdot \max\Big(\nu.\rho_{FL}, \frac{2\rho_{ST}}{\beta} + 4\big(1+\frac{2\rho_{ST}}{\beta}\big)\big(\frac{\nu\cdot\rho_{FL}+\rho_{ST}}{1-\alpha} + \frac{\rho_{ST}}{\beta(1-2\alpha)}\big), \rho_{ST}\big(\frac{2}{1-\alpha} + \frac{4\rho_{ST}}{\beta(1-\alpha)} + 1\big)\Big)$$

Setting $\rho_{ST} = ln(4)$ due to Byrka et al. (2010), $\rho_{FL} = 1.488$ due to Li (2013), α to some value in $(0, \frac{1}{2})$, and β to some value in $(0, 1)$ (and thereby ν to its corresponding value) we obtain the first constant factor approximation algorithm for DDConFL. ■

Theorem 3.1 together with Remark 3.5 implies Corollary 3.2.

One can observe that the approximation ratios obtained by this algorithm (even after setting parameters α and β to their best values) are larger than one thousand.

In the next section, we revise our general algorithmic approach to improve the factors. Incorporating and exploiting random sampling techniques, we develop a new algorithm with an improved approximation factor for BBConFL, which, carries over to an improved algorithm for DDConFL as well.

3.3 Approximation Algorithm for BBConFL

In this section, we present our improved approximation algorithm for the BBConFL problem applying random sampling techniques.

Recall that an access cable of type i, in BBConFL, has a setup cost (per unit length) σ_i and fixed capacity u_i (but no flow dependent cost). With the same argument as for the one used in **AlgDD**, it is essential to prune the set of cable types such that all cable types are considerably different. We hence assume that all values u_i and σ_i are powers of 2, and that $u_1 = \sigma_1 = 1$. This can be easily enforced by increasing the cost of any solution by a factor of at most 4 (see Theorem 3.6).

Remember that due to economies of scale, as demand along an edge increases, there are break-points at which it becomes cheaper to use the next larger cable type. Recall that inequality $\frac{\sigma_i}{u_i} > \frac{\sigma_{i+1}}{u_{i+1}}$ holds for all i. We hence have $u_i < g_i < u_{i+1}$ holds for all i, where $g_i = \frac{\sigma_{i+1}}{\sigma_i} \cdot u_i$. In fact (analogous to b_i in Section 3.2) g_i is the demand at which it gets

more cost-effective to use a cable type $i+1$ rather than a cable type i, implying that for routing $\bar{f} \in [g_i, u_{i+1}]$ flow it is cheaper to use a single copy of cable type $i+1$ rather than $\lceil \frac{\bar{f}}{u_i} \rceil$ copies of cable type i.

Our algorithm follows the same high-level approach as that of **AlgDD**, where the solution is designed incrementally in a bottom-up manner. Starting form the first cable type, in each stage the demand is aggregated into a smaller subset of nodes such that the aggregated demand suffices to use the next larger cable type effectively. Eventually, using the largest cable type, all the demand is routed to open facilities which are next connected via core links. We should remark that early decisions on where to install access cables effect (later) decisions on where to open facilities and how to interconnect them. Therefore, they must be taken carefully.

In order to aggregate the demand from clients at each stage of the algorithm, we apply the redistribution technique introduced by Gupta et al. (2003). They proposed a simple randomized aggregation algorithm that can be used to aggregate demands to a given value via a Steiner tree over the demand points. In our algorithm, however, we can not directly apply their approach as it does not guarantee an unsplittable flow (and so the demand from clients might be routed via several routing paths). Instead, we apply a modification of their redistribution technique that guarantees unsplittable flow as proposed by Jothi and Raghavachari (2004).

Lemma 3.14 (Jothi and Raghavachari (2004))**.** *Let T be a tree with each edge having capacity U. For each vertex v in T, let $x(v) < U$ be the demand originating at v. Assume all values U and $x(v)$, $v \in T$, are powers of 2. There is an efficient randomized algorithm that computes a flow on T that respects the edge capacities and redistributes the demands without splitting any demand such that each vertex receives a new demand $\hat{x}(v) \in \{0, U\}$ and, moreover, $\mathbf{Pr}[\hat{x}(v) = U] = \frac{x(v)}{U}$ for all $v \in T$.*

Recall that it is assumed that all values u_i and σ_i are powers of 2. We also assume that each value d_j, $j \in D$, is a power of 2, losing another factor of at most 2 in the approximation ratio. This permits us to use the above redistribution technique in our algorithm.

Finally, for the sake of simplicity, we assume w.l.o.g. that $d_j = 1$ for all j, replacing j by several copies of co-located unit-demand clients. The algorithm presented in this chapter can be adapted to ensure that those demands travel together along the same path towards its destination (see Jothi and Raghavachari, 2004), and that it runs in polynomial time even when the (original) demands are not polynomially bounded (see Gupta et al., 2003). By adding dummy demands, we can also assume that the number of demands $|D|$ is a power of 2. We are now ready to present our algorithm.

3.3.1 Algorithm

Our algorithm will follow the same general ideas of that for DDConFL and construct a solution in a bottom-up manner. The algorithm starts with all demand nodes $D_1 = D$ in Step 1. Then, repeatedly, in stage i of Step 2 it aggregates the demands from the demand nodes D_i into a smaller, randomly sampled node set D_{i+1} using only cables of type i and $i+1$. Eventually, in Steps 3–7, the demands are aggregated from the nodes in D_k to a randomly sampled subset of the facilities, which is then connected by a core Steiner tree. A complete description of the steps of the algorithm, called **AlgBB**, is given on Page 73.

In this Algorithm, $l(v,u)$ denotes the distance between v and u in G. Let $l(v,U) = \min_{u\in U} l(v,u)$.

One can easily verify that **AlgBB** computes a feasible solution in polynomial time. In the next section we will show that the computed solution is an approximate solution.

3.3.2 Analysis

To analyze the performance of **AlgBB**, we introduce some additional notation. Let OPT denote an optimal solution with access cable cost $C^* = \sum_{t=1}^{K} C_t^*$, where C_t^* is the amount paid for cables of type t in OPT, core Steiner cost S^*, and opening cost O^*. Let F^* and T_{core}^* denote the set of open facilities and the Steiner tree connecting them in OPT, respectively. Let σ^* indicate the mapping of the clients to facilities F^* in OPT: For each $j \in D$, $\sigma^*(j) \in F^*$ indicates the facility that client j is served by in OPT. Let $\delta_i = \frac{\sigma_i}{u_i}$.

The proof of the following lemma is similar to that of Lemma 4.2 by Gupta et al. (2003) and is omitted here.

Lemma 3.15. *For every client $j \in D$ and stage i, $1 \le i \le K$, $\mathbf{Pr}[j \in D_i] = 1/u_i$.*

Let A_i be the total cost incurred in Stage i of Step 2 of the algorithm. This can be bounded as follows.

Lemma 3.16. $\mathbf{E}[A_i] \le \sigma_{i+1}(\rho_{ST}+3)\left[\frac{1}{M}S^* + \sum_{t>i}\frac{1}{\sigma_t}C_t^* + \frac{\delta_i}{\sigma_{i+1}}\sum_{t\le i}\frac{1}{\delta_t}C_t^*\right]$

Proof. Consider Stage i of Step 2 of **AlgBB**. Let $\hat{T}_i$ denote the optimal Steiner tree on F_i and let $c(\hat{T}_i)$ be the cost of this tree. Note that the cost of the cables (of type $i+1$) installed on the edges of T_i (in Part b of this stage) is $\sigma_{i+1} \cdot \rho_{ST} \cdot c(\hat{T}_i)$.

AlgBB– Approximation algorithm for **BBConFL**

1. Let $D_1 = D$. Guess a facility r to open in the solution.
2. **For stage** $i = 1, 2, ..., K-1$ **do**
   a. Mark each client in D_i with probability $p_i = \frac{\sigma_i}{\sigma_{i+1}}$. Let $D_i^{'}$ be the marked clients.
   b. Construct a ρ_{ST}-approximate Steiner tree T_i having nodes $F_i = D_i^{'} \cup \{r\}$ as terminals. Install cable type $i+1$ on each $e \in T_i$.
   c. For each client in D_i, send its demand to the nearest $j \in F_i$ via a shortest path, using cables of type i. Let $d_i(j)$ be demand aggregated at $j \in F_i$.
   d. Redistribute demands $\bar{x}(j) := d_i(j) \ mod \ u_{i+1}$, $j \in F_i$ using Lemma 3.14 with $T = T_i$, $U = u_{i+1}$, and $x(j) = \bar{x}(j)$. The corresponding flow is supported by cables of type $i+1$ installed at Phase b of the current stage. Note that the new demand aggregated at node $j \in F_i$ is now a multiple of u_{i+1}.
   e. For each $j \in F_i$, let $D_i(j) \subseteq D_i$ be the nodes sending demand to j in stage i (including j itself, if $j \neq r$). Partition $D_i(j)$ into groups of $\frac{u_{i+1}}{u_i}$ nodes, each with the total demand u_{i+1}. Send the total demand of each group back to a random member of that group via shortest paths, installing cables of type $i+1$. Let D_{i+1} be the resulting demand locations.
3. Define an instance of the facility location problem having D_K as the set of clients, F as the set of facilities, and assignment cost $c_{ji} = \sigma_K.l(j,i)$, $j \in D_K, i \in F$. Compute a ρ_{FL}-approximate solution $U = (F_U, \sigma_U)$ to this instance, where F_U and σ_U indicate the set of open facilities and the mapping of the clients to these facilities, respectively. Let $\sigma_U^{-1}(i)$ be the set of clients assigned to facility $i \in F_U$ in U.
4. Mark each client in D_K with probability $p_K = \frac{\sigma_K}{M}$. Let $D_K^{'}$ be the marked clients.
5. Open facility $i \in F_U$ if some client in $\sigma_U^{-1}(i)$ is marked. Let I be the set of open facilities.
6. Compute a ρ_{ST}-approximate Steiner tree T_K having $F_K = D_K^{'} \cup \{r\}$ as the set of terminals. Complete this tree by adding edges of the shortest path connecting each $j \in D_K^{'}$ to its corresponding open facility $\sigma_U(j) \in I$. Let T_{core} be a tree spanning I extracted from this subgraph. Install core cables on T_{core}
7. Connect each client in D_K to a closest open facility in $I \cup \{r\}$, via a shortest path, using cables of type K.

By an argument similar to the one used in Lemma 4.4 of Gupta et al. (2003), it can be shown that (i) the cost of the cables (of type i) used to aggregate demand on F_i, incurred in Part c, is at most $2\sigma_{i+1} \cdot c(\hat{T}_i)$, and (ii) the cost of the cables (of type $i+1$), incurred in Part e, to reroute the demands back to random vertices in D_i is at most $\sigma_{i+1} \cdot c(\hat{T}_i)$. Hence

$$\mathbf{E}[A_i] \leq \sigma_{i+1}(\rho_{ST} + 3)c(\hat{T}_i) \tag{3.2}$$

Therefore, to bound the total cost incurred in Stage i of the algorithm, we only need to come up with a bound for $c(\hat{T}_i)$. For this end, we construct a feasible Steiner tree T on F_i as follows. First, add the optimal Steiner tree T^*_{core} and all edges with cable type $i+1$ or higher in OPT to T. Obviously, the cost of the resulting subgraph is at most $\frac{1}{M}S^* + \sum_{t>i} \frac{1}{\sigma_t} C^*_t$.

Remember that the demand of each client is allowed to be routed only along a single path. Let P^*_j be the routing path connecting client j to $\sigma^*(j)$ in OPT. Now, to obtain a tree connecting nodes in F_i, we augment T by adding all the missing edges from paths P^*_j, $j \in F_i$. Let T_{new} be the set of edges added in this point. Note that these new edges have only cables of type $t \leq i$ used in OPT (as $T \setminus T_{new}$ already contain all edges with cable type $i+1$ or higher). For the sake of simplicity, we suppose that only one cable type $t \leq i$ is installed on each $e \in T_{new}$ in OPT (a similar argument will hold for the general case). Now, consider some edge e with cable type $t \leq i$ in OPT. Observe that this edge might be added to T_{new} only if it is part of some routing path P^*_j, $j \in F_i$. Therefore, as only up to u_t routing paths can be crossed along this edge, we have $\mathbf{P}[e \in T] \leq \frac{u_t}{u_i} \cdot \frac{\sigma_i}{\sigma_{i+1}}$. (This follows from the fact that $\mathbf{P}[j \in F_i] = \mathbf{P}[j \in D'_i \mid j \in D_i] \cdot \mathbf{P}[j \in D_i] \leq \frac{1}{u_i} \cdot \frac{\sigma_i}{\sigma_{i+1}}$ by Lemma 3.15). Finally, summing over all edges with cables of type at most i, the expected cost of edges in T_{new} is bounded by $\sum_{t \leq i} \frac{u_t}{u_i} \cdot \frac{\sigma_i}{\sigma_{i+1}} \cdot \frac{1}{\sigma_t} C^*_t = \frac{\delta_i}{\sigma_{i+1}} \sum_{t \leq i} \frac{1}{\delta_t} C^*_t$.

Extracting a tree spanning F_i from T, we obtain

$$\mathbf{E}[c(\hat{T}_i)] \leq \mathbf{E}[c(T)] \leq \frac{1}{M} S^* + \sum_{t>i} \frac{1}{\sigma_t} C^*_t + \frac{\delta_i}{\sigma_{i+1}} \sum_{t \leq i} \frac{1}{\delta_t} C^*_t \tag{3.3}$$

Together, (3.2) and (3.3) imply the claimed bound.

■

Let O_U be the opening cost and C_U be the connection cost of the solution to the facility location problem computed in Step 3.

Lemma 3.17. $\mathbf{E}[O_U + C_U] \leq \rho_{FL}(O^* + \sum_{i=1}^{K} \frac{\delta_K}{\delta_i} C^*_i)$

Proof. We obtain a feasible solution for the facility location problem considered in Step 3 by connecting each client $j \in D_K$ to its BBConFL optimal facility $\sigma^*(j) \in F^*$ via a shortest path in G. Its expected cost is at most

$$O^* + \sigma_K \cdot \mathbf{E}\big[\sum_{j \in D_K} l(j, F^*)\big] = O^* + \delta_K \sum_{j \in D} l(j, F^*) \leq O^* + \sum_{i=1,\dots,K} \frac{\delta_K}{\delta_i} C_i^*$$

The last inequality follows from the fact that

$$\sum_{j \in D} l(j, F^*) \leq \sum_{i=1}^{K} \frac{u_i}{\sigma_i} C_i^* \tag{3.4}$$

Therefore the total cost of the approximate solution computed in Step 3 is at most $\rho_{FL}(O^* + \sum_{i=1}^{K} \frac{\delta_K}{\delta_i} C_i^*)$. ■

To bound the cost of cable installation in Step 7, we will apply the core connection game described in the following lemma which was proved by Eisenbrand et al. (2010) (see Theorems 2 and 5 in Eisenbrand et al. (2010) for more details).

Lemma 3.18. *Given an undirected graph $G = (V, E)$ with weight $w_e \in \mathbb{Z}_{\geq 0}$, $e \in E$; a set of clients $\hat{C} \subseteq V$; a core tree $\hat{T} \subseteq E$; a mapping $\eta : \hat{C} \to V(\hat{T})$; two oppositely directed and weighted edges $(j, \eta(j))$ and $(\eta(j), j)$, $\forall j \in \hat{C}$; and a probability $p \in (0, 1]$. Let H_{in} be the set of all edges that are directed from client nodes to core nodes $V(\hat{T})$ and H_{out} be the set of all oppositely directed edges. Consider the following random core connection game: Sample each client independently with probability p, and denote the sampled clients by C'. Then, connect every client $j \in \hat{C}$ to the closest sampled client in C'. The total cost of this connection game is bounded by*

$$\mathbf{E}\big[\sum_{j \in \hat{C}} l(j, C')\big] \leq \frac{0.807}{p} w(\hat{T}) + w(H_{out} \cup H_{in})$$

Roughly speaking, this lemma states how to bound the distances from a set of clients (connected to a core tree) to a sampled subset of them (against the cost of the core tree).

Let A_K be the total cost incurred in Stage i of Step 2 of the algorithm.

Lemma 3.19. *The cost of cable installation in Step 7 of **AlgBB** satisfies $\mathbf{E}[A_K] \leq 2 \sum_{i=1}^{K} \frac{\delta_K}{\delta_i} C_i^* + 0.807\, S^* + C_U$.*

Proof. We apply the core connection game described in Lemma 3.18 with clients $\hat{C} = D_K$, core $\hat{T} = T^*_{core}$, mapping $\eta = \sigma^*$, and probability $p = \frac{\sigma_K}{M}$. We set $w(e) = c(e)$,

for every edge e in tree $\hat{T}$. For every directed edge $(j, \eta(j))$ in H_{in}, we set its weight to be $l(j, \sigma^*(j)$. For every directed edge $(\eta(j), j)$, we define its weight to be $l(j, \sigma^*(j)) + l(j, \sigma_U(j))$. This yields

$$\begin{aligned}\sigma_K \cdot \mathbf{E}\Big[\sum_{j \in D_K} l(j, I)\Big] &\leq \sigma_K \Big[\frac{0.807}{\sigma_K / M} \cdot \frac{S^*}{M} + \frac{2}{u_K} \sum_{j \in D} l(j, F^*) + \frac{1}{\sigma_K} C_U)\Big] \\ &\leq 0.807\, S^* + 2\delta_K \sum_{j \in D} l(j, F^*) + C_U \end{aligned} \tag{3.5}$$

Together, (3.4) and (3.5) imply that

$$\mathbf{E}[A_K] \leq 0.807\, S^* + 2 \sum_{i=1}^{K} \frac{\delta_K}{\delta_i} C_i^* + C_U$$

■

Finally, the expected cost of approximate Steiner tree T_{core}, computed in Step 6, can be bounded as follows.

Lemma 3.20. $\mathbf{E}[T_{core}] \leq \frac{\rho_{ST}}{M}\big(S^* + \sum_{i=1}^{K} \frac{\delta_K}{\delta_i} C_i^*\big) + \frac{1}{M} C_U$

Proof. We first construct a feasible Steiner tree on clients in $D_K^{'}$. This can be done by augmenting the optimal Steiner tree T_{core}^* by adding edges of the shortest path connecting each client in $D_K^{'}$ to T_{core}^*. Using Lemma 3.15 as well as Inequality (3.4), the expected cost of this tree is at most

$$\frac{1}{M} S^* + \mathbf{E}\big[\sum_{j \in D_K} l(j, F^*)\big] = \frac{1}{M} S^* + \frac{\sigma_K}{u_K \cdot M} \sum_{j \in D} l(j, F^*) \leq \frac{1}{M}\big(S^* + \sum_{i=1}^{K} \frac{\delta_K}{\delta_i} C_i^*\big)$$

This implies that the expected cost of tree T_K (computed in Step 6) is at most $\frac{\rho_{ST}}{M}\big(S^* + \sum_{i=1}^{K} \frac{\delta_K}{\delta_i} C_i^*\big)$.

Now, we obtain the claimed bound by adding to this the expected cost of connecting each client $j \in D_K^{'}$ to facility $\sigma_U(j) \in F$ which is at most $\frac{\sigma_K}{M} \sum_{j \in D_K} l(j, F_U) = \frac{1}{M} C_U$. ■

Together, Lemmas 3.16–3.20 imply our main result.

Proof of Theorem 3.3.

Proof. The total expected cost of aggregation rounds, incurred in Step 2 of the algorithm, is bounded by

$$(\rho_{ST}+3)\Big[\sum_{i=1}^{K-1}\Big(\sum_{t<i}\frac{\sigma_{i+1}}{\sigma_t}+\sum_{t\geq i}\frac{\delta_t}{\delta_i}\Big)C_i^*+\sum_{i=1}^{K-1}\frac{\sigma_{i+1}}{M}S^*\Big]$$
$$\leq (\rho_{ST}+3)\Big[(2+2)\sum_i C_i^*+2S^*\Big]$$

using Lemma 3.16. Note that for the last inequality, we exploit the fact that σ_i and δ_i are powers of 2 and all sums can be bounded by 2. Hence, by Lemma 3.19, the total cost of access cable installation, incurred in Step 2 and 7, can be bounded by

$$[4(\rho_{ST}+3)+2]\sum_i C_i^*+[2(\rho_{ST}+3)+0.807]S^*+C_U$$

Additionally, since we open a subset of the facilities in F_U, by Lemma 3.20 the sum of cost of installing core cables and opening cost is bounded by

$$\rho_{ST}(S^*+\sum_i C_i^*)+C_U+O_U$$

Altogether, by using Lemma 3.17 we obtain a bound of

$$8*\Big[\big(4(\rho_{ST}+3)+2+(\rho_{ST}+2\rho_{FL})\big)\sum_i C_i^*+\big(3\rho_{ST}+6.807\big)S^*+2\rho_{FL}O^*\Big]$$
$$\leq \big(40\,\rho_{ST}+16\,\rho_{FL}+112\big)\Big[C^*+S^*+O^*\Big]. \tag{3.6}$$

Setting $\rho_{ST} = ln(4)$ due to Byrka et al. (2010), and $\rho_{FL} = 1.488$ due to Li (2013), inequality (3.6) implies that the approximation factor of **AlgBB** is not more than 192. ■

3.4 Conclusions

We introduced two new combined facility location buy-at-bulk network design problems, which naturally arise in the planning of access and distribution networks in telecommunications, logistics, and several other application areas. Combining several algorithmic and analytical techniques, we developed the first constant-factor approximation algorithms for these problems. The existence of constant-factor approximation algorithms

for problems of these types is both theoretical and practical interest. The factors proven in our analysis, however, are still too large to be of practical interest. Improving the current ratios to values relevant for applications, either via new algorithmic concepts or via better analytical tools, still remains an open task.

Furthermore, the algorithms developed in this chapter only apply when each facility has infinite capacity (or say enough capacity to serve all demands). It is an interesting open question whether our algorithmic approach can be extended or modified to obtain approximation algorithms also for the case of capacitated facilities.

Chapter 4

LP-based Approximations and Integrality Gaps

The focus of our work in this chapter is on LP-based techniques. We revisit **BBConFL** and **DDConFL** problems. We also consider the buy-at-bulk facility location problem (**BBFL**). We extend the linear programming framework of Talwar (2002) for the single-source buy-at-bulk problem to both facility location and connected facility location buy-at-bulk problems and prove integrality gap upper bounds for these problems. For the unconnected variant we prove an integrality gap bound of $O(K)$, and for the connected version, we get an improved bound of $O(1)$.

Some results presented in this chapter have been developed in joint work with Zachary Friggstad and Mohammad R. Salavatipour (University of Alberta), and José A. Soto (University of Chile), and will appear in Friggstad et al. (2015).

Previous Work

The buy-at-bulk facility location problem (**BBFL**) was first considered by Meyerson et al. (2000). They show that BBFL can be seen as a special case of the *Cost-Distance* problem, and thereby provide the first randomized approximating algorithm with approximation guarantee $O(\log(|D|))$ for this problem. Their algorithm works for the more general version of non-uniform buy-at-bulk where one has a different set of cable types for each edge. The algorithm was then derandomized by Chekuri et al. (2001), who show that the integrality gap of the cost-distance problem is $O(\log(|D|))$. To the best of our knowledge, this is the only LP-based approximation algorithm that works for the BBFL problem. Later, an $O(K)$ approximation for this problem was developed by Ravi

and Sinha (2006), who extended the combinatorial approximation algorithm of Guha et al. (2009) for the single-source buy-at-bulk problem.

Recall that the unsplittable SSBB (**uSSBB**) problem is a special case of the BBConFL problem. Two LP-based approximation algorithms for uSSBB have been proposed in the literature. Using LP rounding techniques, an $O(K)$ approximation was obtained by Garg et al. (2001). Then, Talwar (2002) showed that an LP formulation of this problem has a constant integrality gap and provided a 216 approximation. The currently best approximation ratio for the uSSBB problem is 40.82 due to Grandoni and Rothvoß (2010); an overview of the results on SSBB can be found in Section 1.2.

Contributions

The focus of our work in this chapter is on LP-based techniques. Namely, we extend the LP-based approximation for **uSSBB** by Talwar (2002) to BBConFL and BBFL, thereby establishing a framework for LP-based approximations for buy-at-bulk variants.

We show that the integrality gap of a natural flow-based formulation for BBConFL can be arbitrarily large. Hence, we focus on the DDConFL problem. Recall that the BBConFL and DDConFL problems are closely related, so that a ρ-approximation algorithm for one problem gives a 2ρ-approximation algorithm for the other. In Section 4.1.1, we present a strong flow-based IP, namely (IP-4-2), model for DDConFL. Our main result is the following.

Theorem 4.1. *The integrality gap of (IP-4-2) is at most 234.*

As a consequence, we get an improved constant factor approximation for DDConFL, beating the 384-approximation we obtained in Chapter 3. We also obtain the first LP-based (deterministic) algorithm for the BBConFL problem whose factor is comparable with the (expected) approximation factor we obtained in Chapter 3. Finally, using similar techniques, we obtain a new LP-based approximation algorithm for the BBFL problem in Section 4.2. We propose a flow-based IP, namely (IP-4-4), formulation for the BBFL problem and obtain the following result.

Theorem 4.2. *The integrality gap of (IP-4-4) is at most $O(K)$.*

This matches the approximation guarantee of the combinatorial algorithm of Ravi and Sinha (2006), improving the LP-based $\mathrm{O}(\log(|D|))$-approximation obtained by Chekuri et al. (2001).

4.1 Buy-at-Bulk Connected Facility Location

In this section we consider the BBConFL and DDConFL problems. We first observe that the natural flow-based integer linear program for BBConFL has unbounded integrality gap. Then, we focus on the DDConFL problem instead and get an improved Integrality gap of $O(1)$ for the underlying LP of the problem, hence obtaining a new LP-based $O(1)$-approximation algorithm for the problem.

4.1.1 IP Formulations

In the following we propose some new IP formulations for the considered problems.

IP modeling of BBConFL

We write a natural flow-based integer linear program for the BBConFL problem. For each edge we create a pair of anti-parallel directed arcs, with same length as the original one. Let $\vec{E}$ be the set of these arcs. We denote to the undirected version of an arc $\bar{e} \in \vec{E}$ by e. For an edge $e = uv$ between nodes u and v, we will use a notation (u, v) or (v, u) to explicitly specify the orientation of the corresponding arc $\bar{e} \in \vec{E}$. We will use the notation e and $\bar{e}$, when it is clear from the context, for the sake of compactness. In our model we will use the following variables: For every $\bar{e} \in \vec{E}$ and client $j \in D$, the variable $f_{\bar{e}}^j$ indicates if flow from client j uses arc $\bar{e}$; for $e \in E$ and access cable k, x_e^k indicates if access cable k is installed on edge e; and z_i indicates if facility i is open or not. We use the notation $\delta^+(S) = \{(u, v) \in \vec{E} \colon u \in S, v \notin S\}$, $\delta^-(S) = \delta^+(V \setminus S)$, $\delta(S) = \{uv \in E \colon u \in S, v \notin S\}$ for each $S \subseteq V$ and $\delta^+(v) = \delta^+(\{v\})$ for each $v \in V$.

We are now ready to provide our integer program.

$$\text{(IP-4-1)} \quad \min \sum_{i \in F} \mu_i y_i + \sum_{e \in E} c_e \sum_{k=1}^{K} \sigma_k x_e^k + M \sum_{e \in E} c_e z_e$$

$$\sum_{\bar{e} \in \delta^+(j)} f_{\bar{e}}^j \geq 1 \qquad \forall j \in D \tag{4.1}$$

$$\sum_{\bar{e} \in \delta^+(v)} f_{\bar{e}}^j = \sum_{\bar{e} \in \delta^-(v)} f_{\bar{e}}^j \qquad \forall j \in D, v \in V \setminus (F \cup \{j\}) \tag{4.2}$$

$$\sum_{\bar{e} \in \delta^-(i)} f_{\bar{e}}^j - \sum_{\bar{e} \in \delta^+(i)} f_{\bar{e}}^j \leq y_i \qquad \forall j \in D, i \in F \tag{4.3}$$

$$\sum_{j \in D} d_j (f_{(u,v)}^j + f_{(v,u)}^j) \leq \sum_{k=1}^{K} u_k x_{uv}^k \qquad \forall\ uv \in E \tag{4.4}$$

$$y_i - \sum_{e \in \delta(S)} z_e \leq 0 \qquad \forall S \subseteq V \setminus \{r\} \colon S \cap F \neq \emptyset, i \in S \quad (4.5)$$

$$y_r = 1 \quad (4.6)$$

$$f_{\bar{e}}^j, y_i, z_e \in \{0,1\}$$

$$x_e^k \quad \text{non-negative integers}$$

Constraints (4.1) impose that at least one unit of flow leaves the clients. Constraints (4.2) are flow conservation constraints at non-facility nodes. Constraints (4.3) state that the flow only terminates at open facilities. Constraints (4.4) ensure that we install sufficient capacity to support the flow. Finally, constraints (4.5) guarantee that all open facilities are connected to the root via core links, where the constraint (4.6) defines the root facility.

It is not hard to show that an optimal fractional solution for the LP relaxation of IP-4-1 only uses the last cable type with the lowest cost per capacity ratio; see Section 2.1. This immediately implies the following result.

Remark 4.3. The integrality gap of (IP-4-1) can be arbitrarily large.

The reason why we failed to get an integrality gap of $O(1)$ for the natural linear programming formulation of BBConFL is due to the fact that we have to purchase capacities in discrete unites, however this is not required in DDConFL. Remember that the differences between the BBConFL and DDConFL problems are just due to their considered access cable cost models. In BBConFL, an access cable of type k has a fixed capacity $u_k \in \mathbb{Z}_{>0}$ and fixed setup cost $\sigma_k \in \mathbb{Z}_{\geq 0}$. In DDConFL, an access cable of type k has a setup cost $\sigma_k \in \mathbb{Z}_{\geq 0}$, a flow dependent cost of $r_k \in \mathbb{Z}_{\geq 0}$, and unbounded capacity.

We hence focus only on the DDConFL problem. Recall that a ρ-approximation algorithm for one problem gives a 2ρ-approximation algorithm for the other; see Section 3.1 for more details.

IP Modeling of DDConFL

We write a flow-based integer linear program for DDConFL. In order to obtain a stronger flow-based formulation, we disaggregate flow variables with respect to cable types. We assume w.l.o.g. that a particular facility r is open and thus it belongs to the core network in the optimal solution and that $D \cap F = \emptyset$. Also, to simplify the description of our algorithm it will be useful to add an artificial *root client* r^* with unit demand, connected to r by an edge of 0 length. For every $\bar{e} \in \vec{E}$, cable type $k \in [K] = \{1, \ldots, K\}$ and client $j \in D$, the variable $f_{\bar{e};k}^j$ indicates if flow from client j uses cable type k on arc $\bar{e}$; for $e \in E$ and $k \in [K]$, x_e^k indicates if cable type k is installed on edge e; z_e indicates if the

core cable is installed on edge e; and y_i indicates if facility i is opened. The opening cost C^{op}, the core installation cost C^{core}, the fixed installation cost C^{fixed} and the routing cost C^{route} of a solution are defined as

$$\begin{aligned} C^{\text{op}} &= \sum_{i\in F} \mu_i y_i; & C^{\text{core}} &= M \sum_{e\in E} c_e z_e; \\ C^{\text{fixed}} &= \sum_{k=1}^{K} C_k^{\text{fixed}}; & C^{\text{route}} &= \sum_{k=1}^{K} C_k^{\text{route}}, \\ \text{where} \quad C_k^{\text{fixed}} &= \sigma_k \sum_{e\in E} c_e x_e^k, \quad \text{and} & C_k^{\text{route}} &= r_k \sum_{j\in D} d_j \sum_{\bar{e}\in\vec{E}} c_e f_{\bar{e};k}^j, \end{aligned} \tag{4.7}$$

represent the fixed cost and routing cost of the cables of type k, respectively. Given a set of cables $I \subseteq [K]$ and a client $j \in D$, we define the *access flow* on $\bar{e} \in \vec{E}$ with respect to I and j as $f_{\bar{e};I}^j = \sum_{k\in I} f_{\bar{e};k}^j$; and the *net in-flow* on a vertex $v \in V$ with respect to I and j, as $g_I^j(v) = \sum_{\bar{e}\in\delta^-(v)} f_{\bar{e};I}^j - \sum_{\bar{e}\in\delta^+(v)} f_{\bar{e};I}^j$. We also use the quantity $h_i^j = \max\{g_{[K]}^j(i), 0\}$ for $j \in D$ and $i \in F$. Formally, this quantity indicates whether facility i is serving client j.

With all the notation above, our integer program formulation is as follows.

$$\text{(IP-4-2)} \quad \min C^{\text{op}} + C^{\text{core}} + C^{\text{fixed}} + C^{\text{route}}$$

$$g_{[K]}^j(j) \le -1 \qquad \forall j \in D \tag{4.8}$$

$$g_{[K]}^j(v) = 0 \qquad \forall j \in D, v \in V \setminus (F \cup \{j\}) \tag{4.9}$$

$$g_{[K]}^j(i) \le h_i^j \qquad \forall j \in D, i \in F \tag{4.10}$$

$$h_i^j \le y_i \qquad \forall j \in D, i \in F \tag{4.11}$$

$$f_{(u,v);k}^j + f_{(v,u);k}^j \le x_{uv}^k \qquad \forall j \in D, k \in [K], uv \in E \tag{4.12}$$

$$\sum_{i\in S\cap F} h_i^j - \sum_{e\in\delta(S)} z_e \le 0 \qquad \forall j \in D, S \subseteq V \setminus \{r\} \colon S \cap F \neq \emptyset \tag{4.13}$$

$$y_r = 1 \tag{4.14}$$

$$g_{[q,K]}^j(v) \le 0 \qquad \forall j \in D, v \in V \setminus F, 1 \le q \le K \tag{4.15}$$

$$g_{[q,K]}^j(i) - \sum_{e\in\delta(i)} z_e \le 0 \qquad \forall j \in D, i \in F \setminus \{r\}, 1 \le q \le K \tag{4.16}$$

$$x_e^k, f_{\bar{e};k}^j, y_i, z_e, h_i^j \in \{0,1\} \tag{4.17}$$

Constraints (4.8)–(4.12) are flow conservation and capacity constraints similar to the ones for the IP-4-1 model. Constraints (4.13) are saying that if i is the facility serving demand j (that is, the only i for which $h_i^j = 1$) then for each set S containing i and not containing the root there is a core link connecting S with its complement. In other words, all open facilities are connected to the root via core links, where the constraint (4.14) defines the root facility. Constraints (4.15) and (4.16), called *path monotonicity* constraints, strengthen the linear relaxation of (IP-4-2) – these constraints ensure that

the cable types along any path used to connect clients to facilities are nondecreasing from each client to its facility. The validity of these constraints follows from the fact that (due to economy of scales) the flow (in the optimal fractional solution) never splits. More precisely, if the flow from a subset of clients shares an edge then it shares the subsequent edges to an open facility; see the paper by Garg et al. (2001) for more details.

We remark that the interesting variables of this IP formulation are

$$(f, x, y, z) = ((f^j_{\bar{e};k}), (x^k_e), (y_i), (z_e)).$$

All the other quantities are written in terms of these variables.

The introduction of variables h^j_i may seem artificial, however, in the following section we show that they are needed in order to achieve a constant integrality gap IP.

A naive Model for DDConFL

In this section, we show that an alternative, but perhaps more natural IP formulation for DDConFL has unbounded integrality gap. Consider the following integer programming formulation:

(IP-4-3) $\min\ C^{\text{op}} + C^{\text{core}} + C^{\text{fixed}} + C^{\text{route}}$

$$g^j_{[K]}(j) \le -1 \qquad \forall j \in D \tag{4.8}$$

$$g^j_{[K]}(v) = 0 \qquad \forall j \in D, v \in V \setminus (F \cup \{j\}) \tag{4.9}$$

$$g^j_{[q,K]}(v) \le 0 \qquad \forall j \in D, v \in V \setminus F, 1 \le q \le K \tag{4.15}$$

$$g^j_{[q,K]}(i) - \sum_{e \in \delta(i)} z_e \le 0 \qquad \forall j \in D, i \in F \setminus \{r\}, 1 \le q \le K \tag{4.16}$$

$$f^j_{(u,v);k} + f^j_{(v,u);k} \le x^k_{uv} \qquad \forall j \in D, k \in [K], uv \in E \tag{4.12}$$

$$y_r = 1 \tag{4.14}$$

$$g^j_{[K]}(i) \le y_i \qquad \forall j \in D, i \in F \tag{4.18}$$

$$y_i - \sum_{e \in \delta(S)} z_e \le 0 \qquad \forall S \subseteq V \setminus \{r\} \colon S \cap F \neq \emptyset, i \in S \tag{4.19}$$

$$x^k_e, f^j_{\bar{e};k}, y_i, z_e \in \{0, 1\}$$

The difference between this formulation and (IP-4-2) is that (IP-4-3) does not have variables h. We replace constraints (4.10)-(4.11) and (4.13) by (4.18) and (4.19) respectively.

Theorem 4.4. *The integrality gap of (IP-4-3) can be arbitrarily large.*

Proof. Consider the instance described in Figure 4.1, where the square nodes represent facilities and the circle nodes represent clients. In this instance, $K = 1$, i.e. we have a unique access cable type, and we set $\sigma_1 = r_1 = 1$. The core cable has a cost (per unit length) equal to M with $1 < M < q$. For every facility $i \in \{1, \ldots, p\}$, we set an opening cost of $\mu_i = 1$. We also set $\mu_n = \infty$. The root facility r, which must be opened, has an opening cost of 0. The distances are given by the metric completion of the edge costs depicted in the figure, where $L >> 1$ is a constant larger than any other finite parameter of the instance.

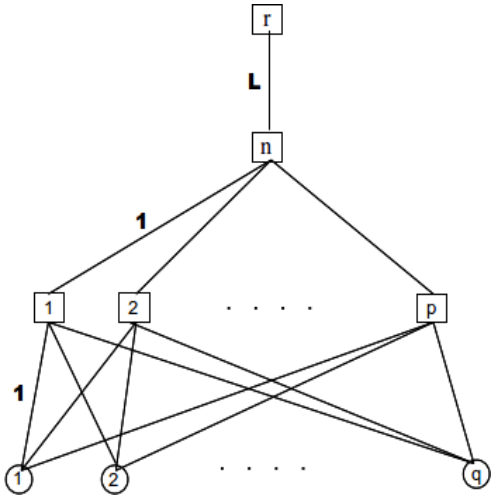


Figure 4.1: An instance of DDConFL with q clients of unit demands and (p+2) potential facilities with facility r as the root node.

The optimal integral solution to this instance can connect all the clients to a fixed facility $i^* \in \{1, \ldots, p\}$[1] via access links. Then this open facility is connected to the root node via (unopened) facility n using core links.

However, the LP relaxation of (IP-4-3) can cheat and open all facilities $i \in \{1, \ldots, p\}$ to the extends of $1/p$ to serve clients demands. Then it can install core links to the extends of $1/p$ on the edges connecting them (via node n) to the root node. This means that LP only pays $M \cdot L/p$ for the core link along edge nr, while the integral solution pays a cost of $M \cdot L$ on that for the same edge. Since L was chosen as an arbitrarily large constant, this is the only relevant value to compare. Hence, the integrality gap is proportional to p and thus it can be made arbitrarily large. ■

[1]Note that they are (almost) collocated.

4.1.2 LP-Based Approximation Solution

We extend the linear programming framework of Talwar (2002) for the single-source buy-at-bulk problem to these variants and prove integrality gap upper bound $O(1)$ for the problem. Let (LP-4-2) be a linear program obtained from (IP-4-2) by replacing the integrality requirements of the variables to just containment in the interval $[0, 1]$, and let (f, x, y, z) be the optimal (possibly) fractional solution to (LP-4-2). It is not hard to show that (LP-4-2) can be solved in polynomial time using, for example, the ellipsoid method. We shall show that this fractional solution can be rounded to an integer solution increasing the total cost by a constant factor. Let gap_{ST} denote the upper bound on the integrality gap of the cut based formulation of the Steiner tree problem, which is 2 (see Agrawal et al., 1995).

4.1.2.1 Rounding Algorithm

Our algorithm has the following four phases:

Preprocessing Phase:

Pruning: We prune the set of access cable types such that all cables are considerably different. Similar to Theorem 3.6, this can be done without increasing the cost of the optimal solution too much.

Theorem 4.5. *Given $\epsilon_1, \epsilon_2 \in (0, 1)$, we can prune the set of access cables such that, for any i, we have $\sigma_{i+1} > \sigma_i/\epsilon_1$ and $r_{i+1} < \epsilon_2 \cdot r_i$ hold and the installation and routing costs of the optimal (fractional) solution increase by a factor of at most $1/\epsilon_1$ and $1/\epsilon_2$, respectively.*

For the sake of notation, let $[K]$ be the set of cables left and let (f, x, y, z) be the new solution of (LP-4-2) after the pruning stage. Furthermore, for each client $j \in D$ and positive radius R, define $B(j, R) = \{v \in V : c_{jv} \leq R\}$ [2] to be the *moat* centered at j. We say that two moats $B_1 = B(j_1, R_1)$ and $B_2 = B(j_2, R_2)$ *overlap* if $c_{j_1 j_2} \leq R_1 + R_2$.

Define also $L_k^j = \sum_{\vec{e} \in \vec{E}} f_{\vec{e};k}^j c_e$ which represents the estimated distance that the flow of client j travels on cables of type k. Note that $C_k^{\text{route}} = r_k \sum_{j \in D} d_j L_k^j$.

[2] Here we are assuming that G is complete, otherwise we use G's metric completion.

Flow path decomposition: Every client j sends (at least) one unit of flow from itself to open facilities, specified by the $f^j_{e,[K]}$ variables. We decompose this fractional flow into a set of paths P_j, with path $p \in P_j$ starting from j and ending at some facility. Let $\phi(p)$ denote the amount of flow of path p.

Filtering: For a predefined constant $\theta \in (0,1)$ and for all $j \in D$, choose a subset of paths $\bar{P}_j \subseteq P_j$ such that $\phi_j := \sum_{p\in\bar{P}_j} \phi(p) \geq \theta$, by selecting paths in increasing order of their lengths until their total $\phi(p)$-value is at least θ. For each $j \in D$, let β_j be the length of the longest path in $\bar{P}_j$. Define a new solution $(\bar{f}, \bar{x}, \bar{y}, \bar{z})$ as follows. For each client $j \in D$, scale the amount of flow sent across each $P \in \bar{P}_j$ by $1/\phi_j$ and set the flow sent across each $P \in P_j - \bar{P}_j$ to 0. The new flow $\bar{f}$ is derived naturally from this new path decomposition. For each cable $k \in [K]$ and edge $e \in E$, define $\bar{x}^k_e$ as x^k_e/θ if there exists some j with $\bar{f}^j_{\bar{e};k} > 0$, where $\bar{e} \in \vec{E}$ is one of the two arcs associated to e; and 0 otherwise. For each i, set $\bar{y}_i = \min\{y_i/\theta, 1\}$. And finally for each $e \in E$, set $\bar{z}_e = \min\{z_e/\theta, 1\}$. It is easy to show that this solution is feasible for (LP-4-2). The basic idea of this part of the algorithm is inspired by Shmoys et al. (1997).

There are two important properties to notice. The first one is that the solution $(\bar{f}, \bar{x}, \bar{y}, \bar{z})$ is such that the entire demand of client j is satisfied by open facilities on the moat $B(j, \beta_j)$. The second property is the following bound which is useful for the analysis. Let $\widetilde{P}_j \subseteq P_j$ be the set of paths with lengths at least β_j. Then, $\widetilde{P}_j$ includes all paths in $P_j \setminus \bar{P}_j$ and at least one path, say p^* (the longest) of $\bar{P}_j$. We conclude that $\sum_{p\in\widetilde{P}_j} \phi(p) \geq \sum_{p\in P_j} \phi(p) - \sum_{p\in\bar{P}_j\setminus\{p^*\}} \phi(p) \geq 1-\theta$, and so

$$\sum_{k=1}^{K} L^j_k = \sum_{p\in P_j} \sum_{e\in p} \phi(p) c_{\bar{e}} \geq \sum_{p\in\widetilde{P}_j} \phi(p) \sum_{e\in p} c_{\bar{e}} \geq \beta_j(1-\theta). \tag{4.20}$$

Facility Selection Phase:

Moat selection: For a predefined constant $\eta > 1$, we consider the set of moats $\mathcal{B}_\eta = \{B(j, \eta\beta_j) \colon j \in D\}$ around clients. We choose a maximal set $\mathcal{B}' \subseteq \mathcal{B}_\eta$ of moats which do not overlap. We do this by processing the moats in $\mathcal{B}_\eta$ in increasing order of their radii, and greedily adding them to $\mathcal{B}'$ so that for each pair of selected moats in $\mathcal{B}'$ with centers $j, j' \in D$, $B(j, \eta\beta_j)$ and $B(j', \eta\beta_{j'})$ do not overlap. Let S_{core} be the set of clients with moats in $\mathcal{B}'$. Observe that for the artificial root client r^*, we have $\beta_{r^*} = 0$ and so $r^* \in S_{\text{core}}$.

Facility opening: For each $j \in S_{\text{core}}$, let $F_j = \{i \colon \bar{h}_i^j > 0\}$ be the set of facilities which fractionally serve demand from j with respect to solution $(\bar{f}, \bar{x}, \bar{y}, \bar{z})$. By the first property mentioned at the end of the preprocessing phase, $F_j \subseteq B(j, \eta\beta_j)$, hence the family $\{F_j : j \in S_{\text{core}}\}$ consist of disjoint sets. On each F_j we open the facility i_j^* with lowest opening cost. In particular, the root r is opened since $F_{r^*} = \{r\}$. Let I be the set of facilities opened on this stage.

For the purpose of analysis, associate each client with a special facility denoted as its $(K+1)$-st *proxy*. Formally, for each $j \in S_{\text{core}}$ we set $\text{proxy}_{K+1}(j) = i_j^*$. For the remaining clients $j \in D \setminus S_{\text{core}}$, we set $\text{proxy}_{K+1}(j) = \text{proxy}_{K+1}(j')$, where $j' \in S_{\text{core}}$ is the center of the smallest moat in $\mathcal{B}'$ that overlapped with $B(j, \eta\beta_j)$. Since the moats in $\mathcal{B}'$ were added in increasing radii and (4.20), we get

$$c(j, \text{proxy}_{K+1}(j)) \leq (1+2\eta)\beta_j \leq \frac{(1+2\eta)}{(1-\theta)} \sum_{q=1}^{K} L_q^j \qquad \forall j \in D. \tag{4.21}$$

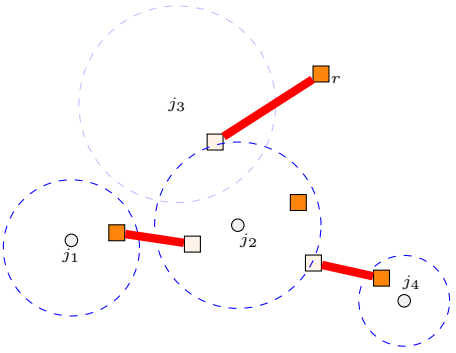
Core Network Phase:

Consider the graph G^{K+1} obtained from G by contracting the nodes of each F_j into single nodes, for $j \in S_{\text{core}}$. We construct an approximately optimal Steiner tree T' in G^{K+1} having the contracted nodes as terminals. For this task, we find an approximate Steiner tree whose cost is within a factor gap_{ST} of the cut-based relaxation. In the uncontracted graph G, the edges of T' form a forest which touches a subset of the facilities in F_j, called $\bar{F}_j$ which may not include the open facility i_j^*; see Figure 4.2a. In order to connect all the open facilities together, we augment T' with all the stars $Q_j = \{ji \colon i \in \bar{F}_j \cup \{i_j^*\}\}$, for $j \in S_{\text{core}}$.

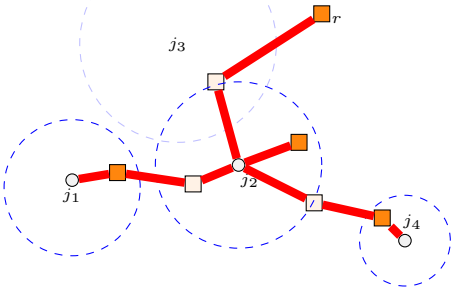
Let T^{core} be the resulting tree, after possibly canceling some cycles; see Figure 4.2b. This is the core Steiner tree our algorithm will output. To conclude this stage, we install core cables on T^{core}.

Access Network Phase:

We construct the access network in a top-down manner, installing cables progressively in stages, which we number from $i = K$ to 1. To start this process, let T_{K+1} be a *minimum spanning tree* on the graph induced by the set I of open facilities, and connect them using an artificial cable type $K+1$. This tree won't appear in the end, as it will be replaced by the core network. In stage i, we augment the current tree T_{i+1}, which uses only cables of type $i+1$ or higher, by installing cables of type i. Define $\bar{L}_k^j$ to be



(a)The tree T'



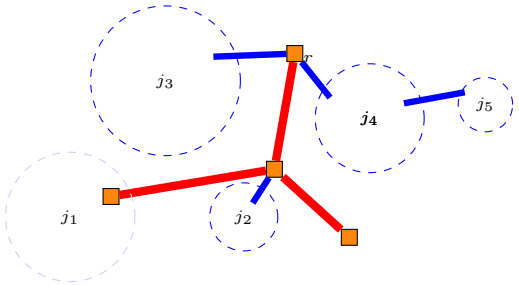
(b)The tree T^{core}

Figure 4.2: An illustration of the core network installation phase on a simple instance where clients set $\{j_1, j_2, j_4\}$ has been chosen as S_{core}.

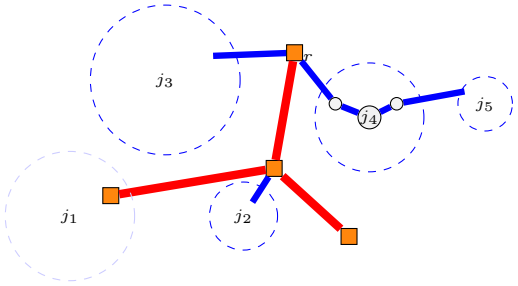
$\sum_{\bar{e}\in\vec{E}} \bar{f}^j_{\bar{e};k} \cdot c_e$. This estimates the distance that flow from j goes on cable type k. Let $\bar{R}^j_l = \sum_{k=1}^{l-1} \bar{L}^j_k$ be the estimated distance beyond that flow from j uses cable type l or higher in the new fractional solution. Intuitively, $\bar{R}^j_l$ tells us how far from j to go before the LP solution installs access cable types l or higher. Stage i consists of the following two steps:

Step 1. Moat Selection: For predefined $\gamma > \zeta > 1$, we construct the set of moats $\mathcal{B}^i_\gamma = \{B(j, \gamma\bar{R}^j_i) : j \in D\}$ around all clients. We define $\hat{S}_i$ to be the set of clients whose moats intersect T_{i+1}. For each $j \in \hat{S}_i$ remove moat $B(j, \gamma\bar{R}^j_i)$ from $\mathcal{B}^i_\gamma$. Similar to what we did for the core network, we choose a maximal set $\mathcal{B}^i \subseteq \mathcal{B}^i_\gamma$ of moats which do not overlap by selecting moats from $\mathcal{B}^i_\gamma$ in increasing order of their radii. Let S_i be the set of clients whose moats are selected in round i.

Step 2. Cable type i installation: We construct the set $\mathcal{B}^i_\zeta = \{B(j, \zeta\bar{R}^j_i) : j \in S_i\}$ of moats around clients in S_i. We obtain a graph G^i from G by contracting each moat in $\mathcal{B}^i_\zeta$ into a super-node, and the current tree T_{i+1} into a super-node called r_{i+1}. We then construct an approximately optimal Steiner tree in G^i, again with integrality gap bound gap_{ST}, where the terminals are all the super-nodes. By uncontracting, we get a forest in G that touches at least one node in T_{i+1} and one node from each moat (see Figure 4.3a). To get a tree, called $\bar{T}_i$, from the resulting forest, we add direct edges from each



(a)A forest connecting chosen moats to T_{K+1}



(b)The resulting tree $\bar{T}_i$ after adding the described edges to the forest

Figure 4.3: An illustration of the access network phase where $i = K$.

client $j \in S_i$ to each node of $B(j, \zeta \bar{R}_i^j)$ that is incident on the forest[3] and then we cancel cycles (see Figure 4.3b).

Using technique by Khuller et al. (1995), we then convert tree $\bar{T}_i$ rooted at r_{i+1}, into an (α, β)-Light Approximate Shortest-path Tree (LAST), for parameters $\beta = \frac{\alpha+1}{\alpha-1}$ and $\alpha > 1$ to be chosen later. Let LAST_i be the resulting tree. The LAST algorithm (see Khuller et al., 1995) transforms tree $\bar{T}_i$ into LAST_i with $c(\text{LAST}_i) \leq \beta c(\bar{T}_i)$ such that the path length of any vertex v to root r_{i+1} in LAST_i is at most α times the length of a shortest v-r_{i+1} path in G^i. We un-contract the moats and install cables of type i on the edges of LAST_i. Let $T_i = T_{i+1} \cup \text{LAST}_i$.

For the purpose of analysis, for each $j \in S_i$, we call an arbitrary node in its moat which is connected to LAST_i as *the proxy*, denoted by $\text{proxy}_i(j)$. For the clients $j \in \hat{S}_i$, we define $\text{proxy}_i(j)$ to be an arbitrary node in $B(j, \gamma \bar{R}_i^j) \cap T_{i+1}$. For the remaining clients $j' \in D \setminus S_i \cup \hat{S}_i$, we define $\text{proxy}_i(j')$ to be $\text{proxy}_i(j)$, where $j \in S_i$ is the center of the smallest moat in $\mathcal{B}^i$ that overlapped with $B(j', \gamma \bar{R}_i^{j'})$. It is easy to verify that $c(j, \text{proxy}_i(j)) \leq 3\gamma \bar{R}_i^j \leq \frac{3\gamma}{\theta} \sum_{k=1}^{i-1} L_k^j$. If we set $\Delta = \max\{\frac{1+2\eta}{1-\theta}, \frac{3\gamma}{\theta}\}$, then by the previous inequality and (4.21), we get

$$c(j, \text{proxy}_{i+1}(j)) \leq \Delta \cdot \sum_{q=1}^{i} L_q^j \qquad \forall j \in D, 1 \leq i \leq K \tag{4.22}$$

[3]It is worth pointing out that this crucial step of adding direct edges is missing from the **uSSBB**-approximation by Talwar (2002), even though it seems necessary for both that algorithm and ours to work.

which will be useful in bounding the routing cost.

Finally, note that $R_1^j = 0$ for all j. This means that in the first step of the last stage, S_1 consists of all clients that have not been connected to the current tree. Therefore, at the end of the last stage, T_1 is a tree spanning all clients and open facilities. The access network we return consists of the forest obtained by removing the artificial tree T_{K+1} from T_1.

4.1.2.2 Analysis

Let $C^{*\text{op}}$, $C^{*\text{core}}$, $C^{*\text{fixed}}$ and $C^{*\text{route}}$ be the opening cost, core installation cost, fixed installation cost and routing cost paid by the LP optimum (see (4.7)). And let C^{op}, C^{core}, C^{fixed} and C^{route} the ones paid by our algorithm. Let OPT be the cost of LP optimum. The following lemma bounds the opening cost.

Lemma 4.6. *The opening cost of the returned solution is at most* $\frac{1}{\theta}C^{*op}$.

Proof. The cost of facility i_j^* can be bounded by using (4.8),(4.10),(4.11), and the fact that i_j^* was chosen as the cheapest facility of $F_j \subseteq B(j, \beta_j)$, as follows

$$\mu_{i_j^*} = \mu_{i_j^*} \cdot \frac{1}{\phi_j} \sum_{p \in \bar{P}_j} \phi(p) = \frac{1}{\phi_j} \sum_{i \in F_j} \mu_{i_j^*} h_i^j \leq \frac{1}{\theta} \sum_{i \in F_j} \mu_i y_i.$$

As sets $F_j, j \in S_{\text{core}}$ are disjoint, the total opening cost is at most $\frac{1}{\theta} \cdot \sum_{i \in F} y_i \mu_i = \frac{1}{\theta} C^{*\text{op}}$. ■

Lemma 4.7. *The cost of core link installation is at most* $\frac{\eta+1}{\theta(\eta-1)} \cdot gap_{ST} \cdot C^{*core}$.

Proof. By (4.13), one can verify that $\sum_{\bar{e} \in \delta^+(S)} \bar{z}_{\bar{e}} \geq 1$ holds for any arbitrary set $S \subset V$ that contains all facilities in F_j (for some j) and it does not contain r. This means that $\bar{z}$ is a feasible fractional solution to the cut based LP relaxation of the Steiner tree problem on the graph G^{K+1} (see the core network phase) whose terminals are all the contracted sets F_j (recall that $F_{r^*} = \{r\}$). In particular, the Steiner tree T' found in the core network phase has cost at most $\text{gap}_{\text{ST}} \cdot \sum_{e \in E} c_e \bar{z}_e$. The cost of the extra edges included in the final tree T^{core} (i.e., the union of all stars Q_j) can be charged to the cost of T' as follows.

For each facility b_j in $\bar{F}_j$ let $e(b_j) = b_j v \in T'$ be any edge incident to it. Since b_j is in $B(j, \beta_j)$ and v is outside $B(j, \eta\beta_j)$, we conclude that the cost of $e(b_j)$ is at least $(\eta - 1)\beta_j$. By a similar argument, if $e = e(b_j) = e(b_k)$ where $b_j \in \bar{F}_j$ and $b_k \in \bar{F}_k$, then we can use

the fact that $B(j, \eta\beta_j)$ and $B(k, \eta\beta_k)$ do not overlap to conclude that the length of e is at least $(\eta - 1)(\beta_j + \beta_k)$. Therefore, the total cost of the union of all Q_j is at most

$$\sum_{j \in S_{\text{core}}} \left(c(ji_j^*) + \sum_{b \in \bar{F}_j} c(jb) \right) \leq 2 \sum_{j \in S_{\text{core}}} \sum_{b \in \bar{F}_j} \beta_j \leq 2 \sum_{e \in T'} \frac{c(e)}{\eta - 1}.$$

Summing up, the cost of T^{core} is at most $1 + \frac{2}{(\eta-1)}$ times the cost of T', and therefore it is at most $\frac{\eta+1}{\theta(\eta-1)} \cdot \text{gap}_{\text{ST}} \cdot \sum_{e \in E} c_e z_e$. ■

In the next two lemmas, we bound the fixed cost and incremental cost of the cables installed at stage i of the last phase of the algorithm, denoted by C_i^{fixed} and C_i^{route}, respectively.

Lemma 4.8. $C_i^{fixed} \leq \sigma_i \cdot gap_{ST} \cdot \dfrac{\gamma\beta\zeta}{(\gamma - \zeta)(\zeta - 1)\theta} \left(\sum_{q=i}^{K} \dfrac{1}{\sigma_q} C_q^{*fixed} + \dfrac{1}{M} C^{*core} \right)$

Proof. Let S be an arbitrary subset of $V \setminus \{r\}$ that contains $B(j, \zeta \bar{R}_i^j)$. We first show that $\sum_{q=1}^{i-1} \bar{b}_{q;S}^j \leq \frac{1}{\zeta}$, where $\bar{b}_{q;S}^j := \sum_{\bar{e} \in \delta^+(S)} \bar{f}_{\bar{e},q}^j$ indicates the amount of flow from j crossing the boundary of S thorough cables of type q. The flow we are considering has to travel from j to the boundary of S using only use cables of type q or thinner. So, as $\bar{b}_{q;S}^j$ travels a distance of at least $\zeta \bar{R}_i^j$, it contributes at least $\bar{b}_{q;S}^j \zeta \bar{R}_i^j$ units to $\bar{R}_i^j = \sum_{k=1}^{i-1} \bar{L}_k^j$. As the contributions from each q are disjoint, we have $\bar{R}_i^j \geq \sum_{q=1}^{i-1} \bar{b}_{q;S}^j \zeta \bar{R}_i^j$, which implies that $\sum_{q=1}^{i-1} \bar{b}_{q;S}^j \leq \frac{1}{\zeta}$. This together with the LP constraints guarantee that $\sum_{q=i}^{K} \bar{b}_{q;S}^j + \sum_{e \in \delta^+(S)} \bar{z}_e \geq 1 - \frac{1}{\zeta}$ and hence $\sum_{e \in \delta^+(S)} \left(\sum_{q=i}^{K} \bar{x}_e^q + \bar{z}_e \right) \geq 1 - \frac{1}{\zeta}$. This means that the vector $\bar{z} + \sum_{q=i}^{K} \bar{x}^q$, scaled by a factor $\frac{\zeta}{\zeta - 1}$, is a feasible fractional solution to the LP relaxation of the Steiner tree connecting balls $B(j, \zeta \bar{R}_i^j)$ to T_{i+1}. Therefore, the cost of the Steiner tree computed in step 2 of the access network phase can be bounded by

$$\frac{\text{gap}_{\text{ST}} \zeta}{\zeta - 1} \left(\sum_{e \in E} \sum_{q=i}^{K} c_e \bar{x}_e^q + \sum_{e \in E} c_e \bar{z}_e \right) \leq \frac{\text{gap}_{\text{ST}} \zeta}{(\zeta - 1)\theta} \left(\sum_{q=i}^{K} \frac{1}{\sigma_q} C_q^{*\text{fixed}} + \frac{1}{M} C^{*\text{core}} \right).$$

Similar to Lemma 4.7, one can show that the cost of extra edges of $\bar{T}_i$, added after uncontracting the moats, is at most $\frac{\zeta}{\gamma - \zeta}$ times the cost of the current forest. Altogether, the cost of the LAST_i tree is at most

$$c(\text{LAST}_i) \leq \text{gap}_{\text{ST}} \cdot \frac{\gamma}{\gamma - \zeta} \cdot \frac{\beta\zeta}{(\zeta - 1)\theta} \left(\sum_{q=i}^{K} \frac{1}{\sigma_q} C_q^{*\text{fixed}} + \frac{1}{M} C^{*\text{core}} \right). \quad ■$$

Recall that $\Delta = \max\{\frac{1+2\eta}{1-\theta}, \frac{3\gamma}{\theta}\}$. The incremental cost of the cables installed at stage i can be bounded as follow.

Lemma 4.9. $C_i^{route} \leq \Delta r_i \alpha \sum_{q=1}^{i} (1+\alpha)^{i-q} \frac{1}{r_q} C_q^{*route}$

Proof. Let $T = \bigcup_{i=1}^{K} \text{LAST}_i$ be the access network (forest) constructed by the algorithm and let V_i be the set of nodes at which flow, routed toward facilities via T, reaches the LAST_i tree, where we assume $V_1 = D$. Also, let $d_T(u, v)$ be the distance between u and v on T. The proof of the lemma is by induction on i. Since we install cables of type 1 on the LAST_1 tree, and also $\text{proxy}_2(j)$ lies in T_2, we have $d_T(j, T_2) \leq \alpha c(j, \text{proxy}_2(j))$. Hence by using (4.22), we get

$$C_1^{\text{route}} = r_1 \sum_{j \in D} d_j \cdot d_T(j, T_2) \leq \Delta r_1 \alpha \sum_{j \in D} d_j L_1^j \leq \Delta r_1 \alpha \frac{1}{r_1} C_1^{*\text{route}},$$

which concludes the case $i = 1$. Assume now, that the claim holds for all $l < i$. Then the total cost of routing along cables of type i, C_i^{route}, can be bounded by

$$r_i \sum_{v \in V_i} D_v \cdot d_T(v, T_{i+1}) \leq r_i \alpha \left[\sum_{p=1}^{i-1} \frac{C_p^{\text{route}}}{r_p} + \sum_{j \in D} d_j \Delta \sum_{q=1}^{i} L_q^j \right],$$

where D_v is the amount of demand routed through LAST_i from node v. The above inequality holds because the cost of of routing the demand from v to T_{i+1} on T can be bounded by the cost of routing the demand back via forest defined by the edges with cables of types $1, \ldots, i-1$ in T to its source (sources), say j, and then from there to $\text{proxy}_{i+1}(j)$, which is lied in T_{i+1}, using the triangle inequality and (4.22). Note that this is true because we install cables on the LAST_i trees. Therefore by induction hypothesis we get

$$\begin{aligned}
C_i^{\text{route}} &\leq \Delta r_i \alpha \Big[\sum_{p=1}^{i-1} \alpha \sum_{q=1}^{p} (1+\alpha)^{p-q} \frac{C_q^{*\text{route}}}{r_q} + \sum_{q=1}^{i} \frac{C_q^{*\text{route}}}{r_q} \Big] \\
&= \Delta r_i \alpha \Big[\sum_{q=1}^{i-1} \frac{C_q^{*\text{route}}}{r_q} \big[\alpha \sum_{p=q}^{i-1} (1+\alpha)^{p-q} \big] + \sum_{q=1}^{i-1} \frac{C_q^{*\text{route}}}{r_q} + \frac{C_i^{*\text{route}}}{r_i} \Big] \\
&= \Delta r_i \alpha \Big[\sum_{q=1}^{i-1} \frac{C_q^{*\text{route}}}{r_q} (1+\alpha)^{i-q} + \frac{C_i^{*\text{route}}}{r_i} \Big] \\
&= \Delta r_i \alpha \sum_{q=1}^{i} \frac{C_q^{*\text{route}}}{r_q} (1+\alpha)^{i-q}.
\end{aligned}$$

■

By Lemma 4.8, Theorem 4.5, and by summing over all cable types, the fixed cost paid by the algorithm can be bounded as follows.

$$\begin{aligned} C^{\text{fixed}} &\leq \text{gap}_{\text{ST}} \cdot \frac{\gamma\beta\zeta}{(\gamma-\zeta)(\zeta-1)\theta}\Big[\sum_{s=1}^{K} C_s^{*\text{fixed}}\big(\sum_{i\leq s}\frac{\sigma_i}{\sigma_s}\big) + C^{*\text{core}}\big(\sum_{i=1}^{K}\frac{\sigma_i}{M}\big)\Big] \\ &\leq \text{gap}_{\text{ST}} \cdot \frac{\gamma\beta\zeta}{(\gamma-\zeta)(\zeta-1)\theta(1-\epsilon_1)}\big[C^{*\text{fixed}} + C^{*\text{core}}\big]. \end{aligned} \tag{4.23}$$

Similarly, by using Lemma (4.9), we bound the routing cost as follows.

$$\begin{aligned} C^{\text{route}} &\leq \Delta\alpha \sum_{i=1}^{K}\sum_{s=1}^{i}(1+\alpha)^{i-s}\frac{r_i}{r_s}C_s^{*\text{route}} \\ &\leq \Delta\alpha \sum_{i=1}^{K}\sum_{s=1}^{i}\big((1+\alpha)\cdot\epsilon_2\big)^{i-s}C_s^{*\text{route}} \\ &\leq \Delta\alpha \sum_{s=1}^{K} C_s^{*\text{route}}\sum_{i\geq s}\big((1+\alpha)\cdot\epsilon_2\big)^{i-s} \\ &\leq \frac{\Delta\alpha}{1-\epsilon_2(1+\alpha)}\cdot C^{*\text{route}}. \end{aligned} \tag{4.24}$$

Proof of Theorem 4.1. We just need to show that the cost of the solution obtained by our rounding algorithm is within a constant factor of the optimal fractional solution cost.

Using (4.23), (4.24), Lemmas 4.6 and 4.7, the total cost of our solution is at most

$$\frac{1}{\theta}C^{*\text{op}} + \frac{(\eta+1)\text{gap}_{\text{ST}}}{\theta(\eta-1)}C^{*\text{core}} + \frac{\gamma\beta\zeta\cdot\text{gap}_{\text{ST}}\,(C^{*\text{fixed}}+C^{*\text{core}})}{(\gamma-\zeta)(\zeta-1)\theta(1-\epsilon_1)} + \frac{\Delta\alpha}{1-\epsilon_2(1+\alpha)}C^{*\text{route}}.$$

Finally, using Theorem 4.5, we can bound the cost of our solution by

$$\leq \max\left(\frac{1}{\epsilon_1}\cdot\frac{\gamma\beta\zeta\cdot\text{gap}_{\text{ST}}}{(\gamma-\zeta)(\zeta-1)\theta(1-\epsilon_1)} + \frac{(\eta+1)\text{gap}_{\text{ST}}}{\theta(\eta-1)},\ \frac{1}{\epsilon_2}\cdot\frac{\Delta\alpha}{1-\epsilon_2(1+\alpha)}\right)OPT. \tag{4.25}$$

This completes the proof of Theorem 4.1. ■

Setting $\alpha = 1.47$, $\gamma = 4.10$, $\epsilon_1 = 0.50$, $\epsilon_2 = 0.20$, $\theta = 0.78$, $\eta = 1.27$ and $\zeta = 2$ and recalling $\text{gap}_{\text{ST}} = 2$, inequality (4.25) implies that the integrality gap of (IP-4-2) is no more than 234. Thus, we obtain the first LP-based (deterministic) algorithm for the DDConFL problem and thereby for the BBConFL problem.

4.2 Buy-at-Bulk Facility Location

Recall that if we omit the requirement to connect the open facilities, the BBConFL becomes the BBFL problem. In this section we study the integrality gap of an LP formulation for the problem. As with the BBConFL problem, we consider a variant of the BBFL problem, called DDFL, in which we replace the capacitated access cables by discount cable types. Recall that similar to the relation between BBConFL and DDConFL, one can transform between BBFL and DDFL with factor 2 loss.

IP modeling of DDFL

Similar to Section 4.1.1, DDFL can be formulated as follows:

$$\text{(IP-4-4)} \quad \min C^{\text{op}} + C^{\text{fixed}} + C^{\text{route}}$$

$$g^j_{[K]}(j) \leq -1 \qquad \forall j \in D \tag{4.8}$$

$$g^j_{[K]}(v) = 0 \qquad \forall j \in D, v \in V \setminus (F \cup \{j\}) \tag{4.9}$$

$$g^j_{[q,K]}(v) \leq 0 \qquad \forall j \in D, v \in V \setminus F, 1 \leq q \leq K \tag{4.15}$$

$$f^j_{(u,v);k} + f^j_{(v,u);k} \leq x^k_{uv} \qquad \forall j \in D, k \in [K], uv \in E \tag{4.12}$$

$$g^j_{[K]}(i) \leq y_i \qquad \forall j \in D, i \in F \tag{4.26}$$

$$g^j_{[q,K]}(i) - y_i \leq 0 \qquad \forall j \in D, i \in F, 1 \leq q \leq K \tag{4.27}$$

$$x^k_e, f^j_{e;k}, y_i \in \{0,1\} \tag{4.28}$$

In this formulation we do not need the z and h^j_i variables anymore, as they were used to model connectivity among facilities. Constraints (4.26) state that the flow only ends at open facilities, and Constraints (4.15) and (4.27) force the path monotonicity discussed in Section 4.1.1.

4.2.1 Proof of Theorem 4.2

Let (f, x, y) be the optimal solution to the LP relaxation of (IP-4-2). We shall show that this fractional solution can be rounded to an integer solution increasing the total cost by a constant factor.

4.2.1.1 Rounding Algorithm

Our rounding algorithm will follow the same general ideas of that for DDConFL. But we replace the last two phases (core network and access network phase) by a single one denoted network phase. Another key difference is that we may open facilities at any

stage of the network phase. Ultimately, this is why our integrality gap bound is $O(K)$ as we have to overestimate and bound the opening cost in each of the K stages by the total opening cost paid by the LP.

Preprocessing Phase. Apply the preprocessing phase (pruning, flow path decomposition and filtering) of Section 4.1.2.1, disregarding variables z. Let $(\bar{f}, \bar{x}, \bar{y})$ be the solution after this phase.

Initial Facility Selection Phase. Perform the same instructions as in the facility selection phase of Section 4.1.2.1 but fixing $\eta = 1$. Let I' be the set of facilities opened in this phase.

Network Phase. We construct a solution in a top-down manner, installing cables *and* and possibly opening more facilities in stages, which we number from $i = K$ to 1. We start with solution $(I_{K+1}, T_{K+1}) = (I', \emptyset)$. At stage i, with $1 \leq i \leq K$, we augment the current solution by (1) opening some extra facilities and (2) installing cables of type i. We will do this while keeping the invariant that T_i is a forest in G such that each connected component (tree) contains an open facility of I_i.

Stage i is similar to the i-th stage of the access network phase in Section 4.1.2.1.

1. For a predefined constant $\gamma > \zeta > 1$, construct the set of moats $B(j, \gamma \bar{R}^i_j)$ around clients $j \in D$. Remove the moats which intersect T_{i+1} and select from the rest a maximal subset $\mathcal{B}^i$ of non-overlapping moats in increasing order of their radii. We define S_i to be the set of clients whose non-overlapping moats are selected in this stage. Construct the set $\mathcal{B}^i_\zeta = \{B(j, \zeta \bar{R}^j_i) \colon j \in S_i\}$ of moats around clients in S_i.

2. Add a dummy node $\tilde{r}$ and connect it to every facility v fractionally opened by the LP (with $\bar{y}_v > 0$). Set the cost of each dummy edge $\tilde{e} = \tilde{r}v$ to be zero if facility $v \in I_{i+1}$; otherwise set it to be f_v. To simplify the analysis, associate each edge $\tilde{e} = \tilde{r}v$ with a variable $\tilde{x}_{\tilde{e}}$ equal to $\bar{y}_v$.

3. Contract each moat in $\mathcal{B}^i_\zeta$, and each component of T_{i+1} into super-nodes. Call the contracted graph $\widetilde{G}$.

4. Construct an approximately optimal Steiner tree $\hat{T}$ on $\widetilde{G}$, where the terminals are $\tilde{r}$ and all the super-nodes. Without loss of generality we assume that $\hat{T}$ includes a dummy edge of cost 0 from $\tilde{r}$ to every super-node associated to a component of T_{i+1} (or, more precisely, to each facility $v \in I_{i+1}$).

5. For each $v \in F \setminus I_{i+1}$, if dummy edge $\tilde{r}v$ is in $\hat{T}$ then open facility v and put it in I_i.

6. Set $I_i = I_i \cup I_{i+1}$.

7. Contract all the dummy edges that are contained in $\hat{T}$, and uncontract the super-nodes associated to the moats. The (real) edges from $\hat{T}$ form a forest in the resulting graph. To get a tree, add for each moat direct edges from its center to all nodes in the moat that are incident to $\hat{T}$. Let $\widetilde{T}$ be the resulting tree.

8. Using the LAST algorithm for appropriate parameters, transform $\widetilde{T}$ rooted at $\tilde{r}$ (or, more precisely, rooted at the contracted node that contains $\tilde{r}$) into a tree called LAST_i.

9. Install cables of type i along LAST_i and let $T_i = T_{i+1} \cup \text{LAST}_i$.

4.2.1.2 Analysis

Let $C^{*\text{op}}$, $C^{*\text{fixed}}$ and $C^{*\text{route}}$ be the opening cost, fixed installation cost and routing cost paid by the optimum to the LP relaxation of (IP-4-4).

Similar to Lemma 4.6, the cost of facilities opened in the facility selection phase can be bounded as follows.

Lemma 4.10. *The cost of facilities opened in the facility selection phase is* $O(1) \cdot C^{*op}$.

In the next lemma, we bound the fixed installation and opening cost incurred in the network phase of the algorithm.

Lemma 4.11. *The total fixed cost and facility cost of the network phase is* $O(1) \cdot C^{*fixed} + O(K) \cdot C^{*op}$.

Proof. Consider Stage i of this phase. Let S be an arbitrary subset of $V \setminus \{r\}$ that contains $B(j, \zeta \bar{R}_i^j)$. Similar to Lemma 4.8, one can show that $\sum_{q=1}^{i-1} \bar{b}_{q;S}^j \leq \frac{1}{\zeta}$, where $\bar{b}_{q;S}^j := \sum_{e \in \delta^+(S)} \bar{f}_{e,q}^j$ indicates the amount of flow from j crossing the boundary of S thorough cables of type q. This together with the LP constraints guarantee that $\sum_{q=i}^{K} \bar{b}_{q;S}^j + \sum_{v \in S \cap F} \bar{y}_v \geq 1 - \frac{1}{\zeta}$ and hence $\sum_{e \in \delta^+(S)} \left(\sum_{q=i}^{K} \bar{x}_e^q + \tilde{x}_e \right) \geq 1 - \frac{1}{\zeta}$. This means that the vector $\tilde{x} + \sum_{q=i}^{K} \bar{x}^q$, scaled by a factor $\frac{\zeta}{\zeta - 1}$, can be viewed as a feasible fractional solution to the LP relaxation the Steiner tree problem on $\widetilde{G}$ connecting balls $B(j, \zeta \bar{R}_i^j)$ to T_{i+1}. Therefore, the cost of this Steiner tree (including edge cost of Step 4

and opening cost of Step 5) can be bounded by

$$\frac{\text{gap}_{\text{ST}}\zeta}{\zeta-1}\left(\sum_{e\in E}\sum_{q=i}^{K} c_e\bar{x}_e^q + \sum_{v\in F}\mu_v\tilde{x}_{\tilde{r}v}\right) \le \frac{\text{gap}_{\text{ST}}\zeta}{(\zeta-1)\theta}\left(\sum_{q=i}^{K}\frac{1}{\sigma_q}C_q^{*\text{fixed}} + C^{*\text{op}}\right).$$

Similar to Lemma 4.8, the cost of the LAST_i tree can be bounded by

$$\text{gap}_{\text{ST}}\cdot\frac{\gamma}{\gamma-\zeta}\cdot\frac{\beta\zeta}{(\zeta-1)\theta}\left(\sum_{q=i}^{K}\frac{1}{\sigma_q}C_q^{*\text{fixed}} + C^{*\text{op}}\right).$$

Summing over all stages $1 \le i \le K$, we see the total fixed cost and facility cost of this phase is at most

$$\text{gap}_{\text{ST}}\cdot\frac{\gamma}{\gamma-\zeta}\cdot\frac{\beta\zeta}{(\zeta-1)\theta}\sum_{i=1}^{K}\left(\sum_{q=i}^{K}\frac{\sigma_i}{\sigma_q}C_q^{*\text{fixed}} + C^{*\text{op}}\right)$$

Which is $O(1)\cdot C^{*\text{fixed}} + O(K)\cdot C^{*\text{op}}$ by Theorem 4.5. ■

Let $T = \bigcup_{i=1}^{K}\text{LAST}_i$ be the access forest constructed by the algorithm. Note that the access network in the BBConFL problem form a forest too. Hence, in a similar way to Lemma 4.9, one can bound the routing cost of each stage, and then the routing cost of the entire solution by $O(1)\cdot C^{*\text{route}}$.

Lemma 4.12. *The total routing cost of the solution is* $O(1)\cdot C^{*route}$.

Using Lemmas 4.10, 4.11, and 4.12, the total cost of the solution is at most

$$O(1)\big(C^{*\text{route}} + C^{*\text{fixed}}\big) + O(K)\cdot C^{*\text{op}}.$$

This completes the proof of Theorem 4.2. ■

4.3 Conclusion

We have shown that the LP rounding framework for **uSSBB** given by Talwar (2002) extends to facility location buy-at-bulk problems. Our integrality gap analysis roughly matches the known approximation ratios of combinatorial algorithms for BBConFL and BBFL, so the obvious open problem is to improve this analysis to derive better approximation algorithms. In particular, can we get an $O(1)$-approximation for BBFL? We were able to bound the gap by $O(1)$ for BBConFL by exploiting the fact that the facility core network is fractionally connected by the LP. However, in BBFL we do not

have this property so we have to pay for the facility opening costs with a copy of the y-values in each stage.

A potentially easier problem is to get an α-approximation for BBFL with running time $n^{f(k)}$ for some function f where α is a constant that does not depend on k.

Chapter 5

Complex Connected Facility Location

The focus of our work in this chapter is on IP based techniques. We consider the Survivable Hop Constrained Connected Facility Location problem (**SHConFL**); see Problem 1.11. We study two strong extended formulations for this problem and present a branch-and-cut algorithm based on Benders decomposition for finding its solution. We evaluate our solution approach on a set of benchmark instances.

Some results presented in this chapter have been developed in joint work with Andreas Bley (Universität Kassel) and S. Mehdi Hashemi (Amirkabir University of Technology) and appeared in Bley et al. (2013b).

Previous Work

To the best of our knowledge the SHConFL problem has not yet been studied in the literature. However, it is related to the following NP-hard problems.

The Connected Facility Location Problem (ConFL). The ConFL problem is a special case of SHCoFL with $\lambda = 1$ and $H = \infty$, where λ is the required level of survivability and H is the hop limit (the maximum allowable path length). Several constant-factor approximation algorithms for this problem have been presented in the literature; we point the reader to Section 1.2.1 for more details. This problem has also gained attention in the operations research community. The first heuristic algorithm for the ConFL problem was given by Ljubić (2007) who combines the variable neighborhood search (VNS) with reactive tabu-search to devise the heuristic algorithm. Later Tomazic and Ljubić (2008) proposed a greedy randomized adaptive search procedure (GRASP)

algorithm to solve the problem. Bardossy and Raghavan (2010) formulated the ConFL problem as a directed Steiner tree problem with a unit degree constraint, and proposed a dual-based heuristic algorithm for the problem. An overview of formulations and polyhedral results for ConFL can be found in the paper by Gollowitzer and Ljubić (2011).

The Hop-Constrained Connected Facility Location Problem (HConFL). The HConFL problem is also a special case of SHConFL with $\lambda = 1$. This problem has been introduced by Ljubić and Gollowitzer (2013) who provided the first (and the only) theoretical and computational study on integer programming models for this problem.

The rooted Survivable Hop-Constrained Network Design Problem (rSHND). If the set of open facilities is given in advance, the SHConFL problem reduces to the rSHND problem. The first IP formulation for rSHND has been given by Botton et al. (2013) who present a branch-and-cut algorithm to solve the problem. The formulation presented by Botton et al. (2013) is then strengthened by Mahjoub et al. (2013) who proposed an extended IP formulation for rSHND, using an additional set of variables that indicate the distance of each demand node from the root in the solution.

In Section 5.2 we shall make use of these ideas to present strong extended formulations for SHCoFL.

Contributions

In Section 5.1 we see that computing a constant-factor approximation for the problem is already NP-hard. We hence focus our research only on IP based techniques. We propose two strong extended formulations for the SHConFL problem and provide their theoretical comparison in Section 5.2. To make the stronger model computationally tractable, in Section 5.3, we applied Benders decomposition approach projecting out the extended flow variables. In Section 5.4 we develop our branch-and-cut algorithm based on the Benders decomposition. To speed up the algorithm, we primarily separate connectivity inequalities stemming from (and valid for) the corresponding problem with connectivity constraints but without hop constraints, meaning that we only solve the computationally expensive separation problem for Benders cuts if the (computationally much easier) separation of connectivity inequalities fails. We report the results of our approach that were tested on a set of benchmark instances in Section 5.5.

5.1 Preliminaries

Recall that in SHConFL we are given an undirected graph $G = (V, E)$ with $V = D \dot{\cup} S$, where D is the set of clients and S is the set of (potential) core nodes. We are also given a set of potential facilities $F \subseteq S$ and a root $r \in S \setminus F$. Furthermore, we are given the costs $\mu_i \in \mathbb{Z}_{\geq 0}$ for opening facility $i \in F$, $a_{ij} \in \mathbb{Z}_{\geq 0}$ for assigning client $j \in D$ to facility $i \in F \cup \{r\}$, and $c_e \in \mathbb{Z}_{\geq 0}$ for installing edge $e \in E(S)$ in the (regional) core network (where $E(S) := \{uv \in E : u, v \in S\}$). Finally, we are given an integer hop limit $H \geq 1$ and an integer connectivity requirement $\lambda \geq 1$.

In SHConFL, we seek for a subset $I \subseteq F$ of facilities to open, a mapping $\sigma(j) : D \to I \cup \{r\}$ assigning clients to open facilities, and a subset of edges $E' \subseteq E(S)$ such that (S, E') contains, for each facility $i \in I$, at least λ edge-disjoint (r, i)-paths of length at most H. The objective is to minimize the total cost

$$\sum_{i \in I} \mu_i + \sum_{j \in D} a_{\sigma(j)j} + \sum_{e \in E'} c_e.$$

The SHConFL problem is obviously NP-hard, as it contains the connected facility location problem as a special case.

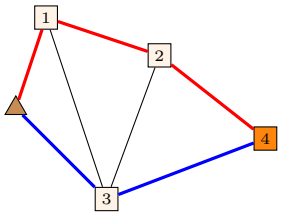
If the set of open facilities is given in advance and the connectivity requirement is fixed to 1 (i.e. $\lambda = 1$), then the SHConFL problem reduces to the *Hop-Constrained Steiner Tree* problem (**HST**). Manyem (2009) shows that the HST problem is not in **APX**, even if the edge weights satisfy the triangle inequality. The results by Manyem (2009) immediately imply the following inapproximability result for SHConFL too.

Theorem 5.1. *There is no approximation algorithm for SHConFL, even with the edge weights satisfying the triangle inequality, which guarantees a worst case approximation ratio better than* $\Theta(log(|V|))$ *unless* $\mathbf{NP} \subseteq \mathbf{DTIME}(n^{\log \log n})$.

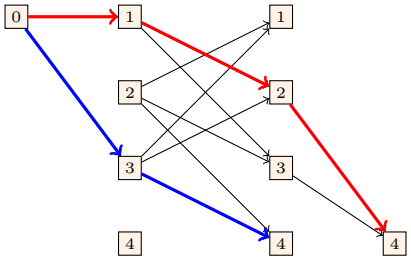
5.2 IP Modeling of SHCoFL on Layered Graphs

In this section we present two multicommodity flow based IP formulations for SHConFL.

As it has been used successfully in the earlier works on network design problems with hop-length restrictions (e.g. Gouveia et al., 2011; Botton et al., 2013), we model the hop-length constraints implicitly using layered graphs.



(a)A simple core network where the triangle node represents the root.



(b)A hop-expanded layered graph with $H = 3$ corresponding to facility 4

Figure 5.1: A core graph and a hop-expanded layered graph corresponding to facility 4. Edges in red and blue represent two routing paths between the root and facility 4 in the core graph and the corresponding hop-expanded layered graph.

Multi-Commodity Flow Based Formulation

We formulate SHCoFL on layered graphs. For each facility $i \in F$, we choose a commodity and create the following hop-extended directed graph $G^i = (S^i, A^i)$ with $H + 1$ layers:

$$S^i = \{r_0\} \cup \{u_h : u \in S,\, 1 \leq h < H\, u \neq r\} \cup \{i_H\} \text{ and}$$
$$A^i = \{(u_{h-1}, v_h) : u_{h-1}, v_h \in S^i,\, uv \in E_S,\, 1 \leq h \leq H\}.$$

where v_h, $v \in S$ and $0 \leq h \leq H$, represents a copy of node v in layer h of G^i. There are two directed edges (u_{h-1}, v_h) and (v_{h-1}, u_h) between layers $h-1$ and h of G^i corresponding to each edge uv in the original graph.

Observe that any (r, i) routing path of length $h \leq H$ in the original (core) graph corresponds to a directed path of the same length from r_0 to i_h in G^i; see Figure 5.1. In other words, the resulting layered graph $G^i = (S^i, A^i)$ contains only the allowable (r, i) routing paths with respect to the hop constraint H.

Notice that, in the structure described above, we made only the core graph layered and left the rest unchanged.

Now, to model SHCoFL as an IP, we introduce the following binary decision variables:

$$x_{ij} = \begin{cases} 1 & \text{if client } j \text{ is assigned to facility } i \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in F, j \in D;$$

$$y_i = \begin{cases} 1 & \text{if facility } i \text{ is opened} \\ 0 & \text{otherwise} \end{cases} \qquad \forall i \in F;$$

$$z_{uv} = \begin{cases} 1 & \text{if edge } uv \text{ is installed} \\ 0 & \text{otherwise} \end{cases} \qquad \forall uv \in E(S).$$

Additional binary flow variables f_{uv}^{ih} (for commodity i) indicate if a routing path from r to facility i contains arc (u, v) in position h or not.

With all the notation above, our multi-commodity flow-based IP formulation is as follows.

$$\text{(IP-5-1)} \quad \min \sum_{i \in F} \mu_i y_i + \sum_{j \in D} \sum_{i \in F \cup \{r\}} a_{ij} x_{ij} + \sum_{e \in E(S)} c_e z_e$$

$$\sum_{i \in F \cup \{r\}} x_{ij} = 1 \qquad \forall j \in D \tag{5.1}$$

$$y_i - x_{ij} \geq 0 \qquad \forall i \in F, j \in D \tag{5.2}$$

$$\sum_{h=0}^{H-1} \sum_{(u_h, i_{h+1}) \in A^i} f_{ui}^{ih} = \lambda y_i \qquad \forall i \in F \tag{5.3}$$

$$\sum_{(v_{h-1}, u_h) \in A^i} f_{vu}^{i(h-1)} - \sum_{(u_h, v_{h+1}) \in A^i} f_{uv}^{ih} = 0 \qquad \forall i \in F, u_h \in S^i; u \notin \{r, i\} \tag{5.4}$$

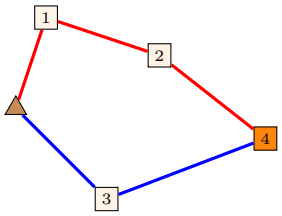
$$f_{rv}^{i0} \leq z_{rv} \qquad \forall i \in F, rv \in E(S) \tag{5.5}$$

$$\sum_{h=1}^{H-2} (f_{uv}^{ih} + f_{vu}^{ih}) \leq z_{uv} \qquad \forall i \in F, uv \in E(S), u, v \notin \{r, i\} \tag{5.6}$$

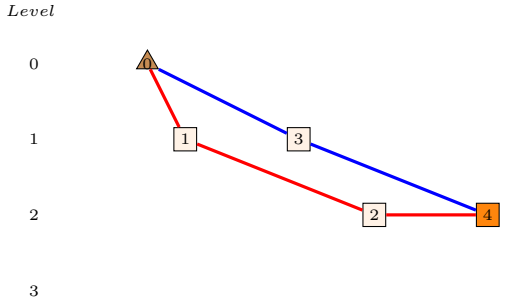
$$\sum_{h=1}^{H-1} f_{ui}^{ih} \leq z_{ui} \qquad \forall i \in F, ui \in E(S), u \neq r \tag{5.7}$$

$$x, y, z, f \in \{0, 1\} \tag{5.8}$$

Constraints (5.1) and (5.2) state that any client has to be assigned to an open facility. Constraints (5.3) and (5.4) ensure λ units of flow going from each open facility to root r on the layered graphs. This hence guarantees λ length bounded flow paths between r and each open facility in the original graph; thanks to layered graphs. Constraints (5.5)-(5.7) finally guarantee the edge-disjointness of these paths and the correct setting of the edge variables z_e.



(a)A core network



(b)A corresponding level-expanded core network

Figure 5.2: The level-expanded core network with $H = 3$ & $\lambda = 2$ corresponding to the above solution

Hop-Level Multi-Commodity Flow Based Formulation.

In this section we provide a stronger formulation for SHCoFL. Similar to Mahjoub et al. (2013), we introduce an additional set of variables and constraints to get the distance of each core node from the root node in the solution.

More precisely, given a feasible solution (I^*, σ^*, E^*) to SHConFL, we partition nodes in S into $H + 2$ levels according to their distances[1] from r (see Figure 5.2) as follows:

- Level 0 only contains r,
- level l, $1 \leq l \leq H$, contains nodes with distance l form r,
- level $H+1$ contains the nodes that are not connected to r in E^*.

Then, for every node $u \in S$ and level $1 \leq l \leq H + 1$ we introduce an binary variable w_u^l that indicates if node u is in level l. We also introduce binary variables a_{uv}^{lk}, $uv \in E(S)$ and $0 \leq l, k \leq H$, indicating if edge uv belongs to the solution with u in level l and v in level k, respectively. Remark that $|k - l| \leq 1$.

Let $\delta(u) = \{uv \in E(S) : v \in S\}$ and let $\hat{E}_S = E_S \setminus \delta(r)$. Constraints (5.9)-(5.20) are devised to transform the core part of a solution over the y and z variables into a

[1]By distance between two nodes u and v in E^* we mean the (minimum) number of edges in E^* connecting u to v. Note that the maximum possible level for an open facility is H.

corresponding level-expanded solution over the y, w, and a variables.

$$\sum_{l=1}^{H+1} w_u^l = 1 \qquad \forall u \in S \setminus \{r\} \tag{5.9}$$

$$w_i^{H+1} \leq 1 - y_i \qquad \forall i \in F \setminus \{r\} \tag{5.10}$$

$$w_v^1 = a_{rv}^{01} = z_{rv} \qquad \forall rv \in E_S \tag{5.11}$$

$$\sum_{l=1}^{H-1} a_{uv}^{ll} + \sum_{l=1}^{H-1} (a_{uv}^{l(l+1)} + a_{vu}^{l(l+1)}) = z_{uv} \qquad \forall uv \in \hat{E}_S \tag{5.12}$$

$$a_{uv}^{11} + a_{uv}^{12} \leq w_u^1 \qquad \forall uv \in \hat{E}_S \tag{5.13}$$

$$a_{uv}^{11} + a_{vu}^{12} \leq w_v^1 \qquad \forall uv \in \hat{E}_S \tag{5.14}$$

$$a_{uv}^{ll} + a_{uv}^{l(l+1)} + a_{vu}^{(l-1)l} \leq w_u^l \qquad \forall uv \in \hat{E}_S, 2 \leq l \leq H-1 \tag{5.15}$$

$$a_{uv}^{ll} + a_{vu}^{l(l+1)} + a_{uv}^{(l-1)l} \leq w_v^l \qquad \forall uv \in \hat{E}_S, 2 \leq l \leq H-1 \tag{5.16}$$

$$a_{vu}^{(H-1)H} \leq w_u^H \qquad \forall uv \in \hat{E}_S \tag{5.17}$$

$$a_{uv}^{(H-1)H} \leq w_v^H \qquad \forall uv \in \hat{E}_S \tag{5.18}$$

$$w_v^l - \sum_{(u,v)\in\delta(v), u\neq r} a_{uv}^{(l-1)l} \leq 0 \qquad \forall v \in S, 2 \leq l \leq H \tag{5.19}$$

$$w, a, y, z \in \{0,1\} \tag{5.20}$$

Constraints (5.9)-(5.10) state that each node should be in exactly one of the possible levels and chosen facilities must be reachable. Constraints (5.11) and (5.12) make the connection between (w, a) variables and variables z. Constraints (5.13)-(5.18) ensure that a variable a_{uv}^{lk} can only be 1 if both w_u^l and w_v^k are one. Constraints (5.19) state that a node can only be in level l if it is reached by at least one edge from level $l-1$.

Recall that, using Constraints (5.9)-(5.20), any binary vector (y, z) defines binary values (y, w, a) in which each node is assigned to a single level. But, analogous to Mahjoub et al. (2013), it can be shown that a fractional (w, a) splits nodes and edges into different levels. This possibly increases the distance between r and open facilities in the level-expanded network induced by (w, a). For example, Figure 5.3(a) illustrates a feasible fractional core network for $H = 3$ & $\lambda = 2$, containing 2 edge-disjoint (fractional) paths of length at most 3 between the root node and each facility which can be specified as follows: $\hat{z}_{01} = \hat{z}_{34} = 1, \hat{z}_{02} = \hat{z}_{04} = \hat{z}_{12} = \hat{z}_{13} = \hat{z}_{23} = \hat{z}_{24} = 1/2$; and Figure 5.3(b) illustrates its corresponding level-expanded solution specified as follows: $\bar{w}_1^0 = \bar{w}_1^1 = \bar{w}_3^1 = 2, \bar{w}_2^1 = \bar{w}_2^2 = \bar{w}_4^2 = \bar{w}_4^1 = 1/2$, $\bar{a}_{01}^{01} = 1$, $\bar{a}_{02}^{01} = \bar{a}_{04}^{01} = \bar{a}_{12}^{12} = \bar{a}_{13}^{12} = \bar{a}_{24}^{12} = \bar{a}_{43}^{12} = \bar{a}_{23}^{22} = \bar{a}_{34}^{22} = 1/2$.

It is easy to see that, unlike the original (fractional) solution, the level-expanded solution violates the required hop constraint; see the routing path (in red) between r and facility (with length $5 > H$). In fact this is the main reason why we are interested in this level based transformation as we are able now to cut away such fractional solutions by

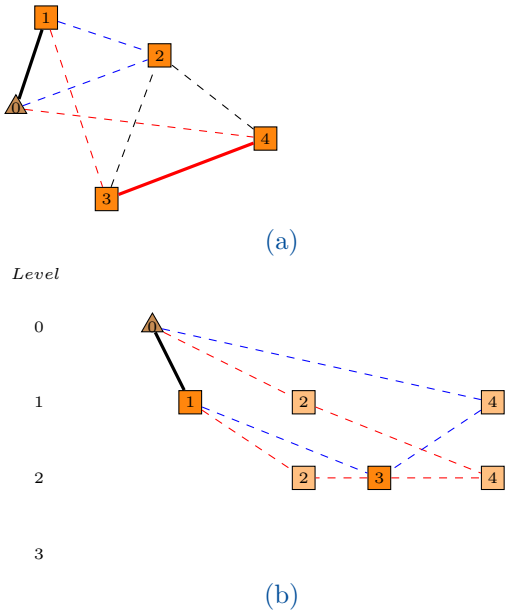


Figure 5.3: A feasible fractional core network with $H = 3$ & $\lambda = 2$ and its corresponding level-expanded solution.

ensuring the existence of the hop-limited arc-disjoint routing paths between r and open facilities in the level-expanded network, instead of the original network. In the following we model this.

We formulate this on layered graphs to ensure the hop constraints. For each facility $i \in F$, we create a directed level- and hop-expanded graph $G_L^i = (S_L^i, A_L^i)$ with

$$\begin{aligned} S_L^i &= \{r_0^0\} \\ &\quad \cup \{u_h^l : u \in S,\ 1 \le l \le h \le H-1,\ u \ne r\} \\ &\quad \cup \{i_H^l : 1 \le l \le H\}, \text{and} \\ A_L^i &= \{(u_{h-1}^l, v_h^k) : u_{h-1}^l, v_h^k \in S_L^i,\ uv \in E_S,\ 1 \le h \le H,\ |l-k| \le 1\} \\ &\quad \cup \{(i_h^l, i_{h+1}^l) : 1 \le l \le h \le H-1\} \end{aligned}$$

In which the copy of node u in layer h and level l is denoted by u_h^l, while (u_{h-1}^l, v_h^k) denotes the directed arc corresponding to the copy of edge uv with node u in layer $h-1$ and level l and node v in layer h and level k; see Figure 5.4.

Now, corresponding to each arc of the level- and hop-expanded layered graph, we introduce a binary flow variable g_{uv}^{ihlk} indicating that a path from r to facility i uses arc (u, v) in hop h where u is at level l and v is at level k.

With all the notation above, our extended IP formulation for SHCoFL is as follows.

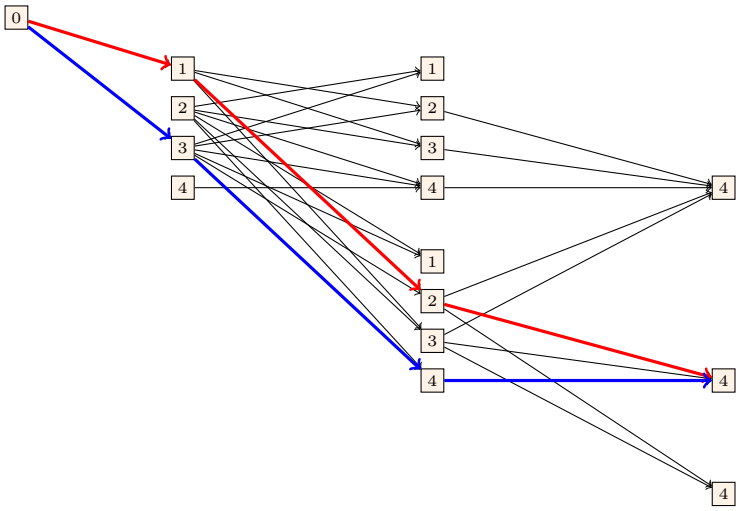


Figure 5.4: The level- and hop-expanded layered graph with $H = 3$ & $i = 4$ corresponding to the core graph shown in Figure 5.1a.

$$\text{(IP-5-2)} \quad \min \sum_{i \in F} \mu_i y_i + \sum_{j \in D} \sum_{i \in F \cup \{r\}} a_{ij} x_{ij} + \sum_{e \in E_S} c_e z_e$$

$$(5.1) - (5.2)$$

$$(5.9) - (5.19)$$

$$\sum_{l=1}^{H} \sum_{(v_{H-1}^k, i_H^l) \in A_L^i} g_{vi}^{i(H-1)kl} = \lambda y_i \qquad \forall i \in F \tag{5.21}$$

$$\sum_{(v_{h-1}^k, u_h^l) \in A_L^i} g_{vu}^{i(h-1)kl} - \sum_{(u_h^l, v_{h+1}^k) \in A_L^i} g_{uv}^{ihlk} = 0 \qquad \forall i \in F, u_h^l \in S_L^i, u \neq r, h \leq H-1 \tag{5.22}$$

$$g_{rv}^{i001} \leq a_{rv}^{01} \qquad \forall i \in F, rv \in E_S \tag{5.23}$$

$$\sum_{h=l}^{H-2} (g_{uv}^{ihll} + g_{vu}^{ihll}) \leq a_{uv}^{ll} \qquad \forall i \in F, uv \in \hat{E}_S \setminus \delta(i), 1 \leq l \leq H-2 \tag{5.24}$$

$$\sum_{h=l}^{H-2} g_{uv}^{ihl(l+1)} + \sum_{h=l+1}^{H-2} g_{vu}^{ih(l+1)l} \leq a_{uv}^{l(l+1)} \qquad \forall i \in F, uv \in \hat{E}_S \setminus \delta(i), 1 \leq l \leq H-2 \tag{5.25}$$

$$\sum_{h=l}^{H-1} g_{ui}^{ihll} \leq a_{ui}^{ll} \qquad \forall i \in F, ui \in \delta(i) \setminus \delta(r), 1 \leq l \leq H-1 \tag{5.26}$$

$$\sum_{h=l}^{H-1} g_{ui}^{ihl(l+1)} \leq a_{ui}^{l(l+1)} \qquad \forall i \in F, ui \in \delta(i) \setminus \delta(r), 1 \leq l \leq H-1 \tag{5.27}$$

$$w, a, y, x, z, g \in \{0, 1\}$$

Constraints (5.21)-(5.27) ensure the existence of the λ arc-disjoint paths between the root node and each open facility in the layer- and level-extended graphs. More precisely, Constraints (5.21)-(5.22) guarantee that at least λ units of flow go from r to each open

facility on the the layer- and level-extended graphs. Constraints (5.23)-(5.27) then link the flow to the a variables.

Let P_1 and P_2 denote the feasible space of the LP relaxation corresponding to formulations IP-5-1 and IP-5-2, respectively.

Theorem 5.2. $Proj_{X,Y,Z}(P_2) \subseteq Proj_{X,Y,Z}(P_1)$.

Proof. Given a fractional solution $(\bar{\mathbf{x}}, \bar{\mathbf{y}}, \bar{\mathbf{z}}, \bar{\mathbf{a}}, \bar{\mathbf{w}}, \bar{\mathbf{g}})$ to IP-5-2. We construct an equivalent solution $(\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}}, \hat{\mathbf{f}})$ to IP-5-1 as follows. We take the vectors $\hat{\mathbf{x}} = \bar{\mathbf{x}}$, $\hat{\mathbf{y}} = \bar{\mathbf{y}}$, and $\hat{\mathbf{z}} = \bar{\mathbf{z}}$ the same for the newly constructed solution. But for the flow variables we set

$$\hat{f}_{uv}^{ih} = \sum_{l,k} \bar{g}_{uv}^{ihlk}, \quad \forall i \in F, uv \in E(S), 0 \leq h \leq H.$$

It is not hard to check that the constructed solution satisfies Constraints (5.1)-(5.8), and hence is a feasible solution to IP-5-1. This completes the proof. ■

Recalling the instance described in Figure 5.3, we conclude that IP-5-2 is stronger than IP-5-1 in terms of the lower bounds provided by their linear relaxations.

5.3 Benders Decomposition

We use the Benders decomposition approach to efficiently handle the huge number of variables and constraints of IP-5-2. Notice that we cannot directly apply Benders decomposition method, because all variables of IP-5-2 are integer and classical duality theory does not allow to project out integer variables. Therefore, we first relax the integrality restrictions of the flow variables and apply Benders decomposition to this relaxation, called MIP-5-2. We then discuss whether MIP-5-2 provides the same optimal solution on design variables x, y, and z as IP-5-2.

Let the master problem be defined as follows.

$$\text{(Master)} \quad \min \sum_{i\in F} \mu_i y_i + \sum_{j\in D} \sum_{i\in F\cup\{r\}} a_{ij}x_{ij} + \sum_{e\in E_S} c_e z_e$$

$$\sum_{i\in F\cup\{r\}} x_{ij} = 1 \qquad \forall j \in D \tag{5.1}$$

$$y_i - x_{ij} \geq 0 \qquad \forall i \in F, j \in D \tag{5.2}$$

$$\sum_{l=1}^{H+1} w_u^l = 1 \qquad \forall u \in S \setminus \{r\} \tag{5.9}$$

$$w_i^{H+1} \leq 1 - y_i \qquad \forall i \in F \setminus \{r\} \tag{5.10}$$

$$w_v^1 = a_{rv}^{01} = z_{rv} \qquad \forall rv \in E_S \tag{5.11}$$

$$\sum_{l=1}^{H-1} a_{uv}^{ll} + \sum_{l=1}^{H-1} (a_{uv}^{l(l+1)} + a_{vu}^{l(l+1)}) = z_{uv} \qquad \forall uv \in \hat{E}_S \tag{5.12}$$

$$a_{uv}^{11} + a_{uv}^{12} \leq w_u^1 \qquad \forall uv \in \hat{E}_S \tag{5.13}$$

$$a_{uv}^{11} + a_{vu}^{12} \leq w_v^1 \qquad \forall uv \in \hat{E}_S \tag{5.14}$$

$$a_{uv}^{ll} + a_{uv}^{l(l+1)} + a_{vu}^{(l-1)l} \leq w_u^l \qquad \forall uv \in \hat{E}_S, 2 \leq l \leq H-1 \tag{5.15}$$

$$a_{uv}^{ll} + a_{vu}^{l(l+1)} + a_{uv}^{(l-1)l} \leq w_v^l \qquad \forall uv \in \hat{E}_S, 2 \leq l \leq H-1 \tag{5.16}$$

$$a_{vu}^{(H-1)H} \leq w_u^H \qquad \forall uv \in \hat{E}_S \tag{5.17}$$

$$a_{uv}^{(H-1)H} \leq w_v^H \qquad \forall uv \in \hat{E}_S \tag{5.18}$$

$$w_v^l - \sum_{(u,v)\in\delta(v), u\neq r} a_{uv}^{(l-1)l} \leq 0 \qquad \forall v \in S, 2 \leq l \leq H \tag{5.19}$$

$$w, a, y, x, z \in \{0,1\}$$

We notice that a solution $\bar{y}, \bar{x}, \bar{z}, \bar{w}, \bar{a}$ of the master problem defines a feasible solution for MIP-5-2 if and only if for each $i \in F$ there exist fractional flow values satisfying the following linear system of equations, called $\mathbf{SUB}^i$.

$$\sum_{l=1}^{H} \sum_{(v_{H-1}^k, i_H^l)\in A_L^i} g_{vi}^{i(H-1)kl} = \lambda \bar{y}_i \tag{5.28}$$

$$\sum_{(v_{h-1}^k, u_h^l)\in A_L^i} g_{vu}^{i(h-1)kl} - \sum_{(u_h^l, v_{h+1}^k)\in A_L^i} g_{uv}^{ihlk} = 0 \qquad \forall u_h^l \in S_L^i, u \neq r, h \leq H-1 \tag{5.29}$$

$$g_{rv}^{i001} \leq \bar{a}_{rv}^{01} \qquad \forall rv \in E_S \tag{5.30}$$

$$\sum_{h=l}^{H-2} (g_{uv}^{ihll} + g_{vu}^{ihll}) \leq \bar{a}_{uv}^{ll} \qquad \forall uv \in \hat{E}_S \setminus \delta(i), 1 \leq l \leq H-2 \tag{5.31}$$

$$\sum_{h=l}^{H-2} g_{uv}^{ihl(l+1)} + \sum_{h=l+1}^{H-2} g_{vu}^{ih(l+1)l} \leq \bar{a}_{uv}^{l(l+1)} \qquad \forall uv \in \hat{E}_S \setminus \delta(i), 1 \leq l \leq H-2 \tag{5.32}$$

$$\sum_{h=l}^{H-1} g_{ui}^{ihll} \leq \bar{a}_{ui}^{ll} \qquad \forall ui \in \delta(i) \setminus \delta(r), 1 \leq l \leq H-1 \tag{5.33}$$

$$\sum_{h=l}^{H-1} g_{ui}^{ihl(l+1)} \leq \bar{a}_{ui}^{l(l+1)} \qquad \forall ui \in \delta(i) \setminus \delta(r), 1 \leq l \leq H-1 \tag{5.34}$$

$$g_{uv}^{ihlk} \in [0,1] \tag{5.35}$$

Farkas' lemma states that a linear system of equations $\{Ax \leq b, x \geq 0\}$ has a solution if and only if $u^T b \geq 0$ for all $u \geq 0$ such that $u^T A \geq 0$. To apply Farkas' lemma to this linear system, we define dual variables π, $\pi_{u,l}^h$, $\sigma_{rv}^{0(1)}$, $\sigma_{uv}^{l(l)}$, $\sigma_{uv}^{l(l+1)}$, $\sigma_{ui}^{l(l)}$, $\sigma_{ui}^{l(l+1)}$ associated to the constraints (5.28)-(5.34), respectively. Therefore, the polyhedron defined by system SUB^i is nonempty if and only if the following dual system is bounded.

$$\text{(D-SUB}^i\text{)} \quad \max \lambda \bar{y}_i \pi - \sum_{uv \in E_S} \sum_{\substack{0 \leq l,k \leq H-1: \\ |k-l| \leq 1}} \bar{a}_{uv}^{lk} \sigma_{uv}^{l(k)}$$

$$\pi_{v,1}^1 - \sigma_{rv}^{01} \leq 0 \qquad \forall rv \in E_S \tag{5.36}$$

$$\pi_{v,l}^{h+1} - \pi_{u,l}^h - \sigma_{uv}^{ll} \leq 0 \qquad \forall uv \in \hat{E}_S \setminus \delta(i); 1 \leq l \leq h \leq H-2 \tag{5.37}$$

$$\pi_{u,l}^{h+1} - \pi_{v,l}^h - \sigma_{uv}^{ll} \leq 0 \qquad \forall uv \in \hat{E}_S \setminus \delta(i); 1 \leq l \leq h \leq H-2 \tag{5.38}$$

$$\pi_{v,l+1}^{h+1} - \pi_{u,l}^h - \sigma_{uv}^{l(l+1)} \leq 0 \qquad \forall uv \in \hat{E}_S \setminus \delta(i); 1 \leq l \leq h \leq H-2 \tag{5.39}$$

$$\pi_{u,l}^{h+1} - \pi_{v,l+1}^h - \sigma_{uv}^{l(l+1)} \leq 0 \qquad \forall uv \in \hat{E}_S \setminus \delta(i); 1 \leq l < h \leq H-2 \tag{5.40}$$

$$\pi_{v,l}^{h+1} - \pi_{u,l+1}^h - \sigma_{uv}^{(l+1)l} \leq 0 \qquad \forall uv \in \hat{E}_S \setminus \delta(i); 1 \leq l < h \leq H-2 \tag{5.41}$$

$$\pi_{u,l+1}^{h+1} - \pi_{v,l}^h - \sigma_{uv}^{(l+1)l} \leq 0 \qquad \forall uv \in \hat{E}_S \setminus \delta(i); 1 \leq l \leq h \leq H-2 \tag{5.42}$$

$$\pi_{i,l}^{h+1} - \pi_{u,l}^h - \sigma_{ui}^{ll} \leq 0 \qquad \forall ui \in \hat{E}_S; 1 \leq l \leq h \leq H-2 \tag{5.43}$$

$$\pi_{i,l+1}^{h+1} - \pi_{u,l}^h - \sigma_{ui}^{l(l+1)} \leq 0 \qquad \forall ui \in \hat{E}_S; 1 \leq l \leq h \leq H-2 \tag{5.44}$$

$$\pi_{i,l}^{h+1} - \pi_{u,l+1}^h - \sigma_{ui}^{(l+1)l} \leq 0 \qquad \forall ui \in \hat{E}_S; 1 \leq l < h \leq H-2 \tag{5.45}$$

$$\pi - \pi_{u,l}^{H-1} - \sigma_{ui}^{ll} \leq 0 \qquad \forall ui \in \hat{E}_S; 1 \leq l \leq H-1 \tag{5.46}$$

$$\pi - \pi_{u,l}^{H-1} - \sigma_{ui}^{l(l+1)} \leq 0 \qquad \forall ui \in \hat{E}_S; 1 \leq l \leq H-1 \tag{5.47}$$

$$\pi - \pi_{u,l+1}^{H-1} - \sigma_{ui}^{(l+1)l} \leq 0 \qquad \forall ui \in \hat{E}_S; 1 \leq l \leq H-1 \tag{5.48}$$

$$\pi_{i,l}^{h+1} \leq \pi_{i,l}^h \qquad \forall i \in F; 1 \leq l \leq h \leq H-2 \tag{5.49}$$

$$\pi \leq \pi_{i,l}^{H-1} \qquad \forall i \in F; 1 \leq l \leq H-1 \tag{5.50}$$

$$\pi \leq 1 \tag{5.51}$$

$$\sigma_{uv}^{lk} \geq 0 \qquad \forall uv \in \hat{E}_S, k, l : |k-l| \leq 1 \tag{5.52}$$

Note that in the case that the subproblem is unbounded, we can add the following *Benders cut* to the master problem to avoid this situation.

$$\lambda y_i \bar{\pi} - \sum_{uv \in E_S} \sum_{\substack{0 \le l,k \le H-1: \\ |k-l| \le 1}} a_{uv}^{lk} \bar{\sigma}_{uv}^{l(k)} \le 0 \quad \forall (\bar{\pi}, \bar{\sigma}) \in \Pi(i) \tag{5.53}$$

Where $\Pi(i)$ denotes the set of extreme rays of the above dual system.

Altogether, the Benders Reformulation (BR) of MIP-5-2 is as follows.

$$\begin{aligned}
\text{(BR)} \quad & \min \sum_{i \in F} \mu_i y_i + \sum_{j \in D} \sum_{i \in F \cup \{r\}} a_{ij} x_{ij} + \sum_{e \in E_S} c_e z_e \\
& (5.1) - (5.2) \\
& (5.9) - (5.19) \\
& \lambda y_i \bar{\pi} - \sum_{uv \in E_S} \sum_{\substack{0 \le l,k \le H-1: \\ |k-l| \le 1}} a_{uv}^{lk} \bar{\sigma}_{uv}^{l(k)} \le 0 \qquad \forall (\bar{\pi}, \bar{\sigma}) \in \cup_{i \in F} \Pi(i) \qquad (5.54) \\
& w, a, y, x, z \in \{0,1\}
\end{aligned}$$

We next discuss whether the Benders Reformulation of MIP-5-2 provides the same optimal design variables x, y, and z as IP-5-2.

Let ξ_{int} and ξ_{frac} denote the set of all binary vectors $(\bar{x}, \bar{y}, \bar{z}, \bar{w}, \bar{a})$ for which there exists an integral or a fractional solution for (SUB^i), $\forall i \in F$, respectively. Since $\xi_{int} \subseteq \xi_{frac}$, any Benders cut (5.53) is a valid inequality for conv(ξ_{int}), as well. The following Lemma shows that these Benders cuts are sufficient to describe ξ_{int} in some special cases, but not in general.

Lemma 5.3. *ξ_{int} and ξ_{frac} are equal for $H=2,3$ with any λ, and $H=4$ with $\lambda=2$. For $H \ge 5$, there exist a $\lambda \ge 2$ for which $\xi_{int} \subsetneq \xi_{frac}$, unless P=NP.*

The results follow from the corresponding results for the SHND problem given by Botton et al. (2013) (see also Bley and Neto, 2010).

For general λ and H, we retreat to the generation of *feasibility cuts* to cut off (integer-) infeasible solutions $(\bar{x}, \bar{y}, \bar{z}, \bar{w}, \bar{a}) \in \xi_{frac} \setminus \xi_{int}$. For this, we need to check if there exist integral flows satisfying (SUB^i). This can be modeled as the following integer linear program.

$$(\mathrm{FC}^i) \quad \min \sum_{uv \in E_S} e_{uv}$$

$$\sum_{l=0}^{H-1} \sum_{(u,i,l) \in A_l^i} f_{ui}^{i,l} = \lambda \bar{y}_i \tag{5.55}$$

$$\sum_{(v,u,l-1) \in A_{l-1}^i} f_{vu}^{i,l-1} - \sum_{(u,v,l) \in A_l^i} f_{uv}^{il} = 0 \qquad \forall (u,l) \in S^i; u \notin \{r,i\} \tag{5.56}$$

$$f_{rv}^{i,0} \leq \bar{z}_{rv} + e_{rv}(1 - \bar{z}_{rv}) \qquad \forall rv \in E_S \tag{5.57}$$

$$\sum_{l=1}^{H-2} (f_{uv}^{il} + f_{vu}^{il}) \leq \bar{z}_{uv} + e_{uv}(1 - \bar{z}_{uv}) \qquad \forall uv \in E_S, u \neq i, v \neq i \tag{5.58}$$

$$\sum_{l=1}^{H-1} f_{ui}^{il} \leq \bar{z}_{ui} + e_{ui}(1 - \bar{z}_{ui}) \qquad \forall ui \in E_S, u \neq r \tag{5.59}$$

$$f_{uv}^{il} \in \{0,1\} \qquad \forall i \in Fuv \in E_S, l \tag{5.60}$$

$$e_{uv} \in \{0,1\} \qquad \forall uv \in E_S \tag{5.61}$$

Let opt^i be an optimal objective value of FC^i, for some $i \in F$. If opt^i is zero, for every i, we have $(\bar{x}, \bar{y}, \bar{z}, \bar{w}, \bar{a}) \in \xi_{int}$; otherwise, we have to add the following valid inequality (feasibility cut) to cut off the infeasible integer solutions

$$\sum_{uv:\bar{z}_{uv}=0} z_{uv} \geq opt^i \cdot y_i \tag{5.62}$$

5.4 Branch-and-Cut Algorithm

We implemented a branch-and-cut algorithm based on the Benders decomposition discussed above. In every node of the branch-and-bound tree, we solve the master problem to get some optimal (fractional) solution. Then, we first separate (non-hop-restricted) connectivity inequalities of the following form

$$z(\delta(U)) \geq \lambda y_i \tag{5.63}$$

With $U \subseteq S$, $r \notin U$, $i \in U$ for each facility $i \in F$. This can be done via max-flow min-cut computations. In other words, in order to speed up the algorithm, we primarily separate connectivity inequalities stemming from (and valid for) the corresponding problem with connectivity constraints but without hop constraints. Only if no violated connectivity cuts are found, we check for violated Benders cuts, which is computationally far more expensive.

Formally, in every node of the branch-and-bound tree the algorithm works as follows:

1. Solve the master problem; take the optimal (fractional) solution.
2. Check if the current solution satisfies the connectivity requirements between root and facilities. As long as there are violated connectivity cuts (5.63), add them to the master and goto step (1).
3. Solve the linear Benders subsystems (D-SUB^i) for the current solution. If this results in a violated Benders cut, add it to the master and goto step (1).
4. Only at integer nodes of tree (and if needed; see Lemma 5.3): Solve the integer subsystems (FC^i) for the current solution. If infeasible, add corresponding feasibility cut (5.62) to the master and goto step (1).

Our computational experiences (see Section 5.5) show that connectivity cuts generated early in step (2) are very important to avoid the exploration of too many infeasible nodes and to reduce the time spent in the computationally expensive generation of Benders cuts.

5.5 Computational Results

The branch-and-cut algorithm has been implemented in C++, using SCIP version 3.1.0. (Achterberg, 2009) as a framework and CPLEX 12.6.0 (IBM, 2013) as an LP solver, and run on a machine with a AMD Phenom(tm) II X6 1090T 3GHz and 8 GiB RAM. For the connectivity cuts we use a push-relabel maximum flow algorithm.

Instances Details

To generate our instances, we combine benchmarks from the SHND problem and benchmarks from uncapacitated facility location problem (UFL) for the core and access part of our instances, respectively. We follow closely Botton et al. (2013) and generate the core graph as follows. Instances are three graphs whose nodes sets are of size $\{50, 75, 100\}$ randomly placed among integer points of a grid 100×100. The first $|F| \in \{30, 40\}$ nodes are selected to be potential facilities and the node with index 1 is selected as the root node. Out of all possible edges for the graph, we pick each edge with probability 0.3. The edge costs are set to be the Euclidean distance between any two points. Now given an instance of UFL, $|F| \in \{30, 40\}$ facilities are randomly selected. The number of clients, facility opening costs and assignment costs are provided by UFL instance. We

consider a set of instances obtained by combining two UFL instances $mp1$ and $mq1$[2] (of size 200×200 and 300×300, respectively) with the three generated core instances as described in Tables 5.1 and 5.2.

Computational Experiments

In Tables 5.1 and 5.2, we report the performance of our approach applied for the randomly generated networks combined with the mp and mq instances. We impose an optimality gap limit of 1.0 % and a time limit of 1000 seconds, and report the number of branch and bound nodes visited, denoted by "B&B"; the number of cuts of types "connectivity cut" and "Benders cut" generated throughout the execution of our algorithm (no feasibility cuts have been generated in any instance), denoted by "#ConnCuts", and "#BendCuts", respectively; the root gap; the final gap; and the solving time. The results show that easily computable connectivity cuts are very important to avoid generating many expensive Benders cuts. This allows us to solve the most of the generated instances within the time limit.

For the algorithm without Step (ii), Table 5.3 reports the results after a run time of 7200 seconds (two hours) only for a subset of the large instances. An optimality gap limit of 1.0 % is used. The results in Table 5.3 (compare to same results in Tables 5.1 and 5.2) show the practical improvement of the algorithm due to the addition of cuts stemming from plain (non-length-restricted) connectivity problem.

Table 5.3: Results of our algorithm without Step (ii) on a subset of the instances

Instance	H	λ	$\lvert S\rvert$	$\lvert E_S\rvert$	$\lvert F\rvert$	$\lvert D\rvert$	#BendCuts	Final-Gap [%]	**Time(s)**
MP1-30–Core100	5	3	100	1,685	30	200	10,768	0.9	5,982.1
MP1-30–Core100	5	5	100	1,685	30	200	21,032	4.3	7,200.11
MP1-40–Core100	5	3	100	1,685	40	200	15,745	1.0	5,977.29
MP1-40–Core100	5	5	100	1,685	40	200	22,625	3.4	7,200.36
MQ1-30–Core100	5	3	100	1,685	30	300	11,647	5.8	7,200.01
MQ1-30–Core100	5	5	100	1,685	30	300	11,590	15.1	7,200.02
MQ1-40–Core100	5	3	100	1,685	40	300	11,090	18.8	7,200.18
MQ1-40–Core100	5	5	100	1,685	40	300	9,983	14.9	7,200.1

[2]Available at http://www.mpi-inf.mpg.de/departments/d1/projects/benchmarks/UflLib.

Table 5.1: Results of our algorithm on the *mp* instances

Instance	H	λ	$\|S\|$	$\|E_S\|$	$\|F\|$	$\|D\|$	$B\&B$	#ConnCuts	#BendCuts	Root-Gap [%]	Final-Gap [%]	**Time(s)**
MP1-30–Core50	3	3	50	413	30	200	177	4,088	24	18.1	0.9	89.19
MP1-30–Core50	3	5	50	413	30	200	49	2,421	8	14.6	0.0	86.7
MP1-30–Core50	5	3	50	413	30	200	181	4,066	32	18.1	0.0	125.56
MP1-30–Core50	5	5	50	413	30	200	51	3,107	0	12.3	0.0	80.82
MP1-30–Core75	3	3	75	918	30	200	68	1,885	20	19.4	0.9	66.56
MP1-30–Core75	3	5	75	918	30	200	396	3,424	448	16.2	0.8	241.34
MP1-30–Core75	5	3	75	918	30	200	71	1,247	26	19.4	0.3	132.9
MP1-30–Core75	5	5	75	918	30	200	112	2,889	7	16.2	0.8	124.78
MP1-30–Core100	3	3	100	1,685	30	200	1,121	9,581	818	18.1	0.9	493.04
MP1-30–Core100	3	5	100	1,685	30	200	356	5,189	42	14.7	0.5	200.11
MP1-30–Core100	5	3	100	1,685	30	200	249	3,408	17	18.4	0.9	459.65
MP1-30–Core100	5	5	100	1,685	30	200	189	3,973	0	14.7	0.7	956.24
MP1-40–Core50	3	3	50	413	40	200	233	5,018	108	19.7	0.3	143.22
MP1-40–Core50	3	5	50	413	40	200	75	3,771	18	16.1	0.0	101.54
MP1-40–Core50	5	3	50	413	40	200	321	4,352	177	19.6	0.9	171.71
MP1-40–Core50	5	5	50	413	40	200	75	4,005	44	16.0	0.0	136.73
MP1-40–Core75	3	3	75	918	40	200	31	859	16	23.2	0.0	55.1
MP1-40–Core75	3	5	75	918	40	200	216	2,690	116	20.6	0.5	98.19
MP1-40–Core75	5	3	75	918	40	200	6	628	1	23.0	0.7	34.36
MP1-40–Core75	5	5	75	918	40	200	68	1,400	0	20.6	0.3	93.18
MP1-40–Core100	3	3	100	1,685	40	200	7,941	6,259	11,069	21.3	1.0	720.3
MP1-40–Core100	3	5	100	1,685	40	200	2,641	10,308	3,899	17.4	4.0	1,000.02
MP1-40–Core100	5	3	100	1,685	40	200	254	2,580	39	21.2	2.3	1,001.3
MP1-40–Core100	5	5	100	1,685	40	200	231	5,113	8	17.3	0.3	820.9

Table 5.2: Results of our algorithm on the *mq* instances

Instance	H	λ	$\|S\|$	$\|E_S\|$	$\|F\|$	$\|D\|$	$B\&B$	#ConnCuts	#BendCuts	Root-Gap [%]	Final-Gap [%]	**Time(s)**
MQ1-30–Core50	3	3	50	413	30	300	291	4,569	166	23.0	0.0	224.33
MQ1-30–Core50	3	5	50	413	30	300	155	3,672	51	19.4	0.9	142.95
MQ1-30–Core50	5	3	50	413	30	300	111	2,362	33	23.8	0.0	124.15
MQ1-30–Core50	5	5	50	413	30	300	284	4,298	162	21.2	0.4	913.68
MQ1-30–Core75	3	3	75	918	30	300	555	4,966	519	25.1	1.0	219.78
MQ1-30–Core75	3	5	75	918	30	300	211	3,252	77	23.0	0.4	307.45
MQ1-30–Core75	5	3	75	918	30	300	853	3,311	1,076	25.1	0.8	350.62
MQ1-30–Core75	5	5	75	918	30	300	161	2,937	26	23.0	0.8	212.61
MQ1-30–Core100	3	3	100	1,685	30	300	311	4,449	157	24.5	0.9	281.38
MQ1-30–Core100	3	5	100	1,685	30	300	1,531	13,592	832	22.1	0.0	838.07
MQ1-30–Core100	5	3	100	1,685	30	300	129	2,265	24	24.5	4.7	1,000.08
MQ1-30–Core100	5	5	100	1,685	30	300	319	5,397	35	22.0	1.0	434.22
MQ1-40–Core50	3	3	50	413	40	300	201	3,458	105	23.0	0.6	162.33
MQ1-40–Core50	3	5	50	413	40	300	407	6,266	204	20.2	0.2	270.91
MQ1-40–Core50	5	3	50	413	40	300	145	3,104	42	22.9	1.0	574.04
MQ1-40–Core50	5	5	50	413	40	300	300	6,042	122	20.3	0.9	385.3
MQ1-40–Core75	3	3	75	918	40	300	256	5,037	95	23.2	0.6	255.64
MQ1-40–Core75	3	5	75	918	40	300	614	7,896	259	21.1	0.3	318.55
MQ1-40–Core75	5	3	75	918	40	300	581	5,468	246	23.3	1.0	540.86
MQ1-40–Core75	5	5	75	918	40	300	361	5,704	144	21.1	0.9	376.5
MQ1-40–Core100	3	3	100	1,685	40	300	565	8,753	109	23.0	0.3	556.8
MQ1-40–Core100	3	5	100	1,685	40	300	822	11,334	117	19.9	14.6	1,000.02
MQ1-40–Core100	5	3	100	1,685	40	300	492	6,686	38	22.9	0.9	879.14
MQ1-40–Core100	5	5	100	1,685	40	300	563	8,020	16	20.1	2.3	1,000.11

5.6 Conclusion

In this chapter we considered the survivable hop constrained connected facility location problem. We proposed two strong extended IP formulations for the problem. To make the stronger model (with huge number of flow variables) computationally tractable, we applied Benders decomposition approach, projecting out the extended flow variables, and generated Benders cuts within a branch-and-cut framework. We could speed up the algorithm by first separating connectivity inequalities stemming from the corresponding problem with connectivity constraints but without hop constraints, and only solving the computationally expensive separation problem for Benders cuts if the separation of connectivity inequalities fails. We empirically observed that connectivity cuts generated early are very important to reduce the solving time by reducing the number of Benders cuts generated by the algorithm. We evaluate the algorithm on a set of benchmark instances. The computational results show that our algorithm is able to solve most of the instances within a reasonable time to (nearly) optimality.

It is an interesting open question whether our algorithm can be extended such that it can also guarantee λ arc-disjoint H-bounded paths between all pairs of facilities and not only between the root and each facility. We should note that the root-facility connectivity arises in metro networks, where a special *backhaul* node (that provides connectivity to the backbone) exists, while the facility-facility connectivity is more relevant in backbone networks.

Bibliography

Achterberg, T. (2009). Scip: solving constraint integer programs. *Mathematical Programming Computation*, 1(1):1–41.

Agrawal, A., Klein, P., and Ravi, R. (1995). When trees collide: An approximation algorithm for the generalized steiner problem on networks. *SIAM Journal on Computing*, 24(3):440–456.

Ahmadian, S. and Swamy, C. (2012). Improved approximation guarantees for lower-bounded facility location. *Approximation and Online Algorithms*, pages 257–271.

Álvarez-Miranda, E., Cacchiani, V., Lodi, A., Parriani, T., and Schmidt, D. R. (2014). Single-commodity robust network design problem: Complexity, instances and heuristic solutions. *European Journal of Operational Research*.

Andrews, M. (2004). Hardness of buy-at-bulk network design. *Foundations of Computer Science, 2004. Proceedings. 45th Annual IEEE Symposium on*, pages 115–124.

Arulselvan, A., Rezapour, M., and Welz, W. A. (2014). A branch-cut-and-price algorithm for a combined buy-at-bulk network design facility location problem. *Matheon Preprint 1057, Technische Universität Berlin*.

Awerbuch, B. and Azar, Y. (1997). Buy-at-bulk network design. *Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on*, pages 542–547.

Barabási, A.-L. and Albert, R. (1999). Emergence of scaling in random networks. *science*, 286(5439):509–512.

Bardossy, M. G. and Raghavan, S. (2010). Dual-based local search for the connected facility location and related problems. *INFORMS Journal on Computing*, 22(4):584–602.

Barnhart, C., Hane, C. A., and Vance, P. H. (2000). Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Operations Research*, 48(2):318–326.

Bartal, Y. (1996). Probabilistic approximation of metric spaces and its algorithmic applications. *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, pages 184–193.

Bartal, Y. (1998). On approximating arbitrary metrices by tree metrics. *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 161–168.

Benders, J. F. (1962). Partitioning procedures for solving mixed-variables programming problems. *Numerische mathematik*, 4(1):238–252.

Bern, M. and Plassmann, P. (1989). The steiner problem with edge lengths 1 and 2. *Information Processing Letters*, 32(4):171–176.

Bley, A., Hashemi, S. M., and Rezapour, M. (2013a). Approximation algorithms for a combined facility location buy-at-bulk network design problem. *In Proceedings of The 10th Conference on Theory and Applications of Models of Computation-TAMC 2013, Lecture Notes in Computer Science Vol. 7876*, pages 72–83.

Bley, A., Hashemi, S. M., and Rezapour, M. (2013b). IP modeling of the survivable hop constrained connected facility location problem. *In Proceedings of INOC 2013; Electronic Notes in Discrete Mathematics*, 41:463–470.

Bley, A. and Neto, J. (2010). Approximability of 3-and 4-hop bounded disjoint paths problems. *Integer Programming and Combinatorial Optimization*, pages 205–218.

Bley, A. and Rezapour, M. (2013). Approximating connected facility location with buy-at-bulk edge costs via random sampling. *Electronic Notes in Discrete Mathematics*, 44:313–319.

Botton, Q., Fortz, B., Gouveia, L., and Poss, M. (2013). Benders decomposition for the hop-constrained survivable network design problem. *INFORMS journal on computing*, 25(1):13–26.

Byrka, J. and Aardal, K. (2010). An optimal bifactor approximation algorithm for the metric uncapacitated facility location problem. *SIAM Journal on Computing*, 39(6):2212–2231.

Byrka, J., Grandoni, F., Rothvoß, T., and Sanità, L. (2010). An improved lp-based approximation for steiner tree. *Proceedings of the 42nd ACM symposium on Theory of computing*, pages 583–592.

Cacchiani, V., Jünger, M., Liers, F., Lodi, A., and Schmidt, D. R. (2014). Single-commodity robust network design with finite and hose demand sets. *Technical Report OR-14-11, University of Bologna.*

Charikar, M. and Guha, S. (1999). Improved combinatorial algorithms for the facility location and k-median problems. *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pages 378–388.

Charikar, M. and Karagiozova, A. (2005). On non-uniform multicommodity buy-at-bulk network design. *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 176–182.

Chekuri, C., Hajiaghayi, M. T., Kortsarz, G., and Salavatipour, M. R. (2010). Approximation algorithms for nonuniform buy-at-bulk network design. *SIAM Journal on Computing*, 39(5):1772–1798.

Chekuri, C., Khanna, S., and Naor, J. S. (2001). A deterministic algorithm for the cost-distance problem. *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, pages 232–233.

Chlebík, M. and Chlebíková, J. (2008). The steiner tree problem on graphs: Inapproximability results. *Theoretical Computer Science*, 406(3):207–214.

Chudak, F. A. and Shmoys, D. B. (2003). Improved approximation algorithms for the uncapacitated facility location problem. *SIAM Journal on Computing*, 33(1):1–25.

Chuzhoy, J., Gupta, A., Naor, J. S., and Sinha, A. (2008). On the approximability of some network design problems. *ACM Transactions on Algorithms (TALG)*, 4(2):23.

Chvatal, V. (1979). A greedy heuristic for the set-covering problem. *Mathematics of operations research*, 4(3):233–235.

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

Eisenbrand, F., Grandoni, F., Rothvoß, T., and Schäfer, G. (2010). Connected facility location via random facility sampling and core detouring. *Journal of Computer and System Sciences*, 76(8):709–726.

Fakcharoenphol, J., Rao, S., and Talwar, K. (2003). A tight bound on approximating arbitrary metrics by tree metrics. *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 448–455.

Fleischer, L., Jain, K., and Williamson, D. P. (2006). Iterative rounding 2-approximation algorithms for minimum-cost vertex connectivity problems. *Journal of Computer and System Sciences*, 72(5):838–867.

Friggstad, Z., Rezapour, M., Salavatipour, M. R., and Soto, J. A. (2015). LP-based approximation algorithms for facility location in buy-at-bulk network design. *To appear in Proceedings of the 14th International Symposium on Algorithms and Data Structures-WADS 2015*.

FTT. Fttx-plan. fttx-plan: Kostenoptimierte planung von fttx-netzen. http://www.fttx-plan.de/.

Garey, M. and Johnson, D. (1979). Computers and intractability: a guide to the theory of np-completeness. *WH Freeman & Co., San Francisco*.

Garg, N., Khandekar, R., Konjevod, G., Ravi, R., Salman, F. S., and Sinha, A. (2001). On the integrality gap of a natural formulation of the single-sink buy-at-bulk network design problem. *Integer Programming and Combinatorial Optimization*, pages 170–184.

Goemans, M. X., Goldberg, A. V., Plotkin, S. A., Shmoys, D. B., Tardos, E., and Williamson, D. P. (1994). Improved approximation algorithms for network design problems. *SODA*, 94:223–232.

Gollowitzer, S. and Ljubić, I. (2011). Mip models for connected facility location: A theoretical and computational study. *Computers & Operations Research*, 38(2):435–449.

Gouveia, L., Simonetti, L., and Uchoa, E. (2011). Modeling hop-constrained and diameter-constrained minimum spanning tree problems as steiner tree problems over layered graphs. *Mathematical Programming*, 128(1-2):123–148.

Grandoni, F. and Rothvoß, T. (2010). Network design via core detouring for problems without a core. *Automata, Languages and Programming*, pages 490–502.

Grandoni, F. and Rothvoß, T. (2011). Approximation algorithms for single and multi-commodity connected facility location. *Integer Programming and Combinatoral Optimization*, pages 248–260.

Guha, S. and Khuller, S. (1999). Greedy strikes back: Improved facility location algorithms. *Journal of algorithms*, 31(1):228–248.

Guha, S., Meyerson, A., and Munagala, K. (2000). Hierarchical placement and network design problems. *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 603–612.

Guha, S., Meyerson, A., and Munagala, K. (2001). A constant factor approximation for the single sink edge installation problems. *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 383–388.

Guha, S., Meyerson, A., and Munagala, K. (2009). A constant factor approximation for the single sink edge installation problem. *SIAM Journal on Computing*, 38(6):2426–2442.

Gupta, A., Kleinberg, J., Kumar, A., Rastogi, R., and Yener, B. (2001). Provisioning a virtual private network: a network design problem for multicommodity flow. *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 389–398.

Gupta, A., Kumar, A., and Roughgarden, T. (2003). Simpler and better approximation algorithms for network design. *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 365–372.

Hasan, M. K., Jung, H., and Chwa, K.-Y. (2008). Approximation algorithms for connected facility location problems. *Journal of Combinatorial Optimization*, 16(2):155–172.

Hochbaum, D. S. (1982). Heuristics for the fixed cost median problem. *Mathematical programming*, 22(1):148–162.

Hromkovic, J. (2010). Algorithmics for hard problems: introduction to combinatorial optimization, randomization, approximation, and heuristics. *Springer-Verlag*.

Huygens, D., Labbé, M., Mahjoub, A. R., and Pesneau, P. (2007). The two-edge connected hop-constrained network design problem: Valid inequalities and branch-and-cut. *Networks*, 49(1):116–133.

IBM (2013). Ibm ilog cplex optimization studio v12.6.

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

Jain, K., Mahdian, M., Markakis, E., Saberi, A., and Vazirani, V. V. (2003). Greedy facility location algorithms analyzed using dual fitting with factor-revealing lp. *Journal of the ACM (JACM)*, 50(6):795–824.

Jain, K., Mahdian, M., and Saberi, A. (2002). A new greedy approach for facility location problems. *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 731–740.

Jain, K. and Vazirani, V. V. (2001). Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and lagrangian relaxation. *Journal of the ACM (JACM)*, 48(2):274–296.

Johnson, D. S., Minkoff, M., and Phillips, S. (2000). The prize collecting steiner tree problem: theory and practice. *SODA*, 1(0.6):4.

Jothi, R. and Raghavachari, B. (2004). Improved approximation algorithms for the single-sink buy-at-bulk network design problems. *Algorithm Theory-SWAT 2004*, pages 336–348.

Jung, H., Hasan, M. K., and Chwa, K.-Y. (2009). A 6.55 factor primal-dual approximation algorithm for the connected facility location problem. *Journal of combinatorial optimization*, 18(3):258–271.

Karget, D. and Minkoff, M. (2000). Building steiner trees with incomplete global knowledge. *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 613–623.

Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302–311.

Khuller, S., Raghavachari, B., and Young, N. (1995). Balancing minimum spanning trees and shortest-path trees. *Algorithmica*, 14(4):305–321.

Kortsarz, G. and Nutov, Z. (2009). Approximating some network design problems with node costs. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 231–243.

Li, S. (2013). A 1.488 approximation algorithm for the uncapacitated facility location problem. *Information and Computation*, 222:45–58.

Lin, J.-H. and Vitter, J. S. (1992). Approximation algorithms for geometric median problems. *Information Processing Letters*, 44(5):245–249.

Ljubić, I. (2007). A hybrid vns for connected facility location. *Hybrid Metaheuristics*, pages 157–169.

Ljubić, I. and Gollowitzer, S. (2013). Layered graph approaches to the hop constrained connected facility location problem. *INFORMS Journal on Computing*, 25(2):256–270.

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

Lübbecke, M. E. and Desrosiers, J. (2005). Selected topics in column generation. *Operations Research*, 53(6):1007–1023.

Mahdian, M., Ye, Y., and Zhang, J. (2006). Approximation algorithms for metric facility location problems. *SIAM Journal on Computing*, 36(2):411–432.

Mahjoub, A. R., Simonetti, L., and Uchoa, E. (2013). Hop-level flow formulation for the survivable network design with hop constraints problem. *Networks*, 61(2):171–179.

Manyem, P. (2009). Constrained spanning, steiner trees and the triangle inequality. *Optimization*, pages 355–367.

Melkote, S. and Daskin, M. S. (2001). An integrated model of facility location and transportation network design. *Transportation Research Part A: Policy and Practice*, 35(6):515–538.

Meyerson, A., Munagala, K., and Plotkin, S. (2000). Cost-distance: Two metric network design. *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 624–630.

Raghavan, S. and Stanojević, D. (2006). A note on search by objective relaxation. *Telecommunications planning: innovations in pricing, network design and management*, pages 181–201.

Randazzo, C., Luna, H. P. L., Mahey, P., et al. (2001). Benders decomposition for local access network design with two technologies. *Discrete Mathematics & Theoretical Computer Science*, 4(2):235–246.

Ravi, R. and Sinha, A. (2006). Approximation algorithms for problems combining facility location and network design. *Operations Research*, 54(1):73–81.

Robins, G. and Zelikovsky, A. (2000). Improved steiner tree approximation in graphs. *Proceedings of the eleventh annual ACM-SIAM symposium on Discrete algorithms*, pages 770–779.

Salman, F. S., Ravi, R., and Hooker, J. N. (2008). Solving the capacitated local access network design problem. *INFORMS Journal on Computing*, 20(2):243–254.

Schrijver, A. (2003). Combinatorial optimization: polyhedra and efficiency. *Springer*, 24.

Shmoys, D. B., Tardos, É., and Aardal, K. (1997). Approximation algorithms for facility location problems. *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 265–274.

Sviridenko, M. (2002). An improved approximation algorithm for the metric uncapacitated facility location problem. *Integer programming and combinatorial optimization*, pages 240–257.

Svitkina, Z. (2010). Lower-bounded facility location. *ACM Transactions on Algorithms (TALG)*, 6(4):69.

Swamy, C. and Kumar, A. (2004). Primal–dual algorithms for connected facility location problems. *Algorithmica*, 40(4):245–269.

Takahashi, H. and Matsuyama, A. (1980). An approximate solution for the steiner problem in graphs. *Math. Japonica*, 24(6):573–577.

Talwar, K. (2002). The single-sink buy-at-bulk lp has constant integrality gap. *Integer Programming and Combinatorial Optimization*, pages 475–486.

Tarjan, R. E. (1974). A note on finding the bridges of a graph. *Information Processing Letters*, 2(6):160–161.

Tomazic, A. and Ljubić, I. (2008). A grasp algorithm for the connected facility location problem. *Applications and the Internet, 2008. SAINT 2008. International Symposium on*, pages 257–260.

Vazirani, V. V. (2001). Approximation algorithms. *springer*.

Vygen, J. (2005). Approximation algorithms facility location problems. *Report No. 05950-OR, Research Institute for Discrete Mathematics, University of Bonn*.

Williamson, D. P., Goemans, M. X., Mihail, M., and Vazirani, V. V. (1993). A primal-dual approximation algorithm for generalized steiner network problems. *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, pages 708–717.

Williamson, D. P., Goemans, M. X., Mihail, M., and Vazirani, V. V. (1995). A primal-dual approximation algorithm for generalized steiner network problems. *Combinatorica*, 15(3):435–454.

Williamson, D. P. and Shmoys, D. B. (2011). The design of approximation algorithms. *Cambridge University Press*.

Wolsey, L. A. (1998). Integer programming. *Wiley New York*, 42.

Abbreviations

IP	**I**nteger **P**rogramming
UFL	**U**ncapacitated **F**acility **L**ocation
ST	**S**teiner **T**ree
ConFL	**Con**nected **F**acility **L**ocation
SSRoB	**S**ingle **S**ink **R**ent **o**r **B**uy
SND	**S**urvivable **N**etwork **D**esign
rSND	**r**ooted **S**urvivable **N**etwork **D**esign
SHND	**S**urvivable **H**op-constrained **N**etwork **D**esign
BBND	**B**uy-at**B**ulk **N**etwork **D**esign
SSBB	**S**ingle-**S**ink **B**uy-at**B**ulk Network Design
BBConFL	**B**uy-at**B**ulk **Con**nected **F**acility **L**ocation
BBFL	**B**uy-at**B**ulk **F**acility **L**ocation
DDConFL	**D**eep-**D**iscount **Con**nected **F**acility **L**ocation
DDFL	**D**eep-**D**iscount **F**acility **L**ocation
SHConFL	**S**urvivable **H**op-constrained **Con**nected **F**acility **L**ocation