Discrete and Robust Optimization Approaches to Network Design with Compression and Virtual Network Embedding

Von der Fakultät für Mathematik, Informatik und Naturwissenschaften der RWTH Aachen University zur Erlangung des akademischen Grades eines Doktors der Naturwissenschaften genehmigte Dissertation

vorgelegt von

Master of Science RWTH Aachen University

Martin Tieves

aus Neuss

Berichter: Univ.-Prof. Dr. Ir. Arie M. C. A. Koster
Prof. Dr. Edoardo Amaldi

Tag der mündlichen Prüfung: 16. Dezember 2016

Abstract

In this thesis, we study two optimization problems, the Network Design Problem with Compression (NDPC) and the Virtual Network Embedding Problem (VNE). In both cases, our interest into the topic is motivated by the importance of these problems within the telecommunication industry, where they arise in the context of introducing new services and technologies.

Throughout this work, we employ concepts and methods from the area of mathematical, respectively combinatorial, optimization. We aim to provide new insights, both from a theoretical and from a practical point of view. For that purpose, we carry out extensive computational experiments to strengthen our theoretical results. Wherever possible, we put our results into context with the existing literature.

We follow a similar line of thought for both problems. For the NDPC problem, we present a mixed integer linear programming (MILP) formulation, detailed polyhedral investigations, and considerations on the problems computational complexity as well as a discussion on the problem under data uncertainty. We conclude our work on NDPC by computational results and an outlook into further research directions.

For the VNE problem, we also start with an MILP formulation. We discuss heuristic problem approaches and investigate the problem's computational complexity in great detail. We consider the VNE problem with data uncertainty and develop exact and heuristic solution approaches for this case. As for the NDPC problem, we present extensive computational experiments to evaluate our results. The chapter is closed by a short summary and a brief introduction to future research topics.

We conclude this thesis by a final overview on the here presented results and with some final remarks.

Zusammenfassung

Den Schwerpunkt der hier vorliegenden Dissertation bildet die mathematische Untersuchung zweier Optimierungsprobleme aus dem Telekommunikationssektor. Das Erste betrifft die Dimensionierung von Kommunikationsnetzen wenn die Möglichkeit besteht Datenströme zu komprimieren (NDPC). Das Zweite entsteht bei der Einbettung von virtuellen Kommunikationsnetzen in gegebene Substrat-Netzwerke (VNE). Beide Probleme sind insbesondere für die Telekommunikationsindustrie relevant. Unter anderem treten sie dort bei der Betrachtung von Technologien zur Einführung neuer Dienstleistungen beziehungsweise von Serviceverbesserungen auf.

In dieser Arbeit verwenden wir hauptsächlich die Methoden und Konzepte der mathematischen beziehungsweise der kombinatorischen Optimierung. Das Ziel dieser Arbeit ist es neue Einsichten in die Thematik sowohl aus praktischer als auch aus theoretischer Perspektive zu erarbeiten. Wir präsentieren ausführliche Rechenstudien um unsere theoretischen Ergbenisse zu unterstützen. Wenn immer es möglich ist ordnen wir unsere Resultate in den Kontext der bereits existierenden Literatur ein.

Für beide Probleme verfolgen wir einen ähnlichen Ansatz. Für das NDPC Problem präsentieren wir eine Formulierung als gemischt ganzzahliges, lineares Programm (MILP), detailierte Untersuchungen bezüglich des davon induzieren Polyeders und Betrachtungen zur Berechnungskomplexität sowie zur Unsicherheit von Eingabedaten. Wir schließen dieses Kapitel mit einer Diskussion unserer Rechenstudien ab und geben eine kurze Zusammenfassung der zum NDPC erzielten Resultate sowie einen Ausblick auf weitere Forschungsmöglichkeiten.

Auch für das VNE Problem untersuchen wir zunächst eine Formulierung als MILP. Im Folgenden diskutieren wir heuristische Lösungsansätze und untersuchen die Berechnungskomplexität des VNE Problems. Wir betrachten das VNE Problem unter Datenunsicherheit und entwickeln exakte und heuristische Lösungsverfahren für diesen Fall. Wie für das NDPC Problem diskutieren wir zuletzt die Ergebnisse unserer Rechenstudien und geben einige Bemerkungen über potentielle zukünftige Forschungsrichtungen.

Wir schließen diese Dissertation mit einer Zusammenfassung unserer Ergebnisse sowie mit einigen finalen Bemerkungen.

Danksagung

Die hier vorliegende Dissertation wäre ohne die Mithilfe meiner Freunde und Kollegen und ohne die Unterstützung, die ich erfahren habe, nicht denkbar gewesen. Ich möchte nun die Gelegenheit ergreifen den beteiligten Personen dafür explizit zu danken.

Zuerst möchte ich mich bei meinem Betreuer Herrn Prof. Arie M.C.A. Koster bedanken. Er gab mir die Gelegenheit hier am Lehrstuhl II zu arbeiten und stand mir jederzeit mit Rat und Tat zur Seite. Besonders herausstellen möchte ich sein Bemühen mich in die wissenschaftliche Gemeinschaft einzugliedern. Für die vielen Möglichkeiten an Konferenzreisen teilzunehmen und "über meinen Tellerrand hinaus zu schauen" bin ich ausgesprochen dankbar.

Weiter bedanke ich mich bei Herrn Prof. Edoardo Amaldi sowohl für seine Bereitschaft als zweiter Berichter meine Dissertation zu Begutachten als auch für das gemeinschaftliche Arbeiten im Rahmen gemeinsamer Publikationen.

Ich bedanke mich bei meinen Büro-Kollegen, bei Sebastian Goderbauer, bei Nils Spiekermann und bei Stefano Coniglio. In unserem Büro herrschte durchweg eine angenehme Atmosphäre, ich habe sowohl die gemeinsame Arbeit als auch die gemeinsam verbrachte Zeit genossen.

Ebenso möchte ich mich bei Herrn Prof. Eberhard Triesch und allen aktuellen und ehemaligen Kollegen des Lehrstuhls für die Jahre des gemeinsamen Arbeitens bedanken.

Im Hinblick auf diese Dissertation bedanke ich mich bei Martin Comis, Prof. Arie M.C.A. Koster, Clara Nadenau und Nils Spiekermann für das Korrekturlesen meiner Arbeit, sie ersparten mir so manche Peinlichkeit.

Schlussendlich möchte ich meine Familie nicht unerwähnt lassen. Ich kann mir keine stärkere Unterstützung vorstellen als ich sie durch mein Elternhaus erfahren habe. Zu meiner Familie zähle ich auch meine Lebenspartnerin Clara. Ich schätze mich glücklich sie an meiner Seite zu wissen.

Ich widme diese Arbeit meinen Großvätern.

Aachen, im September 2016 Martin Tieves

Contents

Introduction

Communication systems are a cornerstone of our modern society. On a daily basis, many people rely on such systems, either directly and deliberately, or indirectly and obliviously. Consider, for instance, a mobile telephone call (active reliance) or simply the usage of GPS-tracking devices, e.g., for a car (passive reliance). In both cases, directly or indirectly, data is created and transferred somewhere for some purpose. The infrastructure which allows for this process is what we understand as a communication network. One particularly prominent example of a communication network is given by the fiber and copper lines inter-connecting computers and routers and thus forming the Internet.

Apparently, the amount, the importance, and the scale of these communication systems drastically increases nowadays. As a consequence, strategic long term decision making is one of the critical tasks in a networking environment as to enable data exchange, respectively as to sustain operability of these networks. While, in general, "strategic decision making" encompasses a multitude of different aspects, in this work, we focus on decisions concerning the dimensioning (layout) and the usage of these infrastructures.

To do so, we rely on the tools provided by "mathematical optimization", being one particular discipline of mathematics supporting such decision processes. Mathematical optimization provides concepts and tools to at first, abstract decision processes, and then, model them as optimization problems, and finally, obtain solutions for the resulting problems. In turn, these solutions describe the optimal decisions in which we were interested in, in the first place. One well-known example for such optimization problem is the Network Design Problem (NDP), which deals with one of the most fundamental decisions in a telecommunication environment, namely the question of how to cost effectively lay out a capacity constrained network so as to support a given set of communications.

In this thesis, we consider two specific optimization problems which arise in the context of telecommunication systems and which both generalize on the Network Design Problem. The first problem which we will consider is the Network Design Problem with Compression (NDPC). The second problem which we will investigate is the Virtual Network Embedding Problem (VNE). The former extends the Network Design Problem by allowing data streams to be compressed, thus effectively reducing their capacity requirements. The latter exploits the question of how many services, say virtual networks, can jointly be sustained by a given infrastructure. We will tackle both problems

using tools from "mathematical optimization". For both problems, we present formal definitions and formulations as mixed integer linear programs (MILPs). We will explore the computational complexity of the problems, and we will show how data uncertainty can be accounted for. For the NDPC problem, we will focus on polyhedral aspects with respect to it's MILP formulation while for VNE, the emphasis is on heuristic solution approaches. Our results are supported by extensive computational experiments, indicating their practical relevance.

We point out that the here considered Network Design with Compression Problem as well as the Virtual Network Embedding Problem arise in the telecommunication industry, where they are very relevant for enabling future (Internet) service technologies. Thus, this work contributes to "mathematics for innovation in industry and service" and is hence part of the federal high-tech strategy for strengthening the information and communication technology in Germany.

Acknowledgement This work was supported by the BMBF grant 05M13PAA, joint project 05M2013 - VINO: Virtual Network Optimization [122], by the German Federal Ministry of Education and Research, and was developed in collaboration with the industry partners ADVA Optical Networking SE and Deutsche Telekom AG – Innovation Laboratories.

Contributions We present a collection of results and insights which have been developed over the last few years. As it is common in the area of applied mathematical optimization, parts of this work originated from various collaborations and subsequently, partial results have been presented at numerous conferences, and respectively have been published before. In particular, this concerns the works [5, 41–43, 84, 86]. At the beginning of each chapter, we will explicitly state which results have been published before, also referring to our co-authors in this context. We list the main contributions of this thesis:

For the Network Design with Compression Problem:

- An improved mixed integer linear programming (MILP) formulation and an extensive analysis of the corresponding polyhedral region.
- An analysis of the the problem's computational complexity, with emphasis on the Compressor Placement Problem as the subproblem which describes the additional elements of NDPC with respect to the Network Design Problem. This includes a discussion on special cases, for which pseudo-polynomial algorithms are presented.
- A discussion on how data uncertainty can be taken into account, in particular relying on the two concepts Γ-robustness and two-source robustness.
- A detailed computational evaluation of the previously mentioned results, including a comparison between NDPC and Network Design, an evaluation of the polyhedral results, and investigations of the robust problems.

For the Virtual Network Embedding Problem:

- A structured and extendable MILP formulation which can also be applied to other problem variations and which directly induces heuristic solution approaches.
- An extensive analysis of the computational complexity of VNE, including hardness results for special cases and a review of dynamic programming approaches.
- A discussion on how data uncertainty can be incorporated into the problem. Relying on Γ-robustness, exact and heuristic solution methods are presented.
- A computational evaluation of the deterministic problem and of the one under data uncertainty, based on the here presented exact and heuristic solutions methods.

Outline This thesis is composed of four chapters.

In the *first chapter – Basic Concepts: Flows, Network Design and Data Uncertainty*, we give an introduction to the ideas and concepts providing the methodological background of this thesis. In particular, in Section 1.1, we start by discussing how basic telecommunication systems, e.g., traffic flows in telecommunication networks, can be modeled within the field of mathematical optimization. Therefore, we recapitulate some basic notation and a selection of the most fundamental results in this field. Based on the notion of flows, we introduce the maximum flow problem as a basic building block to model communications, e.g., the transmission of communication signals, respectively of data, in a telecommunication network. Subsequently, we extend this problem by introducing the single- and the multi-commodity flow problem and, finally, the Network Design Problem in Section 1.2. After some short side remarks on variations and extension of the Network Design Problem in Section 1.3, we conclude the chapter by an introduction on data uncertainty in mathematical optimization in Section 1.4. In particular, we introduce a general framework for data uncertainty which allows to incorporate such uncertainty within the optimization problems. We show how Γ-robustness can be derived as a special case and define the two-source robustness which is a robustness concept, especially tailored to the NDPC problem.

In the *second chapter – Network Design with Compression*, we focus on the NDPC problem. We start with an introduction and a motivation of the problem and then, present a formal definition and a formulation as MILP in Section 2.1 and in Section 2.2 respectively. Based on this, we investigate the polyhedral region of this formulation, focusing on cutting planes as, for example, Cutset Inequalities in Section 2.3. In the next section, we analyze the computational complexity of NDPC. Therefore, we introduce the Compressor Placement Problem (CPP) as a restriction of the NDPC problem to illustrate the differences between Network Design and Network Design with Compression. We present hardness results as well as a (pseudo-) polynomial solvable case for the CPP, respectively for the NDPC problem. In Section 2.5, we discuss how data uncertainty can be tackled in NDPC by applying the Γ-robustness and the two-source robustness

to the MILP formulation introduced before. Finally, in Section 2.6, we present an evaluation of our computational results. We show how NDPC relates to the Network Design Problem, how cutting planes can be employed to improve the linear relaxation of the MILP formulation, and we discuss the results obtained by the different robust formulations. The chapter is concluded by a review of our results and an indication of further research directions for the NDPC problem.

In the *third chapter – Virtual Network Embedding*, we focus on the VNE problem. At first, in Section 3.1, we give an introduction to VNE, motivating the importance of VNE with regard to large scale telecommunication services. In Section 3.2, we present a formal definition of VNE, which is subsequently extended to an MILP formulation. We discuss how such formulation can be modified so to include "rent-at-bulk" aspects and demonstrate how this formulation naturally induces heuristic solution approaches. In the following, in Section 3.3, we analyze the computational complexity of VNE. We present hardness results on the general problem and on special cases, where we fix a dimension of the problem. We conclude by focusing on dynamic programming approaches and on the special case where the virtual networks are isomorphic to stars. In Section 3.4, we investigate the VNE problem under data uncertainty. Starting from a chance constrained formulation, we show how this problem can be tackled by applying Γ-robustness to its MILP formulation. Based on the same approach, we also derive heuristic solution approaches to the problem with data uncertainty. Finally, in Section 3.5, we present a computational evaluation of the VNE problem. We consider both, the deterministic problem and the one under data uncertainty and evaluate the here presented exact and heuristic solution approaches. We wrap the chapter up with a summary of our results and an outlook into further research directions for the VNE problem.

The thesis is concluded by a brief discussion of the contributions of this work and the there-in contained results in *Chapter 4*. To this end, we provide a short discussion on possible extension of this work, as well as mentioning some open questions.

CHAPTER 1

Basic concepts: flows, network design, and data uncertainty

In this thesis, we focus on two mathematical optimization problems which arise in the telecommunication industry. We will address the specific problems in Chapter 2 and in Chapter 3. In the current chapter, we give a brief introduction to the most important concepts required to model these problems within the field of mathematical optimization. Therefore, we give a brief recapitulation of some basic notation and a selection of the most fundamental results in this field.

We start with the notion of network flows and introduce the maximum flow problem as a basic building block to model communications in a telecommunication network, given a fixed communication infrastructure. Subsequently, we extend this problem by introducing the single- and the multi-commodity flow problem.

In the second section, we extend these problems to the network design problem. This problem also takes the dimensioning of the infrastructure into account, for example, by interpreting the available bandwidth of a link as capacity. In the following, we discuss further extensions of the network design problem, also exploring some special cases which we will extend on in the course of this work.

We conclude the chapter by discussing mathematical concepts to deal with data uncertainty, a topic which is very relevant for applying optimization tools to real world problems. We provide a short introduction on robust optimization and present a general framework to model data uncertainty in optimization problems. Subsequently, we show how the Γ-robustness can be derived from this framework as a special case and derive a new concept of uncertainty, called bijective, respectively two-source, uncertainty.

1.1 Network flows

The following topics are well established in the area of mathematical optimization. Anything but presenting the most basic ideas is beyond the scope of this work. We refer the reader to the excellent monographs by Schrijver [114] and by Ahuja et al. [3] for additional information and further references.

For the mathematical abstraction of telecommunication problems, one of the fundamental concepts is that of a *flow* .Let $G = (V, A)$ be a directed graph with node set V and arc set A, where each arc $uv \in A$ is endowed with a capacity $k_{uv} \in \mathbb{R}_+$. A flow is a function, assigning real values to the arcs of a given graph, indicating the arc-wise movement, e.g., of a data volume, from a source node to a sink node. In this context, we often refer to such flow as "traffic"- or "data"-flow or, even shorter, as *traffic*. We formalize this as follows: Let $s \in V$ be the so-called source and let $t \in V$ be the sink. A flow is a function $f : A \to \mathbb{R}_+, uv \mapsto f_{uv}$ with the two properties: *i*) f obeys the capacity restrictions, i.e., $f_{uv} \leq k_{uv}$ for all arcs $uv \in A$. *ii*) For any node $u \in V \setminus \{s, t\}$, the amount of incoming flow is equal to the amount of outgoing flow, i.e., $\sum_{uv \in A} f_{vu} = \sum_{vu \in A} f_{vu}$. We derive a formal definition:

Definition 1.1 (Flow)**.** *Given a directed graph $G = (V, A)$, $s, t \in V$, and a capacity function $k : A \to \mathbb{R}_+, uv \mapsto k_{uv}$, a **flow** is a function*

$$f : A \to \mathbb{R}_+, uv \mapsto f_{uv} \tag{1.1a}$$

with the properties that

$$f_{uv} \leq k_{uv} \qquad \forall uv \in A, \tag{1.1b}$$

$$\sum_{uv \in A} f_{uv} = \sum_{vu \in A} f_{vu} \qquad \forall u \in V \setminus \{s, t\}\,. \tag{1.1c}$$

The value of f is denoted as $val(f) = \sum_{vt \in A} f_{vt}$.

We say that a flow f transports a *commodity* and refer to f_{uv} as the *flow* on arc $uv \in A$. We say that G permits such flow and we refer to Condition (1.1c) as *flow conservation* or *flow-balance* constraints.

A flow can be interpreted as a data volume that is traveling through a network. It can be used, for example, to address questions about the maximal amount of data that can travel across the network, i.e., "How much data can be sent from a source node s to a sink node t?". In the context of mathematical optimization, we refer to such questions as *problems*. The corresponding problem in this case, is the *maximum flow problem*. In this problem, a graph, a source node s, a sink node t, and a capacity function are given and we are looking for a flow f between source and sink, with the additional property *iii*) that its value $val(f)$ is maximized. Formally, this writes as

Definition 1.2 (Maximum Flow Problem)**.** *Given a directed graph $G = (V, A)$, $s, t \in V$, and a capacity function $k : A \to \mathbb{R}_+, uv \mapsto k_{uv}$, the **maximum flow problem** calls for a flow as described in Definition 1.1 with the additional property that it's value $val(f)$ is maximized.*

The maximum flow problem can be solved in polynomial time.

Theorem 1.1 (Ford and Fulkerson [57])**.** *The maximum flow problem is in $\mathcal{P}$.*

There are many variations and generalizations of the standard maximum flow problem. In this context, we mention its feasibility version, which can be formulated as: "Given two specific nodes $s, t \in V$ and a parameter $d \in \mathbb{R}_+$, does G permit a flow f of value d from s to t?". We say that d units of a commodity are available in s (or s outputs d units of the commodity) and that t requests d units of this commodity, which we also refer to as the *demand* (value) of the sink node t. The task is to find a flow between s and t with value d. If such flow exists, we say that the complete demand of t is supported.

A further generalization of a flow is the *(S,T)-flow*. In this problem, instead of a single source s and a single sink t, multiple source nodes $S \subseteq V$ and multiple sink nodes $T \subseteq V, T \cap S = \emptyset$ are given. Then, an (S,T)-flow is a function on the arcs, which obeys the capacity restrictions as in Definition 1.1, but where the flow-balance constraints only apply to all nodes $u \in V$, for which $u \notin S$ and $u \notin T$, respectively where the flow-balance constraints do not apply to all nodes $u \in S$ and $u \in T$. We formalize this in the following definition:

Definition 1.3 ((S,T)-flow)**.** *Given a directed graph $G = (V, A)$, $S \subseteq V$, $T \subseteq V$ with $S \cap T = \emptyset$, and a capacity function $k : A \to \mathbb{R}_+, uv \mapsto k_{uv}$, an **(S,T)-flow** is a function*

$$f : A \to \mathbb{R}_+, uv \mapsto f_{uv} \tag{1.2a}$$

with the properties that

$$f_{uv} \leq k_{uv} \qquad \forall uv \in A, \tag{1.2b}$$

$$\sum_{uv \in A} f_{uv} = \sum_{vu \in A} f_{vu} \qquad \forall u \in V, u \notin S, u \notin T. \tag{1.2c}$$

Similar to the maximum flow problem above, we derive a decision problem based on an (S,T)-flow. The *(S,T)-flow problem* asks: "Given G, S, and T as specified above and given that each node $s \in S$ outputs $d_s \in \mathbb{R}_+$ units of the commodity and each node $t \in T$ requests $r_t \in \mathbb{R}_+$ units of the commodity, does a single commodity flow which fulfills $\sum_{sv \in A} f_{sv} = d_s$ for all $s \in S$ and $\sum_{vt \in A} f_{vt} = r_t$ for all $t \in T$ exist?". Formally, this problem can be stated as:

Definition 1.4 ((S,T)-flow Problem)**.** *Given a directed graph $G = (V, A)$, two sets $S \subseteq V$ and $T \subseteq V$ with $T \cap S = \emptyset$, a supply function d, a demand function r, and a capacity function k, where*

$$d : S \to \mathbb{R}_+, s \mapsto d_s, \tag{1.3a}$$

$$r : T \to \mathbb{R}_+, t \mapsto r_t, \qquad \text{and} \tag{1.3b}$$

$$k : A \to \mathbb{R}_+, uv \mapsto k_{uv}, \tag{1.3c}$$

*the **(S,T)-flow problem** asks whether a flow f exists, as in Definition 1.3, with the additional property that*

$$\sum_{sv \in A} f_{sv} = d_s \quad \forall s \in S \text{ and } \sum_{vt \in A} f_{vt} = r_t \quad \forall t \in T. \tag{1.3d}$$

By adding a super source and a super sink with arcs to all sources and sinks, respectively, and with the corresponding arc capacities, the (S,T)-flow problem can be expressed as decision version of the maximum flow problem.

As a next step, we consider a *multi commodity flow*. In contrast to the (S,T)-flow, it is not necessarily $S \cap T = \emptyset$ and we require that the different flows from the sources to the sinks are not interchanged. Let a graph G, a capacity function k, and a collection Q of pairs $(s,t) \in S \times T$ be given. Henceforth, we refer to such pairs as commodities. We employ the notation $q := (s,t) \in Q$ and write $s^q = s$ and $t^q = t$. A *multi commodity flow* is a vector of flows, where each entry corresponds to a single (s,t) commodity. These flows are independent, except for the linking capacity constraints, i.e., the sum over the components of the multi commodity flow of the flow values associated to the arc uv cannot exceed its capacity k_{uv}. We formalize this as follows.

Definition 1.5 (Multi Commodity Flow)**.** *Given a directed graph $G = (V, A)$, $S \subseteq V$, $T \subseteq V$, a collection Q of pairs $(s,t) \in S \times T$, and a capacity function*

$$k : A \to \mathbb{R}_+, uv \mapsto k_{uv}, \tag{1.4a}$$

a ***multi commodity flow*** *consists of $|Q|$ functions*

$$f^q : A \to \mathbb{R}_+, uv \mapsto f^q_{uv} \qquad \forall q \in Q \tag{1.4b}$$

with the properties that

$$\sum_{q \in Q} f^q_{uv} \leq k_{uv} \qquad \forall uv \in A \tag{1.4c}$$

$$\sum_{uv \in A} f^q_{uv} = \sum_{vu \in A} f^q_{vu} \qquad \forall q \in Q, u \in V : u \neq s^q, u \neq t^q. \tag{1.4d}$$

As for the (S,T)-flow, we introduce demand volumes for the single commodities and obtain a decision problem. The *multi commodity flow problem* asks: "Given G, k, and Q as defined above and $d^q \in \mathbb{R}_+$ for all $q \in Q$, does a multi commodity flow with $\sum_{s^q v \in A} f^q_{s^q v} = d^q = \sum_{v t^q \in A} f^q_{v t^q}$ for all $q \in Q$ exist?". Formally, this can be stated as:

Definition 1.6 (Multi Commodity Flow Problem)**.** *Given a directed graph $G = (V, A)$, $S \subseteq V$, $T \subseteq V$, a collection Q of pairs $(s,t) \in S \times T$, a demand function d, and a capacity function k, where*

$$d : Q \to \mathbb{R}_+, q \ \mapsto d^q \qquad \text{and} \tag{1.5a}$$

$$k : A \to \mathbb{R}_+, uv \mapsto k_{uv}, \tag{1.5b}$$

the ***multi commodity flow problem*** *asks whether a multi commodity flow exists, as in Definition 1.5, with the property that*

$$\sum_{s^q v \in A} f^q_{s^q v} = d^q = \sum_{v t^q \in A} f^q_{v t^q} \qquad \forall q \in Q. \tag{1.5c}$$

This multi commodity flow problem can still be solved in polynomial time.

Theorem 1.2. *The multi commodity flow problem is in $\mathcal{P}$.*

Proof. The problem can be solved by linear programming. □

In contrast to the maximum flow problem or the single commodity flow problem, the multi commodity flow problem allows to model multiple (different), simultaneous communications (commodities) within a single network. Its relevance for telecommunication applications is clear. As an example, consider that G represents a network between some routers. Given that a certain amount of data (traffic) has to be sent between the routers, the multi-commodity flow problem asks whether this can be done simultaneously, respecting the given bandwidth-capacities.

1.2 Network design

In this section, we consider the (capacitated) *Network Design Problem* (NDP). In this problem, in contrast to the previous problems, the arc capacity is not fixed a priori but has to be determined at minimum cost, while allowing for all commodities to be routed. We assume that for each arc $uv \in A$, a cost $c_{uv} \in \mathbb{R}_+$ is given. This cost has to be paid to make k_{uv} units of capacity available on the arc uv (we say the capacity is *installed* or *activated* on uv). We call one unit of installed capacity an *arc (edge-) module* or a *module of capacity*. For a given set of commodities Q, the network design problem asks for "a minimum cost capacity installation on the arcs in A, such that, for any $q \in Q$, a flow f^q of value d^q (between s^q and t^q) exists such that all these flows together do not exceed the capacity on the arcs".

We extend Definition 1.6 as follows:

Definition 1.7 (Network Design Problem)**.** *Given a directed graph $G = (V, A)$, $S \subseteq V$, $T \subseteq V$, a collection Q of pairs $(s, t) \in S \times T$, a demand function d, a capacity function k, and a cost function c, where*

$$d : Q \to \mathbb{R}_+, q \mapsto d^q, \tag{1.6a}$$

$$k : A \to \mathbb{R}_+, uv \mapsto k_{uv}, \qquad \text{and} \tag{1.6b}$$

$$c : A \to \mathbb{R}_+, uv \mapsto c_{uv}, \tag{1.6c}$$

*the **Network Design Problem** (NDP) asks for a function*

$$x : A \to \mathbb{Z}_+, uv \mapsto x_{uv} \tag{1.6d}$$

such that a multi commodity flow as described in Definition 1.5 exists, where the capacity constraint for the multi commodity flow is replaced by

$$\sum_{q \in Q} f^q_{uv} \leq k_{uv} x_{uv} \qquad \forall uv \in A, \tag{1.6e}$$

so as to minimize the total cost $\sum_{uv \in E} c_{uv} x_{uv}$.

With respect to Constraint (1.6e), we say that x_{uv} *batches* or *bulks* (of capacity k_{uv}) are installed on the arc uv. This expression is especially important in the so-called *rent-at-bulk* model. This is an extension of the NDP problem where different batches, yielding different amounts of capacity, are available at different prices for installation on each arc. We refer to the work of Awerbuch and Azar [11] for additional information on this extension. In this case, economies of scale usually dictate that larger bulks of capacity are *relatively* (or even proportionally) cheaper than smaller bulks. This extension can be easily incorporated into Constraint (1.6e) by adding different x variables for the different kinds of batches. We will come back to the extended case in Section 3.2.

As we will see in Chapter 2 and Chapter 3, NDP is an important part of the telecommunication problems presented in this work, as both the *Network Design Problem with Compression* and the *Virtual Network Embedding Problem* can be seen as generalizations of NDP. We conclude this section with some basic results.

It is well known that the NDP problem is "theoretically difficult", as is more formally stated in the next Theorem:

Theorem 1.3 (Johnson et al. [81])**.** *NDP is strongly $\mathcal{NP}$-hard.*

NDP can be modeled as a MILP. In the corresponding literature, there are multiple formulations available. We present an arc-flow formulation:

Remark 1.1. *Denoting by $x_{uv} \in \mathbb{Z}_+$ the number of installed capacity modules on arc $uv \in A$ and by f^q_{uv} (f^q_{vu}) the percentage of flow corresponding to commodity $q \in Q$ routed along arc uv, an edge-flow formulation of the NDP problem is given by:*

$$\min \sum_{uv \in A} c_{uv} x_{uv} \tag{1.7a}$$

$$\text{s.t.} \sum_{uv \in A} f^q_{uv} - \sum_{vu \in A} f^q_{vu} = \begin{cases} 1 & u = q^t \\ -1 & u = q^s \\ 0 & else \end{cases} \qquad \forall q \in Q, u \in V \tag{1.7b}$$

$$\sum_{q \in Q} d^q f^q_{uv} \leq k_{uv} x_{uv} \qquad \forall uv \in A \tag{1.7c}$$

$$f^q_{uv} \in [0,1], x_{uv} \in \mathbb{Z}_+. \tag{1.7d}$$

The Objective Function (1.7a) *consists of the costs of the capacity installation. Constraint* (1.7b) *models the flow balance, ensuring that all commodities are routed, and Constraint* (1.7c) *enforces that sufficient capacity is installed on every edge. Constraint* (1.7d) *denotes the domains of the variables. In contrast to the previous definitions, the f variables model the percentage of the demand (corresponding to a commodity) traveling along an arc and not the absolute value. A model where f denotes a fraction of the total demand volume of a commodity is possible as well. In this work, we prefer the above model as Constraint* (1.7b) *is "cleaner". Especially with respect to the later chapters, this allows for an easier application of data uncertainty concepts.*

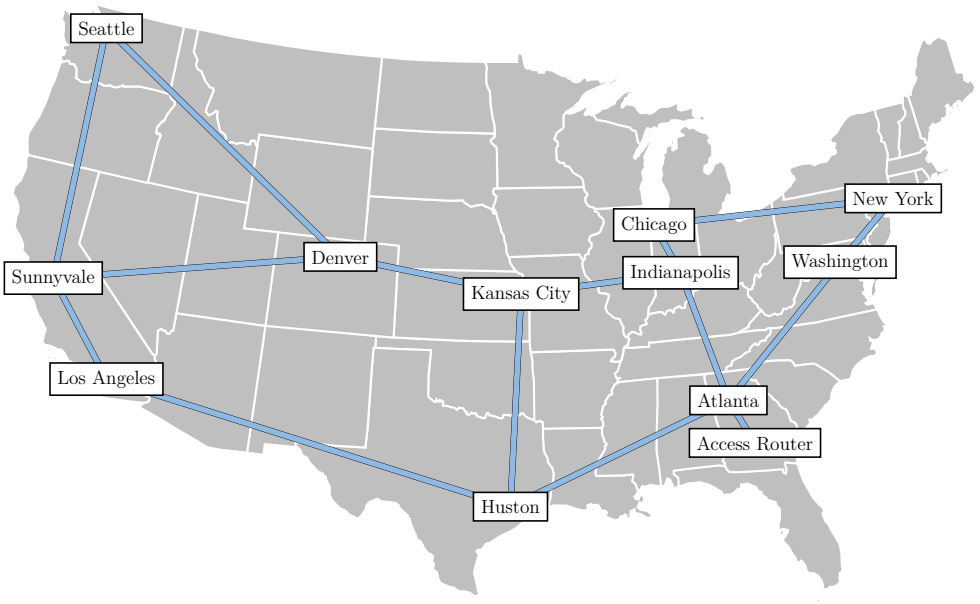


Figure 1.1: The ABILENE network as specified in the SNDLIB [100]. The instance was originally provided by Yin Zhang, University of Texas.

We remark that the NDP problem has been extensively studied and we refer to the early survey by Magnanti and Wong [90] and the monograph by Pióro and Medhi [104] for further information and additional references.

Concluding this section, we refer to Figure 1.1 for an illustration of a telecommunication network. The picture shows the ABILENE network, a real-life high-performance backbone network as specified in the SNDLIB [100]. The nodes correspond to certain cities in the United States, the edges represent the immediate connections between those.

1.3 Extensions, variations, and special cases

In this section, we briefly comment on variants of the problems presented above. Note that in some works, e.g., in Magnanti et al. [92], the NDP problem is also referred to as the *Network Loading* problem. We point out that, while these problems have been defined on directed graphs in this work, the definitions can easily be adapted to the undirected case. In the latter, any flow on an edge $uv \in E$ has to be indexed over the two directions it can take, either from u to v or vice versa. We assume that, in the undirected case, the two flows in the two directions of an edge share the same capacity. Then, explicitly writing $\{u, v\}$ for an edge in E and (u, v), respectively (v, u) for the direction of the flow, the new Capacity Constraint (1.7c) reads:

$$\sum_{q \in Q} d^q \left(f^q_{(u,v)} + f^q_{(v,u)} \right) \leq k_{\{u,v\}} x_{\{u,v\}} \qquad \forall \{u, v\} \in E. \tag{1.8}$$

In the following, the specific model we consider will be made clear, however the additional brackets will be omitted. In this case and under mild conditions, the underlying polyhedron of Formulation (1.7a)–(1.7d) in the x-space is full dimensional.

Lemma 1.1. *Let G be a connected, undirected graph and consider the NDP problem as described above. The projection of the underlying polyhedron of Formulation* (1.7a)–(1.7d) *on the x-space has dimension $|E|$.*

Proof. Since G is connected, there exists a spanning tree in G. We obtain a feasible solution by installing a sufficiently high capacity on each edge of this tree. Additional solutions are obtained by installing an additional element on any edge in the network. □

We further remark that, occasionally, we will refer to a flow as *traffic*, since this terminology is closer to the application.

1.3.1 Traffic routing variations

The arcs a flow utilizes, i.e., the arcs where a flow value is unequal to zero, can be important for the application. We refer to the decision which arcs to use or, in general, to any constraints relating the flows to the arcs, as the *routing scheme* or *routing protocol* (of the traffic). While the routing in Definition 1.7 is not constrained, there are applications where this is necessary or desired. In practice, e.g., *single path routing* is often enforced to avoid packet reordering issues or, in general, so to keep the routing scheme simple. This way, when considering a single commodity $q \in Q$, the arcs $uv \in A$ for which $f^q_{uv} > 0$ are obliged to form a (simple) path in G.

In the formal sense, such constraints obviously yield a restriction of the original problem. Usually, these can be easily enforced in MILP formulations. Compare, for example, the Formulation (1.7a)-(1.7d) for NDP. In this formulation, single path routing is enforced by imposing $f^q_{uv} \in \{0, 1\}$. Naturally, such restrictions can have a significant impact on the computational complexity of the corresponding problem. Consider the case where the single path routing is even more constrained, i.e., where for any commodity $q \in Q$ the routing has to take a fixed path, e.g., the *shortest path* with respect to some length function, between the source and the sink node. Then, NDP becomes computationally "easy" as is shown in the following Lemma.

Lemma 1.2. *Consider the NDP problem. Let, for any commodity $q \in Q$, a single routing path P^q be given on which the commodity has to be routed. That is, let*

$$f^q_{uv} = \begin{cases} 1 & \forall\, uv \in A : uv \in P^q \\ 0 & \forall\, uv \in A : uv \notin P^q. \end{cases} \tag{1.9}$$

Under these restrictions, NDP is in $\mathcal{P}$.

Proof. For any $q \in Q$ and any arc $uv \in A$, the flow f^q_{uv} on this arc either is fixed to $f^q_{uv} = 0$ or to $f^q_{uv} = 1$, because the routing is given as a single path. Hence, the minimum

capacity necessary on any arc $uv \in A$ is

$$x_{uv} := \left\lceil \frac{\sum_{q \in Q} d^q f^q_{uv}}{k_{uv}} \right\rceil \tag{1.10}$$

and the minimum cost is completely determined by these values. □

Clearly, Lemma 1.2 can be generalized to any situation in which the routing variables are fixed. We point out that in many applications, such restrictions are implicitly imposed by the network structure, e.g., if the underlying network is a tree or even a star. In this case, the NDP problem is easy as shown above. We will encounter such special cases within the following two chapters.

However, if the routing is not fixed to a specific path but assumed to be *some* path whose specific form is part of the optimization problem, the problem can still be difficult or, worse, can even become *more* difficult. Compare, for instance, the Multi Commodity Flow Problem, which is originally in $\mathcal{P}$, but where it holds that if, for each commodity, the flows have to form a path, the problem becomes $\mathcal{NP}$ hard.

Theorem 1.4 (Karp [82])**.** *The integral Multi Commodity Flow problem, that is where $f^q_{uv} \in \{0, 1\}$ is required for all $uv \in A$, $q \in Q$, is strongly $\mathcal{NP}$-hard.*

Therefore, we will carefully point out any constraints imposed on the routing schemes within the optimization problems as considered in this work. Concluding this subsection, we mention that, in the following work, we will refer to a routing scheme as *splittable* if $f^q_{uv} \in [0, 1]$ or, as *unsplittable*, if $f^q_{uv} \in \{0, 1\}$ for all $q \in Q$ and $uv \in A$.

1.3.2 Demand values and capacities

In the context of the different variants of the NDP problem, structure within the problem's parameters can be important. Let us consider the parameters d^q representing the demand volume of a commodity $q \in Q$ and the parameters k_{uv} defining the capacity of an arc $uv \in A$. Depending on the structure of these parameters, various results hold for NDP or do not apply. For example, if k_{uv} is a constant, say, equal to one, certain important classes of inequalities are facet defining. The same inequalities are "just" valid, if this is not the case. Consider the following example:

Example 1.1. *Consider the NDP problem on an undirected graph G and assume that $k_{uv} = 1$ for all $uv \in A$. Let $S \subseteq A$ describe a set of arcs forming a cut in G. Then, the* ***Cutset Inequality*** *(see Magnanti et al. [92])*

$$\sum_{uv \in S} x_{uv} \geq \left\lceil c_0 \right\rceil, \qquad \text{where} \qquad c_0 := \sum_{\substack{q \in Q: \\ s^q \in S, \\ t^q \notin S}} d^q + \sum_{\substack{q \in Q: \\ s^q \notin S, \\ t^q \in S}} d^q, \tag{1.11}$$

is valid for the convex hull of the NDP problem as of Formulation (1.7a)–(1.7d)*.*

If $c_0 \in \mathbb{Z}_+$, the inequality is implied. If c_0 is fractional and S as well as $V \setminus S$ are connected, the inequality describes a facet of the formulation.

We discuss the case that k_{uv} is not constant in the following example.

Example 1.2. *Let a Cutset Inequality be given where the cut contains exactly two edges, on which an edge-module of $k_1 = 3$, respectively of $k_2 = 7$, units of capacity can be installed. Assume that $c_0 := 19.9$ units have to be sent across the cut. Assuming the connectivity restrictions as mentioned above, the Cutset Inequality* (1.11) *writes as*

$$3x_1 + 7x_2 \geq \lceil 19.9 \rceil = 20. \tag{1.12}$$

Apart from $x_1 = 2$ and $x_2 = 2$, there is no positive integer combination of x_1 and x_2 such that the inequality holds tight. As a consequence, the inequality cannot be a facet of the convex hull. The same holds for the rank-one Chvátal-Gomorry Inequalities, obtained by applying the the Chvátal-Gomorry procedure, see Nemhauser and Wolsey [98]:

$$x_1 + \left\lceil \frac{7}{3} \right\rceil x_2 \geq \left\lceil \frac{19.9}{3} \right\rceil \Leftrightarrow x_1 + 3x_2 \geq 7, \text{and} \tag{1.13a}$$

$$\left\lceil \frac{3}{7} \right\rceil x_1 + x_2 \geq \left\lceil \frac{19.9}{7} \right\rceil \Leftrightarrow x_1 + x_2 \geq 3. \tag{1.13b}$$

We refer to Figure 1.2 for a visualization. For completeness sake, we remark that the convex hull of the (x_1, x_2) space is given by

$$x_1 \geq 0, \qquad x_2 \geq 0, \tag{1.14a}$$

$$x_1 + 2x_2 \geq 6 \tag{1.14b}$$

$$2x_1 + 5x_2 \geq 14 \tag{1.14c}$$

The example shows that, if the Cutset Inequality is extended to the more general setting, it is not necessarily facet defining any more. As a consequence, in the literature, those variants where $d^q \notin \mathbb{Z}_+$ for all $q \in Q$ and where k_{uv} is constant are most common. Without loss of generality, being constant implies that $k_{uv} = 1$ for all arcs $uv \in A$. Although, the more general setting is less appealing from a theoretical point of view, it is often closer to real life applications.

We conclude this subsection by mentioning another variation of the capacity constraints, which mixes the assumptions of the constant and the non-constant capacity values. This is the so called *rent-at-bulk* model. In this model, units of capacities are installed in batches of different size, i.e., there are different edge-modules available which offer different capacities when installed. Thereby, usually economies of scale apply as well. In this sense, in the standard NDP problem, there is a single batch of capacities available per arc. Commonly, in this situation the capacity of a certain batch is equal for all arcs in the network and typically, there are constraints present which limit the total amount of edge modules available.

Throughout this work, we will properly specify which variant we currently consider.

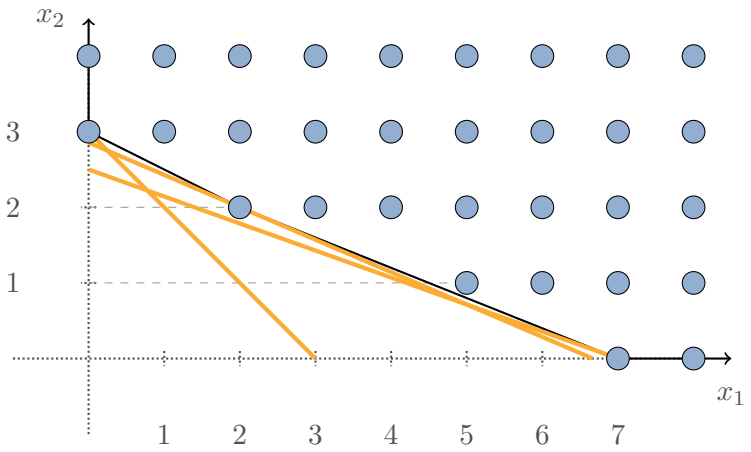


Figure 1.2: The two-edge NDP polytope, as described in Example 1.2, projected on the (x_1, x_2)-space with Cutset Inequalities. Integer, feasible points are marked in blue. The dotted gray lines indicate the coordinate axis. The black lines display the convex hull, the orange lines show the rounded Cutset Inequalities.

1.4 Data uncertainty in optimization problems

1.4.1 Motivation and introduction

In the previous sections, we have investigated *deterministic* versions of the Maximum Flow or the NDP problem. That is, we have assumed that all input data is known. However, in applications, this knowledge is often not available. In this section, we focus on the case that such data is uncertain and we show how –given that the problem is formulated as a MILP– this uncertainty can be accounted for in the problem's formulation. For the remainder of this section, consider the following situation:

For a set of row indices $\mathcal{I}$ and a set of column indices $\mathcal{J}$, let for $A \in \mathbb{R}^{|\mathcal{I}|\times|\mathcal{J}|}$, $b \in \mathbb{R}^{|\mathcal{I}|}$ and $c \in \mathbb{R}^{|\mathcal{J}|}$ an optimization problem be given in the form

$$\begin{aligned} \min \quad & cx && \text{(1.15a)} \\ \text{s.t.} \quad & Ax \leq b && \text{(1.15b)} \\ & x \geq 0. && \text{(1.15c)} \end{aligned}$$

For any $i \in \mathcal{I}$ and $j \in \mathcal{J}$, we assume that the coefficient $a_{ij} \in A$ is not precisely known. That is, for instance, the case, because of measurement or prediction errors or, simply, because of natural fluctuations in the data, for example, if day and night traffic traces are provided. This implies that an "optimizer" has to find a solution to the Problem (1.15a)–(1.15c) *without* explicit knowledge of the coefficients a_{ij}. We assume, though, that an *estimation* of the data (i.e., the coefficients) is available. We define such estimation as the set $\mathcal{A}_{ij} \subseteq \mathbb{R}$ of values, from which a_{ij} can take (*realize*) a value, depending on a certain probability distribution $\mathcal{P}_{ij}$. We denote $\bar{a}_{ij}$ as the expected value of a_{ij} and we assume that $\mathcal{A}_{ij}$ and $\bar{a}_{ij}$, but not necessarily $\mathcal{P}_{ij}$, is known to the optimizer.

A naive approach to account for such data uncertainty is shown in the following example.

Example 1.3. *Consider the case that all $a_{ij} \in A$ are bounded. We alleviate the uncertainty of the coefficients a_{ij} by solving Problem* (1.15a)–(1.15c) *where a_{ij} is replaced by its* ***worst case*** *realization, i.e., by $a_{ij} := \sup \mathcal{A}_{ij}$. Clearly, this guarantees that the solution will be feasible after revealing a_{ij}, independently of its realization. However, this comes at the expense of a relatively high objective value, compared to solutions obtained by e.g., setting $a_{ij} := \bar{a}_{ij}$. Nevertheless, in this case, we say that the solution is* ***protected*** *against all possible realizations of the parameter a_{ij}.*

In general, a solution to Problem (1.15a)–(1.15c) with data uncertainty is assessed by two qualities: the first one being the objective value (i.e., the *cost*, respectively the *profit* in case of a maximization problem) and the second one being the probability that the solution is feasible (we refer to this probability as *protection*). In Example 1.3, the solution excels from the latter point but falls behind at the first. In most cases, we observe a trade-off between the protection and the profit/cost we obtain: the higher the desired protection of a solution, the more general realizations of a_{ij} have to be taken into account, and thus, the less profitable, respectively the more costly, (read *conservative*) a solution becomes. From an economical point of view, this trade-off is very appealing since, in many applications, it is not necessary to protect a solution against all possible data realizations, instead it is sufficient to consider only a few likely cases.

Research on how such data uncertainty can be modeled within an optimization problem, especially with respect to the trade-off mentioned before, is gathered in the field of *robust optimization*. Note that, in general, we call an optimization problem *robust* if it includes data uncertainty (as described above). Among the many works in that field, we mention that of Soyster [117] as one of the first. In this work, the data uncertainty is tackled by a linear optimization model such that the resulting solutions are feasible for all input data belonging to some convex set. Following up, the works by Ben-Tal and Nemirovski [14, 15, 16], El Ghaoui and Lebret [47], and by El Ghaoui et al. [48] refined and extended the field, developing frameworks, allowing to model a flexible trade-off between objective value and feasibility.

In this work, we focus especially on the works of Bertsimas and Sim [17, 18]. The authors present a robust optimization approach, named Γ-*robustness*, which is very attractive because of its computational tractability and which offers full control over the level of conservatism of the obtained solutions. To indicate the scientific impact of Bertsimas and Sim's work, we refer to the works of Altın et al. [4] and Koster et al. [85] where this concept has been successfully applied to several network optimization problems. Additionally, we mention the works of Coudert et al. [43] and Coniglio et al. [41, 42] where the author tackles data uncertainty in two optimization problems via Γ-robustness. On a more theoretical side, the concept of Γ-robustness has been expanded and refined in the works of Büsing and D'Andreagiovanni [26, 27]. For further details on the Γ-robustness, we refer to Section 1.4.3.

In the following, we propose a robustness concept that can be interpreted as an extension or a refinement of the Γ-robustness concept of Bertsimas and Sim. Encompassing the Γ-robustness as a special case, this framework allows to model data uncertainty in a much more refined way, allowing a better control of the obtained solutions with respect to e.g., the level of conservatism. We start by defining the general setting in Section 1.4.2. Subsequently, we show how Γ-robustness can be derived as a special case in Section 1.4.3, and, in Section 1.4.4, we discuss how uncertain influences can be modeled more elaborately by the extended concept.

1.4.2 A framework for data uncertainty

As indicated above, we assume that some parameters $a_{ij} \in A$ of Problem (1.15a)–(1.15c) are not precisely known beforehand but realize in some set $\mathcal{A}_{ij}$. At first, we formalize the realization of such coefficient. In detail, we assume each a_{ij} is determined by $K \in \mathbb{Z}_+$ *random events* whose joint outcome yields the value of a_{ij} after each of these events occurred. We formalize this as follows.

Let E denote the set of all random events that influence the data in the model and let $|E| = K$. Throughout this work, we assume that each random event $k \in E$ can be characterized as a random variable with a certain probability distribution. Such variable realizes an outcome $\tilde{e}_k$ in a symmetric interval $[\bar{e}_k - \hat{e}_k, \bar{e}_k + \hat{e}_k]$, where $\bar{e}_k \in \mathbb{R}$ is its expected outcome and $\hat{e}_k \in \mathbb{R}$ equals its maximal deviation from the expected value. Then, a_{ij} is constructed by a function A_{ij} operating on the outcome of the events $k \in E$:

$$\begin{aligned} A_{ij}: \quad \mathbb{R}^K &\longrightarrow \mathbb{R}, \\ \begin{pmatrix} \tilde{e}_1 \\ \tilde{e}_2 \\ \vdots \\ \tilde{e}_K \end{pmatrix} &\longmapsto a_{ij}. \end{aligned} \tag{1.16}$$

We rely on the function A_{ij} to introduce multi-source data uncertainty:

Definition 1.8 (Multi-source data uncertainty)**.** *Consider Problem* (1.15a)–(1.15c)*. Assume that the coefficients $a_{ij} \in A$ are uncertain and that their realization is given as described in* (1.16)*. We refer to this setting as* ***multi source data uncertainty****.*

We refer to Figure 1.3 where the situation of Definition 1.8 is depicted. In this figure, the relation between the random events E (orange) and the coefficients a_{ij} (blue) is shown as a bipartite graph. Note that, in general, each event relates to multiple coefficients and vice versa (many to many mapping).

As a consequence, the underlying optimization problem can be expressed in terms of the random events, respectively their outcomes $\tilde{e}_k$, instead of the uncertain coefficients. We extend Example 1.3 to the situation of Definition 1.8.

Example 1.4. *We consider the situation of Definition 1.8 and require a solution of Problem* (1.15a)–(1.15c) *to be feasible for* all *possible realizations of the events in E. A*

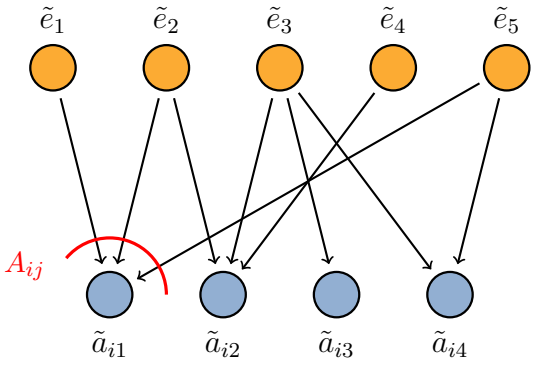


Figure 1.3: Multi-source data uncertainty – multiple random events $k \in E$ with realization $\tilde{e}_k$ (orange) determine the single coefficient a_{ij} (blue) of the constraint matrix A. One variable can influence multiple coefficients.

robust version of Problem (1.15a)–(1.15c) *is given by*

$$\min \quad c^t x \tag{1.17a}$$

$$\text{s.t.} \quad \sup_{\tilde{e}_k \in [\bar{e}_k - \hat{e}_k, \bar{e}_k + \hat{e}_k],\ k=1,\dots,K} \sum_{j \in \mathcal{J}} A_{ij} \begin{pmatrix} \tilde{e}_1 \\ \vdots \\ \tilde{e}_K \end{pmatrix} x_j \leq b_i \quad \forall\, i \in \mathcal{I} \tag{1.17b}$$

$$x \geq 0. \tag{1.17c}$$

In the particular case that the uncertain influences are appropriately structured, the supremum in Constraint (1.17b) *can be derived beforehand. For example, assuming that every coefficient is uniquely determined by a single event, the robust problem boils down to solving the Problem* (1.15a)–(1.15c) *with worst case data as shown in Example 1.3.*

As argued before, a worst-case setting as described in Example 1.4 might be too restrictive for practical applications. One way to obtain a less conservative solution is to impose additional restrictions onto the realizations $\tilde{e}_k$ of the events in E such that not all possible outcomes have to be considered. This can be accomplished as follows:

Consider a collection $\mathcal{C} = \{\mathcal{S}_1, \dots, \mathcal{S}_{|\mathcal{C}|}\}$ of sets $\mathcal{S}_i \subseteq E$, $i = 1, \dots, |\mathcal{C}|$ with $\bigcup_{\mathcal{S} \in \mathcal{C}} \mathcal{S} = E$. For each $\mathcal{S} \in \mathcal{C}$ define a parameter $\Gamma_{\mathcal{S}} \in \mathbb{Z}_+$. We expand Definition 1.8 to only consider such realizations $\tilde{e}_k$ for which at most $\Gamma_{\mathcal{S}}$ many values deviate from their expectation per set $\mathcal{S}$. In this case, the parameters $\Gamma_{\mathcal{S}}$ serve as a bound on the amount of uncertainty, that is, the number of deviations from the expected values, which can occur in the set $\mathcal{S}$. Note that, in general, the sets in $\mathcal{C}$ need not to be disjoint.

In such setting, the parameters $\Gamma_{\mathcal{S}}$ model the trade-off mentioned in the previous section: a high value of the $\Gamma_{\mathcal{S}}$ leads to solutions with a high cost but which are feasible with a high probability while a low value of $\Gamma_{\mathcal{S}}$ yields solutions with the opposite characteristics. The visualization of sets $\mathcal{S}$ and their influence on the random events, respectively the coefficients $a_{ij} \in A$, can be found in Figure 1.4.

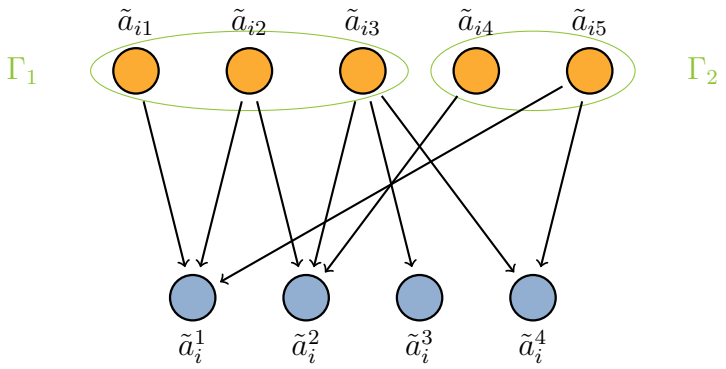


Figure 1.4: Restricting the uncertainty – Different random events $e_k \in E$ are partitioned into two sets $\mathcal{S}_1$ and $\mathcal{S}_2$. In each set, only a limited number of the events may have an outcome different to their expected value.

Note that, for $\Gamma_\mathcal{S} \geq |\mathcal{S}|$ for all $\mathcal{S} \subseteq \mathcal{C}$, the robust problem boils down to the worst-case setting, while $\Gamma_\mathcal{S} = 0$ prevents any deviations from the nominal (expected) values. We define the *multi-source robust* problem as follows:

Definition 1.9 (The multi-source robust problem)**.** *Let a problem under multi-source data uncertainty as in Definition 1.8 be given. Denote the set of random events as E with $|E| = K \in \mathbb{Z}_+$ and let $\tilde{e}_k \in [\bar{e}_k - \hat{e}_k, \bar{e}_k + \hat{e}_k]$ for all $k \in K$. Let a collection $\mathcal{C}$ of sets $\mathcal{S}_i$, $i = 1, \dots |\mathcal{C}|$ be a cover of E, and let parameters $\Gamma_\mathcal{S} \in \mathbb{Z}_+$ $\forall \mathcal{S} \in \mathcal{C}$ be given as described above. Then*

$$\min \quad c^t x \tag{1.18a}$$

$$\text{s.t.} \quad \sup_{\substack{\tilde{e}_k \in [\bar{e}_k - \hat{e}_k, \bar{e}_k + \hat{e}_k],\ k=1,\dots,K: \\ \forall \mathcal{S} \in \mathcal{C}: |\{k \in \mathcal{S} | \tilde{e}_k \neq \bar{e}_k\}| \leq \Gamma_\mathcal{S}}} \sum_{j \in \mathcal{J}} A_{ij} \begin{pmatrix} \tilde{e}_1 \\ \vdots \\ \tilde{e}_K \end{pmatrix} x_j \leq b_i \quad \forall\, i \in \mathcal{I} \tag{1.18b}$$

$$x \geq 0 \tag{1.18c}$$

*is the **multi-source robust** version of Problem* (1.15a)–(1.15c) *with data uncertainty.*

The solutions obtained by the multi-source robust problem are deterministically feasible if at most $\Gamma_\mathcal{S}$ many values deviate from their expectation in each set $\mathcal{S}$. However, they are not necessarily feasible if *more* deviations occur.

Thus, in the following subsections, we discuss the multi-source robust problem with respect to different functions $\mathcal{A}_{ij}$.

1.4.3 Bijective uncertainty (Γ-robustness)

Defining bijective uncertainty

In this section, we assume that $K = |\mathcal{I}||\mathcal{J}|$, i.e., we assume that there are as many random events as there are entries in the constraint matrix A. For illustration purposes,

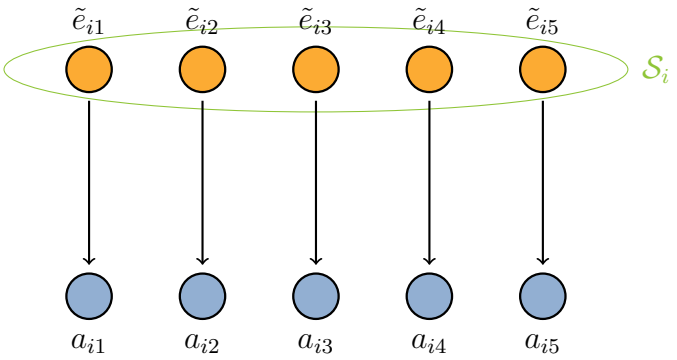


Figure 1.5: Bijective Uncertainty – given a row $i \in \mathcal{I}$ of the constraint matrix A, all coefficients a_{ij} are uniquely identified by a singe random event $ij \in E$ and determined by its realization $\tilde{e}_{ij}$. All the coefficients/events of row i are gathered in $\mathcal{S}_i$ and at most Γ_i many may deviate from their expectation.

we write $ij \in E$ instead of $k \in E$ and we identify E with the cross-product $\mathcal{I} \times \mathcal{J}$ throughout this subsection. Further, for any $i \in \mathcal{I}$ and $j \in \mathcal{J}$, let A_{ij} be the projection onto the ij^{th} component of the input vector, i.e., let

$$\begin{aligned} A_{ij} : \quad \mathbb{R}^{|\mathcal{I}||\mathcal{J}|} &\longrightarrow \mathbb{R}, \\ \begin{pmatrix} \tilde{e}_{11} \\ \tilde{e}_{12} \\ \vdots \\ \tilde{e}_{|\mathcal{I}||\mathcal{J}|} \end{pmatrix} &\longmapsto \tilde{e}_{ij} =: a_{ij}. \end{aligned} \tag{1.19}$$

This way, every event $ij \in E$ can be *uniquely* identified by a coefficient a_{ij} and vice versa. Let $\mathcal{C} := \{\mathcal{S}_1, \ldots, \mathcal{S}_{|\mathcal{I}|}\}$ be such that for $i \in \mathcal{I}$, it is $S_i := \{ij \in E \mid j \in \mathcal{J}\}$, and let $\Gamma_i \in \mathbb{Z}_+$ for all $i \in \mathcal{I}$ be given. By construction, for each row i of the constraint matrix the set $\mathcal{S}_i$ contains exactly the "uncertainty" of this row. Restricting this uncertainty, we only consider realizations $\tilde{e}_{ij}$ where at most Γ_i many events deviate from their expected values per row i of the matrix A, respectively per set S_i. Again, the Γ_i model a trade-off between feasibility and objective value. A sketch of the relation between the random events and the coefficients is given in Figure 1.5. For future reference, we call this model the *bijective-uncertainty* model. For this setting, we define the Γ-robust problem:

Definition 1.10 (The Γ-robust problem)**.** *Let a problem under multi-source data uncertainty be given as in Definition 1.8. Denote the set of random events as E with $|E| = |\mathcal{I}||\mathcal{J}|$ and let $\tilde{e}_{ij} \in [\bar{e}_{ij} - \hat{e}_{ij}, \bar{e}_{ij} + \hat{e}_{ij}]$ for all $i \in \mathcal{I}, j \in \mathcal{J}$.*

Let $\mathcal{C} = \{\mathcal{S}_1, \ldots, \mathcal{S}_{|\mathcal{C}|}\}$ with $|\mathcal{C}| = |\mathcal{I}|$ and $S_i := \{ij \in E \mid j \in \mathcal{J}\}$ for $i \in \mathcal{I}$ be given. Further, let parameters $\Gamma_i \in \mathbb{Z}_+$ $\forall i \in \mathcal{I}$ be given and for $i \in \mathcal{I}$ and $j \in \mathcal{J}$ let A_{ij} be

defined as in (1.19)*. Then*

$$\min \quad c^t x \tag{1.20a}$$

$$\text{s.t.} \quad \sup_{\substack{\tilde{e}_{ij}\in[\bar{e}_{ij}-\hat{e}_{ij},\bar{e}_{ij}+\hat{e}_{ij}],\ ij\in E:\\ |\{ij\in\mathcal{S}_i|\tilde{e}_{ij}\neq\bar{e}_{ij}\}|\leq\Gamma_i}} \sum_{j\in\mathcal{J}} \tilde{e}_{ij}x_j \leq b_i \quad \forall\, i\in\mathcal{I} \tag{1.20b}$$

$$x \geq 0. \tag{1.20c}$$

is the Γ***-robust*** *version of Problem* (1.15a)–(1.15c) *with data uncertainty.*

By definition, there is one parameter Γ_i per row of A. To keep the notation simple, we assume that $\Gamma_i = \Gamma \in \mathbb{Z}_+$ for all $i \in \mathcal{I}$. Note that all following results do also apply to the more general case. We directly have

Corollary 1.1. *The* Γ*-robust Problem* (1.20a)–(1.20c) *can be equivalently written as*

$$\min \quad c^t x \tag{1.21a}$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{J}} \bar{e}_{ij}x_j + \max_{\substack{Q\subseteq\mathcal{J}:\\ |Q|\leq\Gamma}} \sum_{j\in Q} \hat{e}_{ij}x_j \leq b_i \quad \forall\, i\in\mathcal{I} \tag{1.21b}$$

$$x \geq 0. \tag{1.21c}$$

Proof. The argument of the supremum is a continuous function whose arguments themselves are closed and bounded: $\tilde{e}_{ij} \in [\bar{e}_{ij} - \hat{e}_{ij}, \bar{e}_{ij} + \hat{e}_{ij}]$. □

By identifying the random events with their corresponding coefficients, that is, by replacing e_{ij} by a_{ij}, i.e., $\tilde{a}_{ij} \in [\bar{a}_{ij} - \hat{a}_{ij}, \bar{a}_{ij} + \hat{a}_{ij}]$, we obtain a robust problem which is equivalent to the Γ-robustness concept as proposed by Bertsimas and Sim [17, 18].

Theorem 1.5 (Bertsimas and Sim [18])**.** *With respect to the* Γ*-robust Problem* (1.21a)–(1.21c)*, it holds that:*

1. *Each solution is deterministically feasible, if at most* Γ *many coefficients deviate from their expected value.*

2. *The problem can be compactly reformulated as a MILP.*

If the $\tilde{a}_{ij}$ *are symmetrically distributed and stochastically independent, it holds that*

3. *Given a solution* x^* *and consider* $|\mathcal{J}|$ *sufficiently large, the probability that a constraint of type* (1.21b) *is violated is approximately*

$$1 - \Phi\left(\frac{\Gamma - 1}{\sqrt{|\mathcal{J}|}}\right), \tag{1.22}$$

where Φ *is the cumulative distribution function of the standard normal distribution.*

From a practical perspective, the reformulation as a MILP is most interesting. We derive this reformulation in the following, referring for a proof of the other statements to Bertsimas and Sim [18]. For a fixed row $i \in \mathcal{I}$ and for any fixed x_j^*, the maximization subproblem in Constraint (1.21b) can be expressed as a MILP. Let the variable $\alpha_j \in \{0,1\}$ denote whether the coefficient of variable $j \in \mathcal{J}$ is deviating. Then, denoting the variables of the LP dual in brackets, the subproblem can be stated as

$$\max \quad \sum_{j\in\mathcal{J}} \left(\hat{e}_{ij} x_j^*\right) \alpha_j \tag{1.23a}$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{J}} \alpha_j \leq \Gamma \qquad [\pi] \tag{1.23b}$$

$$\alpha_j \in \{0,1\}\,. \qquad [\rho] \tag{1.23c}$$

The constraint matrix is totally unimodular, i.e., it allows to relax the integrality constraints without changing integrality of the obtained solutions. The resulting LP relaxation can be dualized to obtain

$$\min \quad \Gamma\pi + \sum_{j\in\mathcal{J}} \rho_j \tag{1.24a}$$

$$\text{s.t.} \quad \pi + \rho_j \geq \hat{e}_{ij} x_j^* \qquad \forall j \in \mathcal{J} \tag{1.24b}$$

$$\pi, \rho_j \geq 0. \tag{1.24c}$$

Substituting with Problem (1.24a) – (1.24c) the inner problem in Constraint (1.21b) (adding the index i), we obtain a compact formulation for the robust problem:

$$\min \quad c^t x \tag{1.25a}$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{J}} \bar{e}_{ij} x_j + \Gamma\pi_i + \sum_{j\in\mathcal{J}} \rho_{ij} \leq b_i \qquad \forall\, i \in \mathcal{I} \tag{1.25b}$$

$$\pi_i + \rho_{ij} \geq \hat{e}_{ij} x_j^* \qquad \forall i \in \mathcal{I}, j \in \mathcal{J} \tag{1.25c}$$

$$x, \pi, \rho \geq 0. \tag{1.25d}$$

Remark 1.2. *All results of Theorem 1.5 are independent of the distributions $\mathcal{P}_{ij}$.*

We stress the importance of Property 3 in Theorem 1.5. This result offers a direct and easy way to evaluate (approximative) bounds on the protection of a Γ-robust solution. We illustrate this in the following example:

Example 1.5. *Consider $|\mathcal{J}| \in \{100, 1000\}$ and assume that, for a $p \in \{0.05, 0.02, 0.01\}$, we look for a solution, for which the probability that Constraint* (1.21b) *is violated is smaller than p. By Theorem 1.5, Property* 3*, we derive different values of Γ, see Table 1.1, which will approximatively give the desired protection.*

According to Theorem 1.5, we can solve the Γ-robust problem as a MILP. We conclude this subsection by formulating an alternative way to solve the Γ-robust problem.

Table 1.1: Minimal (integer) Γ to derive a desired protection of value p as derived from Theorem 1.5(3), depending on $|\mathcal{J}|$.

$\lvert\mathcal{J}\rvert$	p	Γ	Approx.: (Th. 1.5 (3))
100	0.05	18	0.0446
100	0.02	22	0.0179
100	0.01	25	0.0082
1000	0.05	54	0.0469
1000	0.02	66	0.0199
1000	0.01	75	0.0096

Remark 1.3. *We reformulate the* Γ*-robust Problem* (1.21a)–(1.21c) *as*

$$\min \quad c^t x \tag{1.26a}$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{J}} \tilde{e}_{ij}x_j \le b_i \qquad \begin{array}{l}\forall\, i \in \mathcal{I}, \tilde{e} \in \mathbb{R}^{|\mathcal{J}|}: \\ \tilde{e}_{ij} \in [\bar{e}_{ij} - \hat{e}_{ij}, \bar{e}_{ij} + \hat{e}_{ij}], \\ \left|\{\tilde{e}_{ij} \neq \bar{e}_{ij} \mid e_{ij} \in \mathcal{S}_i\}\right| \le \Gamma\end{array} \tag{1.26b}$$

$$x \ge 0. \tag{1.26c}$$

The problem has an infinite number of constraints. However, we can solve the Problem (1.15a)–(1.15c) *(with* $a_{ij} = \bar{e}_{ij}$*) and add Constraint* (1.26b) *in a separation routine. Given a solution* x*, we can therefore employ Problem* (1.23a)–(1.23c) *as separation problem. This way of solving the* Γ*-robust problem has been evaluated, e.g., by Fischetti and Monaci [54]. We note that this subproblem can be solved in* $O(|\mathcal{J}|log(|\mathcal{J}|))$*, for example by sorting the coefficients* $\hat{e}_{ij}x^*_j$*.*

1.4.4 Two-source uncertainty

Defining two-source uncertainty

In this section, we assume that $K = 2|\mathcal{I}||\mathcal{J}|$, i.e., for each entry in the constraint matrix A, there are exactly *two unique*, random events, jointly yielding the coefficient. We write e^1_{ij}, respectively e^2_{ij}, for the events e_k. Let, for $i \in \mathcal{I}$ and $j \in \mathcal{J}$, A_{ij} be the product of two entries of the input vector, i.e.,

$$\begin{array}{rccc} A_{ij}: & \mathbb{R}^{2|\mathcal{I}||\mathcal{J}|} & \longrightarrow & \mathbb{R}, \\ & \begin{pmatrix} \tilde{e}^1_{11} \\ \vdots \\ \tilde{e}^1_{|\mathcal{I}||\mathcal{J}|} \\ \tilde{e}^2_{11} \\ \vdots \\ \tilde{e}^2_{|\mathcal{I}||\mathcal{J}|} \end{pmatrix} & \longmapsto & \tilde{e}^1_{ij} \cdot \tilde{e}^2_{ij} =: a_{ij}. \end{array} \tag{1.27}$$

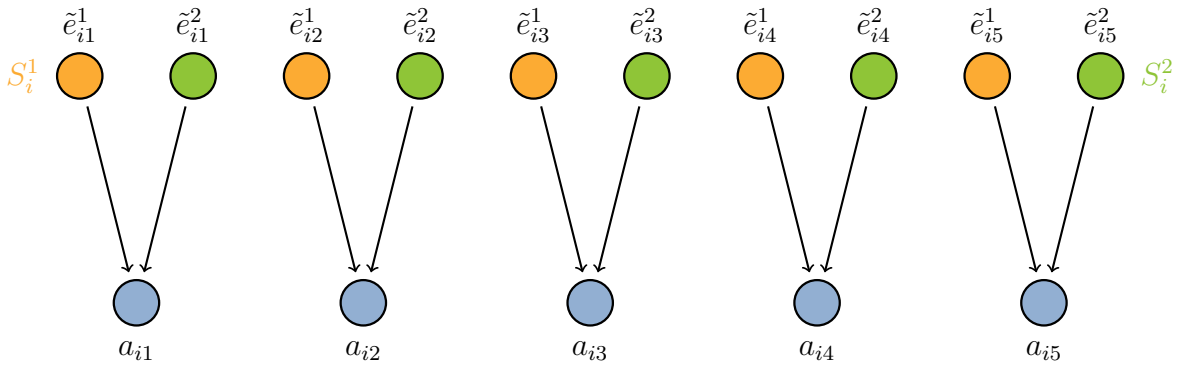


Figure 1.6: Two-source uncertainty: Let for each row $i \in \mathcal{I}$ of the constraint matrix A two unique events e^1_{ij} and e^2_{ij} per coefficient a_{ij} be given. For each of the two groups of events (orange and green), only Γ_1 many events $\tilde{e}^1_{ij}$ (orange), respectively Γ_2 many events $\tilde{e}^2_{ij}$ (green), are allowed deviate to from their expectation.

Let $\mathcal{C} := \{\mathcal{S}^1_1, \dots, \mathcal{S}^1_{|\mathcal{I}|}, \mathcal{S}^2_1, \dots, \mathcal{S}^2_{|\mathcal{I}|}\}$ such that for $i \in \mathcal{I}$ it is $S^1_i := \{e^1_{ij} \mid j \in \mathcal{J}\}$ and $S^2_i := \{e^2_{ij} \mid j \in \mathcal{J}\}$. Additionally, for fixed $\Gamma_1, \Gamma_2 \in \mathbb{Z}_+$, let $\Gamma_{\mathcal{S}^1_i} = \Gamma_1$ and $\Gamma_{\mathcal{S}^2_i} = \Gamma_2$ for all $i \in \mathcal{I}$. The relation between the uncertainty and the coefficients is depicted in Figure 1.6. That is, for each row, the events are partitioned into two groups and for each group $i \in \{1, 2\}$, at most Γ_i many events are allowed to deviate from their expectation. We define the corresponding robust problem:

Definition 1.11 (The two-source robust problem)**.** *Consider Definition 1.8 and let* $K = 2|\mathcal{I}||\mathcal{J}|$ *and* $|\mathcal{C}| = 2|\mathcal{I}|$*. Let* $\Gamma_1, \Gamma_2 \in \mathbb{Z}_+$ *be given. For* $i \in \mathcal{I}$ *and* $j \in \mathcal{J}$*, let* A_{ij} *be as defined in* (1.27)*. The* ***two-source robust*** *version of Problem* (1.15a)–(1.15c) *with data uncertainty is given by*

$$\min \quad c^t x \tag{1.28a}$$

$$\text{s.t.} \quad \sup_{\substack{\tilde{e}^1_{ij} \in [\bar{e}^1_{ij} - \hat{e}^1_{ij}, \bar{e}^1_{ij} + \hat{e}^1_{ij}],\\ \tilde{e}^2_{ij} \in [\bar{e}^2_{ij} - \hat{e}^2_{ij}, \bar{e}^2_{ij} + \hat{e}^2_{ij}]:\\ \left|\{\tilde{e}^1_{ij} \neq \bar{e}^1_{ij} \mid e^1_{ij} \in \mathcal{S}^1_i\}\right| \leq \Gamma_1,\\ \left|\{\tilde{e}^2_{ij} \neq \bar{e}^2_{ij} \mid e^2_{ij} \in \mathcal{S}^2_i\}\right| \leq \Gamma_2}} \sum_{j \in \mathcal{J}} \tilde{e}^1_{ij} \tilde{e}^2_{ij} x_j \leq b_i \quad \forall\, i \in \mathcal{I} \tag{1.28b}$$

$$x \geq 0. \tag{1.28c}$$

To obtain a "simpler" or a more "compact" formulation, we exploit that a_{ij} is given by the product of two random events, i.e., by a slight abuse of notation, we have

$$\begin{aligned} & a_{ij} = \tilde{e}^1_{ij} \cdot \tilde{e}^2_{ij} \\ \Rightarrow \quad & a_{ij} \in [\bar{e}^1_{ij} - \hat{e}^1_{ij}, \bar{e}^1_{ij} + \hat{e}^1_{ij}] \cdot [\bar{e}^2_{ij} - \hat{e}^2_{ij}, \bar{e}^2_{ij} + \hat{e}^2_{ij}]. \end{aligned}$$

This way, we can express the coefficients a_{ij} in terms of the underlying events e^1_{ij} and e^2_{ij}, depending on which of these are deviating from their expectations. Restricting to

deviations to the maximum, i.e., to $\tilde{e}^1_{ij} = \bar{e}^1_{ij} + \hat{e}^1_{ij}$, respectively $\tilde{e}^2_{ij} = \bar{e}^2_{ij} + \hat{e}^2_{ij}$, it is:

$$a_{ij} = \begin{cases} \left(\bar{e}^1_{ij} \cdot \bar{e}^2_{ij}\right) = \bar{e}^1_{ij} \cdot \bar{e}^2_{ij} & \text{if neither } e^1_{ij} \text{ nor } e^2_{ij} \text{ are deviating,} \\ \left(\left(\bar{e}^1_{ij} + \hat{e}^1_{ij}\right) \cdot \bar{e}^2_{ij}\right) = \bar{e}^1_{ij}\bar{e}^2_{ij} + \hat{e}^1_{ij}\bar{e}^2_{ij} & \text{if deviation happens only in } e^1_{ij}, \\ \left(\bar{e}^1_{ij} \cdot \left(\bar{e}^2_{ij} + \hat{e}^2_{ij}\right)\right) = \bar{e}^1_{ij}\bar{e}^2_{ij} + \bar{e}^2_{ij}\hat{e}^2_{ij} & \text{if deviation happens only in } e^2_{ij}, \\ \left(\bar{e}^1_{ij} + \hat{e}^1_{ij}\right) \cdot \left(\bar{e}^2_{ij} + \hat{e}^2_{ij}\right) & \\ = \bar{e}^1_{ij}\bar{e}^2_{ij} + \bar{e}^1_{ij}\hat{e}^2_{ij} + \hat{e}^1_{ij}\bar{e}^2_{ij} + \hat{e}^1_{ij}\hat{e}^2_{ij} & \text{if deviation occurs in both events.} \end{cases} \tag{1.29}$$

Exploiting this, we formulate the following corollary.

Corollary 1.2. *The two-source robust problem* (1.28a)–(1.28c) *can be written as*

$$\min \quad c^t x \tag{1.30a}$$

$$\text{s.t.} \quad \sum_{j \in \mathcal{J}} \bar{e}^1_{ij}\bar{e}^2_{ij}x_j + \max_{\substack{Q_1 \subseteq \mathcal{S}^1_i \\ Q_2 \subseteq \mathcal{S}^2_i \\ |Q_1| \leq \Gamma_1 \\ |Q_2| \leq \Gamma_2}} \left\{ \sum_{j \in Q_1} \hat{e}^1_{ij}\bar{e}^2_{ij}x_j + \sum_{j \in Q_2} \bar{e}^1_{ij}\hat{e}^2_{ij}x_j + \sum_{j \in Q_1 \cap Q_2} \hat{e}^1_{ij}\hat{e}^2_{ij}x_j \right\} \leq b_i \ \ \forall\, i \in \mathcal{I} \tag{1.30b}$$

$$x \geq 0. \tag{1.30c}$$

Similar to the case of the Γ-robustness, we have

Theorem 1.6. *Consider the two-source robust Problem* (1.30a)–(1.30c). *It holds that*

- *Each solution is deterministically feasible, if, per row i, at most Γ_1 many events deviate within the event class S^1_i and at most Γ_2 many events deviate within the event class S^2_i.*
- *The problem can be compactly reformulated as a MILP.*

Proof. The first statement is clear by construction. We prove the second one. As for the bijective case, the supremum operator can be replaced by the maximum operator as the functions A_{ij} are continuous and have bounded and closed domains.

Then, for any fixed x^*_j the subproblem in Constraint (1.30b) can be formulated as a mixed integer linear program. We carry out the reformulation for a fixed row $i \in \mathcal{I}$ (and omit the index i for readability in the following).

We use three types of variables: $\alpha_j \in \{0,1\}$ equals one if $\tilde{e}^1_{ij} = \bar{e}^1_{ij} + \hat{e}^1_{ij}$ and $\beta_j \in \{0,1\}$ equals one if $\tilde{e}^2_{ij} = \bar{e}^2_{ij} + \hat{e}^2_{ij}$ and zero, otherwise. Similarly, $\gamma_j \in \{0,1\}$ equals one if $\alpha_j = \beta_j = 1$ and zero, otherwise. I.e., it equals one, if and only if both events deviate from their expectations. Thus, denoting the dual variables of the LP relaxation in

brackets on the right, we can write the subproblem as

$$\max \quad \sum_{j\in\mathcal{J}} \left(\hat{e}^1_{ij}\bar{e}^2_{ij}\alpha_j + \bar{e}^1_{ij}\hat{e}^2_{ij}\beta_j + \hat{e}^1_{ij}\hat{e}^2_{ij}\gamma_j\right) x^*_j \tag{1.31a}$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{J}} \alpha_j \le \Gamma_1 \qquad [\pi^1] \tag{1.31b}$$

$$\sum_{j\in\mathcal{J}} \beta_j \le \Gamma_2 \qquad [\pi^2] \tag{1.31c}$$

$$\gamma_j \le \alpha_j \le 1 \quad \forall\, j \in \mathcal{J} \qquad [\rho, \tau] \tag{1.31d}$$

$$\gamma_j \le \beta_j \le 1 \quad \forall\, j \in \mathcal{J} \qquad [\sigma, \nu] \tag{1.31e}$$

$$\alpha_j, \beta_j, \gamma_j \in \{0, 1\}\,. \tag{1.31f}$$

We show that the constraint matrix of (1.31a)–(1.31f) is totally unimodular. As appending the identity matrix does not impact the total unimodularity property, we neglect the constraints $\alpha_j \le 1$, $\beta_j \le 1$, and $\gamma_j \le 1$. Clearly, there are only two non zero entries in $\{1, -1\}$ per column. We partition the rows into two classes, M_1 and M_2, by defining $M_1 := \{(1.31b), (1.31d)\}$ and $M_2 := \{(1.31c), (1.31e)\}$. If two entries of a single column have different signs, either both are in M_1 or both are in M_2. If both signs are equal, one is in M_1 and one is in M_2. The total unimodularity follows as is shown in the work by Heller and Tompkins [72]. Hence, we can relax integrality constraints for the subproblem and still obtain integer solutions. Since the primal is feasible and bounded, relaxing integrality, (strong) dualization yields

$$\min \quad \Gamma_1\pi^1 + \Gamma_2\pi^2 + \sum_{j\in\mathcal{J}} \tau_j + \sum_{j\in\mathcal{J}} \nu_j \tag{1.32a}$$

$$\text{s.t.} \quad \pi^1 - \rho_j + \tau_j \ge \hat{e}^1_{ij}\bar{e}^2_{ij}x^*_j \qquad \forall\, j \in \mathcal{J} \tag{1.32b}$$

$$\pi^2 - \sigma_j + \nu_2 \ge \bar{e}^1_{ij}\hat{e}^j_{ij}x^*_j \qquad \forall\, j \in \mathcal{J} \tag{1.32c}$$

$$\rho_j + \sigma_j \ge \hat{e}^1_{ij}\hat{e}^2_{ij}x^*_j \qquad \forall\, j \in \mathcal{J} \tag{1.32d}$$

$$\pi^1, \pi^2, \rho_j, \tau_j, \sigma_j, \nu_j \ge 0. \tag{1.32e}$$

We obtain a compact formulation of Constraint (1.30b) as MILP by inserting the above duals into the corresponding constraints. Adding a row index $i \in \mathcal{I}$, we obtain a compact reformulation of the robust problem as

$$\min \quad c^t x \tag{1.33a}$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{J}} \bar{e}^1_{ij}\bar{e}^2_{ij}x_j + \Gamma_1\pi^1_i + \Gamma_2\pi^2_i + \sum_{j\in\mathcal{J}} \tau_{ij} + \sum_{j\in\mathcal{J}} \nu_{ij} \le b_i \qquad \forall\, i \in \mathcal{I} \tag{1.33b}$$

$$\pi^1_i - \rho_{ij} + \tau_{ij} \ge \hat{e}^1_{ij}\bar{e}^2_{ij}x^*_j \qquad \forall\, i \in \mathcal{I}, j \in \mathcal{J} \tag{1.33c}$$

$$\pi^2_i - \sigma_{ij} + \nu_{ij} \ge \bar{e}^1_{ij}\hat{e}^j_{ij}x^*_j \qquad \forall\, i \in \mathcal{I}, j \in \mathcal{J} \tag{1.33d}$$

$$\rho_{ij} + \sigma_{ij} \ge \hat{e}^1_{ij}\hat{e}^2_{ij}x^*_j \qquad \forall\, i \in \mathcal{I}, j \in \mathcal{J} \tag{1.33e}$$

$$\pi^1_i, \pi^2_i, \rho_{ij}, \tau_{ij}, \sigma_{ij}, \nu_{ij} \ge 0 \tag{1.33f}$$

$$x \ge 0. \tag{1.33g}$$

□

As for the bijective case, by Theorem 1.6, we can solve the two-source robust problem as LP, respectively as MILP. Again we formulate an alternative way to solve it.

Remark 1.4. *We can reformulate the two-source robust Problem* (1.30a)–(1.30a) *as*

$$\min \quad c^t x \tag{1.34a}$$

$$\text{s.t.} \quad \sum_{j\in\mathcal{J}} \tilde{e}^1_{ij}\tilde{e}^2_{ij}x_j \le b_i \quad \begin{array}{l} \forall\, i \in \mathcal{I}, \forall j \in \mathcal{J}, \forall k \in \{1,2\}, \\ \tilde{e}^k_{ij} \in [\bar{e}^k_{ij} - \hat{e}^k_{ij}, \bar{e}^k_{ij} + \hat{e}^k_{ij}] : \\ \left|\left\{\tilde{e}^k_{ij} \neq \bar{e}^k_{ij} \mid e^k_{ij} \in \mathcal{S}^k_i\right\}\right| \le \Gamma_k \end{array} \tag{1.34b}$$

$$x \ge 0. \tag{1.34c}$$

The problem has an infinite number of constraints. However, we can solve the Problem (1.15a)–(1.15c) *(with* $a_{ij} = \bar{e}^1_{ij}\bar{e}^2_{ij}$*) and add Constraint* (1.34b) *in a separation routine. In order to do so, given a solution* x^**, we can employ Problem* (1.31a)–(1.31e) *as separation problem. Since the constraint matrix is totally unimodular, it can be solved in polynomial time.*

We conclude the subsection by a comparison of the two-source robustness with the Γ-robustness. Therefore, we need the following definition.

Definition 1.12 (Two-source and Γ-robustness correspondence)**.** *Let* $\Gamma = \Gamma_1 = \Gamma_2 \ge 0$ *be fixed. Let a two-source robust problem be given. For this problem, we call the* Γ*-robust problem where each coefficient* a_{ij} *is determined by an event* e^Γ_{ij} *with*

$$\tilde{e}^\Gamma_{ij} \in \left[\bar{e}^1_{ij}\bar{e}^2_{ij} - \bar{e}^1_{ij}\hat{e}^2_{ij} - \hat{e}^1_{ij}\bar{e}^2_{ij} - \hat{e}^1_{ij}\hat{e}^2_{ij}, \bar{e}^1_{ij}\bar{e}^2_{ij} + \bar{e}^1_{ij}\hat{e}^2_{ij} + \hat{e}^1_{ij}\bar{e}^2_{ij} + \hat{e}^1_{ij}\hat{e}^2_{ij}\right] \tag{1.35}$$

the ***corresponding*** Γ*-robust problem.*

Relying on this correspondence, we derive that a two-source robust solution is at least as protected as the corresponding Γ-robust solution.

Corollary 1.3. *Consider a two-source robust problem and its corresponding* Γ*-robust problem. Then, any solution for the two-source robust problem is at least as protected as a solution of the* Γ*-robust problem.*

Proof. Let x^* be a solution to Problem (1.15a)–(1.15c) (with $a_{ij} = \bar{e}^1_{ij}\bar{e}^2_{ij}$). If x^* is not feasible for the Γ-robust problem, then, by Constraint (1.21b), there exists $i \in \mathcal{I}$, and a set $S^* \subseteq \mathcal{J}$ with $|S^*| \le \Gamma$ such that

$$\sum_{j\in\mathcal{J}} \bar{e}^1_{ij}\bar{e}^2_{ij}x^*_j + \sum_{j\in S}\left(\bar{e}^1_{ij}\hat{e}^2_{ij} + \hat{e}^1_{ij}\bar{e}^2_{ij} + \hat{e}^1_{ij}\hat{e}^2_{ij}\right)x^*_j > b_i \tag{1.36a}$$

$$\Leftrightarrow \quad \sum_{j\in\mathcal{J}} \bar{e}^1_{ij}\bar{e}^2_{ij}x^*_j + \sum_{j\in S}\bar{e}^1_{ij}\hat{e}^2_{ij}x^*_j + \sum_{j\in S}\hat{e}^1_{ij}\bar{e}^2_{ij}x^*_j + \sum_{j\in S}\hat{e}^1_{ij}\hat{e}^2_{ij}x^*_j > b_i. \tag{1.36b}$$

Now, the latter inequality yields a violated Constraint (1.30b). So, the solution is also infeasible for the two-source robust problem. □

The relation between the two corresponding robust problems can be strict:

Remark 1.5. *Consider a constraint with two variables,* $a_1x_1 + a_2x_2 \leq 28$. *Let the coefficients* a_1, *respectively* a_2, *for the two-source robust problem be determined by the events* e_1^1 *and* e_1^2, *respectively by* e_2^1 *and* e_2^2, *with*

$$\tilde{e}_1^1 \in [20-10, 20+10], \qquad \tilde{e}_1^2, \in [0.5-0.1, 0.5+0.1] \tag{1.37a}$$

$$\text{and} \quad \tilde{e}_2^1 \in [0.5-0.1, 0.5+0.1], \qquad \tilde{e}_2^2 \in [20-10, 20+10]. \tag{1.37b}$$

Both coefficients for the corresponding Γ*-robust problem take values in* $[10-8, 10+8]$. *Let* $\Gamma = \Gamma_1 = \Gamma_2 = 1$. *This way, the solution* $x_1 = x_2 = 1$ *is feasible for the* Γ*-robust problem. However, it is not feasible for the two-source robust problem, as* $\tilde{e}_1^1 = \tilde{e}_2^2 = 30$ *and* $\tilde{e}_1^2 = \tilde{e}_2^1 = 0.5$ *is a valid realization of the coefficients, leading to the inequality*

$$15x_1 + 15x_2 \leq 28 \tag{1.37c}$$

violated by $x_1 = x_2 = 1$. *Hence, the two-source robust problem is more restricted than the* Γ*-robust problem and therefore, the resulting solution has a better protection.*

CHAPTER 2

Network design with compression

In this chapter, we focus on the Network Design Problem with Compression (NDPC). NDPC is an extension of the NDP problem as presented in Chapter 1, where the costs of the infrastructure, for example prices which are to be paid for used edge capacities, are induced by the energy consumption of the components. The chapter is structured into seven sections.

We start with an introduction to the problem in Section 2.1. There, we motivate the importance of NDPC to telecommunication applications, especially with respect to green networking, and present a brief review on the literature on NDPC.

In Section 2.2, we give a formal definition of the NDPC problem, and show how the problem and some of its variations can be modeled as MILP.

Subsequently, in Section 2.3, we investigate the polyhedron of the MILP formulation. In the first subsection, we discuss the dimension and the trivial facets of the polytope. In the following, we present theoretical results, in particular focusing on valid inequalities, from which we point out two classes of facets. We conclude by showing how cut based inequalities can be employed in a separation approach.

In Section 2.4, the theoretical difficulty of NDPC, especially with respect to the compression functionality, is analyzed. We show that NDPC is, in a sense, "more difficult" than the NDP problem by introducing the Compressor Placement Problem (CPP). Furthermore, we present some special cases, i.e., tree and path instances, for which (pseudo-) polynomial algorithms can be derived.

In Section 2.5, we consider the NDPC problem under data uncertainty. Data uncertainty can be incorporated into the optimization problem by relying on the robustness concepts as presented in Section 1.4. We present the corresponding reformulations for the NDPC problem in detail and discuss the different concepts.

Finally, in Section 2.6, we evaluate the NDPC problem computationally. In particular, this entails a comparison of NDP and NDPC, an evaluation of the polyhedral results, i.e., insights into the effects of employing additional cutting planes, and an evaluation of the NDPC problem under data uncertainty.

We conclude this chapter with a brief summary of our results and an outlook into further research directions.

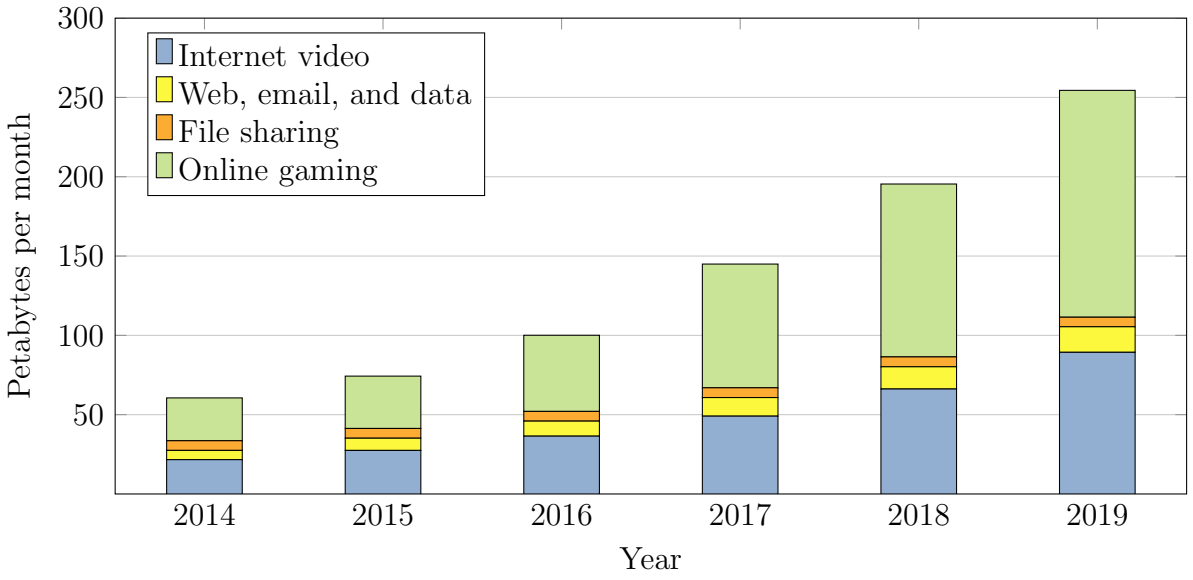


Figure 2.1: The fore-casted global consumer Internet traffic between 2014 and 2019 by Cisco Systems Inc. [37], categorized into Internet video, Web, email, and data, File sharing, and Online gaming.

Previous publications

Some results presented in this chapter were produced in collaboration with different co-authors and have been published before. In particular, similar results to those of Section 2.3 have been presented and analyzed by Koster, Phan, and Tieves [86]. Partial results on the complexity of NDPC as presented in Section 2.4 have also been published before, by Koster and Tieves [84]. With respect to data uncertainty, see Section 2.5, the concept of Γ-robustness was employed to tackle uncertainty in the compression factor by Coudert, Koster, Phan, and Tieves [43].

2.1 Introduction

2.1.1 A motivation for network design with compression

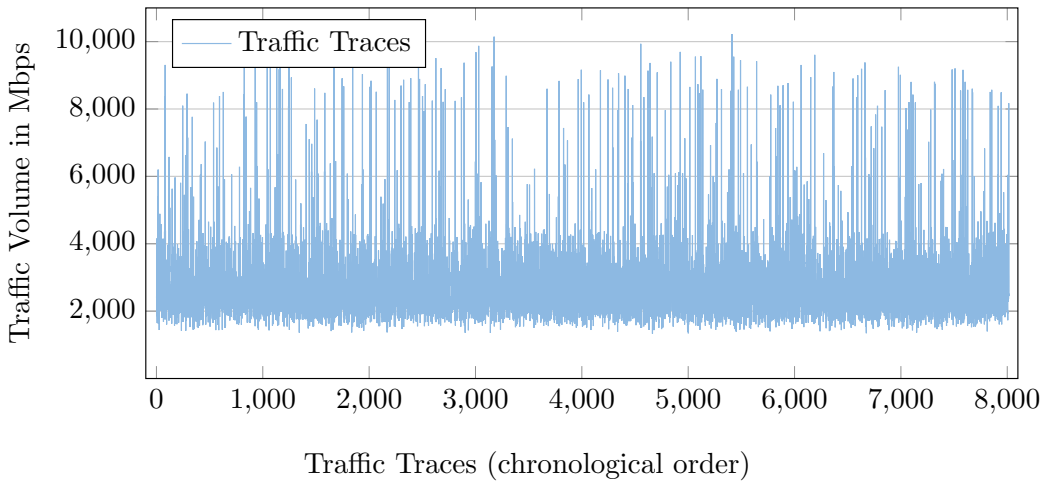
Telecommunication services and especially services associated with the Internet are one of the central aspects in our society. Apparently, the importance and the sheer amount of such services has been rapidly growing over the last decades and is expected to do so in the future as well. Figures supporting this claim can be found in many works within the corresponding literature. Here, we just point the reader to two of them. One is a report by Cisco Systems Inc. [37] in which it is stated that "Global Internet Protocol (IP) traffic has increased more than fivefold in the past 5 years, and will increase nearly threefold over the next 5 years. Overall, IP traffic will grow at a compound annual growth rate (CAGR) of 23 percent from 2014 to 2019" and "Global Internet traffic in

2019 will be equivalent to 64 times the volume of the entire global Internet in 2005. Globally, Internet traffic will reach 18 gigabytes (GB) per capita by 2019, up from 6 GB per capita in 2014.". Another interesting source is from Greentouch [65], in which the authors estimate that, from 2010 to 2020, the global (wire line) Internet traffic will increase by a factor of 16 (to 250 exabytes per month). For a visualization of this growth, we refer to Figure 2.1. This figure concerns the fore-casted global consumer Internet traffic (not confined to a single service provider's network) which grows in a similar manner as the global Internet traffic. It hsows that the traffic growth between 2014 and 2019, categorized into Internet video, Web, email, and data, File sharing, and Online gaming. Interestingly, the growth is mainly carried by the categories Internet video (e.g., YouTube, Netflix) and Online Gaming. These areas, in particular streaming services, are accounted to be most innovative, more and more popular, and require more demanding (with respect to traffic volumes) Internet services than ever before. In this context, see Hartley [70] for an analysis of YouTube's traffic.

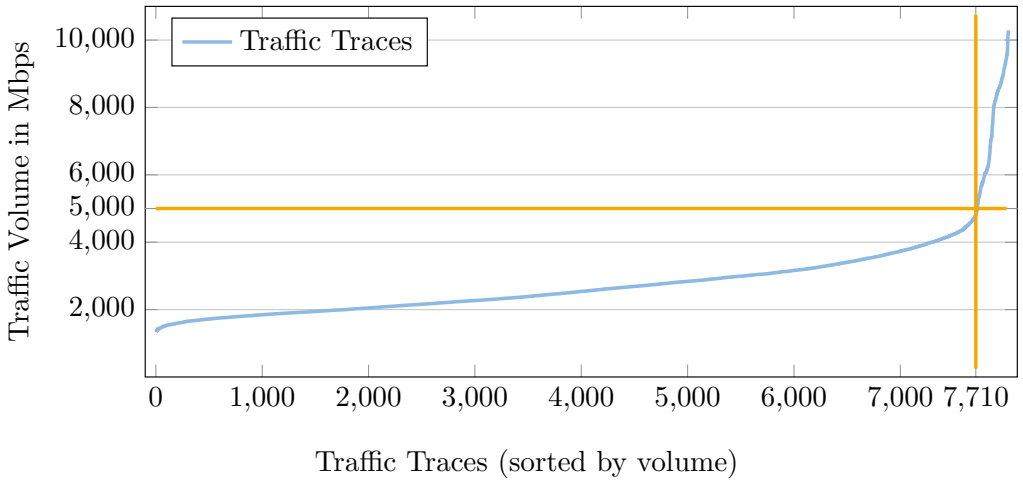
Naturally, such growth must be matched with the underlying infrastructures of these services, e.g., within the backbone networks or data centers. In the remaining work, we will refer to such infrastructure as the *communication network* or short – the network. As a consequence, it is expected that the CO_2 emission produced by the Information and Communication Technology will increase from 2% to 10% of the total man made CO_2 emission, see Lubritto et al. [89]. This problematic trend is enhanced by the conventional network design principles. In general, many networks have been designed and are operated in a way that offers the maximum reliability for their accommodated traffic. Among other implications, this means that such networks are usually designed to be able to carry the peak loads of the expected traffic plus some additional safety measure. Consequently, many networks are hugely overcapacitated and contain many components which are redundant if the network only has to sustain a smaller load. For a visualization of the dimension of such over-provisioning, we refer to Example 2.1.

Example 2.1. *Consider the* ABILENE *network as presented in Figure 1.1. In the SNDLIB [100], 48096 real-life traffic traces, each covering 5 minutes, are specified for the* ABILENE *network. In Figure 2.2, we plot the total traffic load specified in the first 8,000 traces, in sub-figure a) as occurring over time, and in b) sorted by volume. As we can see, the total traffic fluctuates heavily. The minimum is below 2,000 units (in Mbps) and the maximum above 10,000. Hence, a network which accounts for the peak value (in the sense of the NDP problem) is substantially over-sized if smaller load occurs. For comparison, with respect to the 8,000 traces, in 7,710 cases, the traffic load is below 5,000 units, indicating that in 77% of the time, the network is heavily over-sized. Note that this estimation does not account for an additional safety measure on top of the estimated peak load, which would further increase the over-provisioning of capacities.*

The above gives rise to the topics of *green networking*, i.e., to topics that deal with the reduction of the energy consumption of the involved services because of economical and environmental reasons. In this context, the NDP problem (see Definition 1.7) is one of the most basic problems where the relation between traffic (increase) and the corresponding infrastructure can be modeled. That is, in the NDP problem, we are tasked

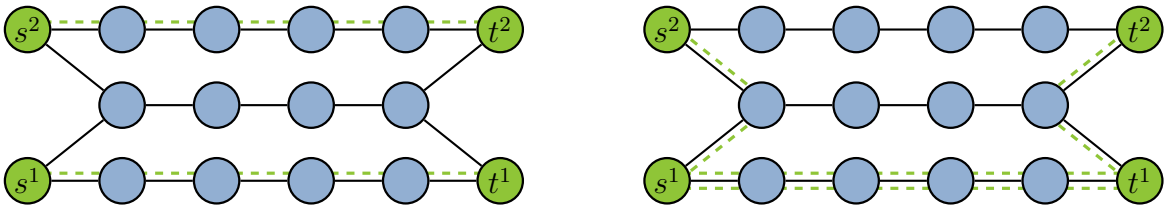


(a) Traffic load per time interval.



(b) Traffic load (blue) sorted by size. The orange lines indicate that 7710 traces have a load of less than 5000 Mbps.

Figure 2.2: The load of the ABILENE network over discretized time intervals as specified in the SNDLIB [100]. Each interval defines a traffic traces, i.e., an aggregated traffic volume in Megabytes per second, for a certain time frame of five minutes. In a), the traffic volumes as they appear over time, in b) the occurring traffic loads sorted by volume.



(a) NDP with shortest-path routing, ten edge modules required

(b) Classic NDP, nine edge modules required

Figure 2.3: Different routing schemes of a NDP instance and the thereof required amount of edge modules.

to capacitate a given network topology in such a way that all requested commodities can be routed at minimum cost. When considering the prices for the link capacities as prices related to the energy consumption of the installed modules, in this context we say activated, we directly observe the relevance of the NDP problem for green networking. We point out that the NDP problem facilitates *energy-aware* routing (EAR), since the energy consumption of the utilized links is minimized.

With respect to the different versions of the NDP problem (see Section 1.3), we give the following remark.

Remark 2.1. *As pointed out in Section 1.3, it is computationally attractive to restrict the NDP problem to shortest path routing. However, this restriction can severely increase the energy consumption, i.e., the amount of necessary capacity installations. A visualization of this is given in Figure 2.3(a) and Figure 2.3(b). In this figure, we consider a network* $G = (V, E)$ *with two commodities,* $q^1 := (s^1, t^1)$ *and* $q^2 := (s^2, t^2)$ *and* $d^{q^1} = d^{q^2} = 1$*. Assume that* $k_{uv} = 1$ *for all* $uv \in E$*, except for the lowest horizontal path* $(s^1, \ldots, t^1)$*, for which we assume that* $k_{uv} = 2$*. We observe that a "naive" shortest path routing requires ten capacity modules, see Figure 2.3(a), while the non restricted NDP problem produces a solution which employs at most nine capacity modules, see Figure 2.3(b). Since the non-restricted NDP problem offers much more flexibility to route multiple flows on the links, and hence, uses their capacity much more efficiently, this version of the NDP problem is preferable from a green networking perspective.*

In this chapter, we do not consider the (classic) NDP problem, but an extended version which is called the *Network Design Problem with Compression* (NDPC). NDPC arises from NDP by introducing so called *compression* and *decompression* devices. Differently to the edge capacity modules, these can only be installed in the nodes of the network. We motivate the NDPC problem and explain the functionality of the newly introduced devices in the following example.

Example 2.2. *Consider a telecommunication network and let us assume that some content, for example a popular movie, is requested repeatedly. In detail, we assume that two commodities* $q_1, q_2 \in Q$ *include this movie and that both commodities are routed over*

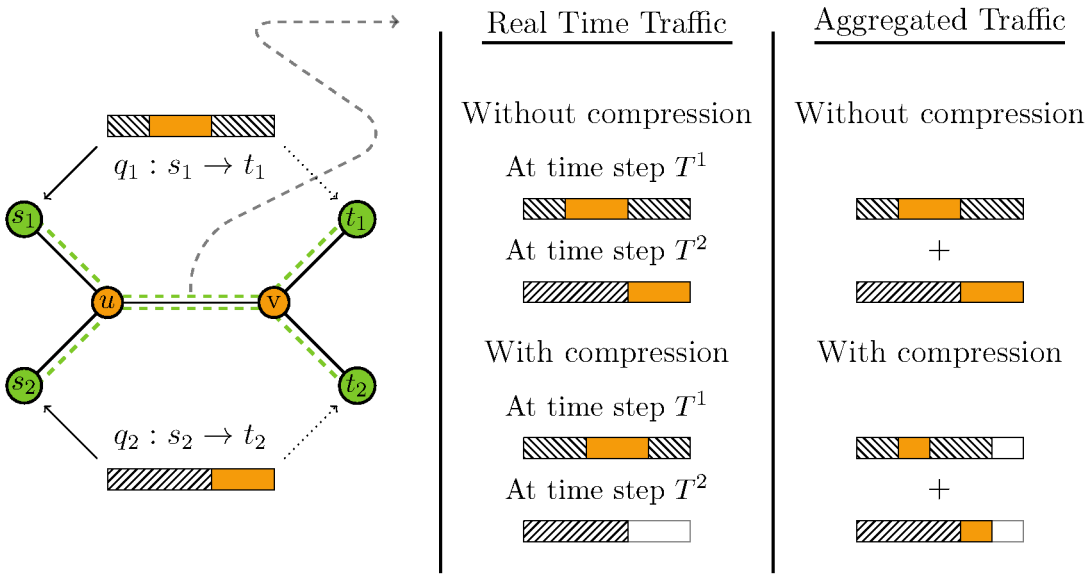


Figure 2.4: Left, a communication network with two commodities q_1 and q_2. Right, the capacity requirement on the edge uv. q_1 and q_2 have a fraction of the traffic in common (orange), and share the central edge uv in their routing paths. In real time, q_1 is transmitted first. If compression is enabled, the common traffic in q_2 does not need to be sent over uv a second time. It is erased from the traffic stream in u and restored back in v. From an aggregated point of view, this reduces the total traffic volume by a certain fraction.

a common arc uv. We assume that q_1 is transmitted a short time before q_2 is. If a compressing device is active in node u, it scans the passing traffic and, at some point in time, it registers the traffic of q_1, i.e., it stores all content locally. The same happens at the node v if a decompressing device is active there.

Afterwards, when the traffic of q_2 traverses node u, the compressor recognizes that some content (the movie) is requested for a second time. This means it "knows" that the content can be restored from memory, somewhere later in the routing path, such that it is not necessary to sent the content again. Therefore, it replaces the movie by a short code (identifier), effectively reducing the data volume of the traffic stream and sends the reduced data stream.

When the reduced data stream arrives at node v, the decompressing device recognizes the code, restores the movie from memory, and replaces the code with the original content. This way, capacity is saved on the arc uv. The example is visualized in Figure 2.4. During the course of this work, we refer to this caching procedure as compression, respectively, decompression. Note that in other works (see the next subsection), this caching scheme is sometimes referred to as redundancy elimination.

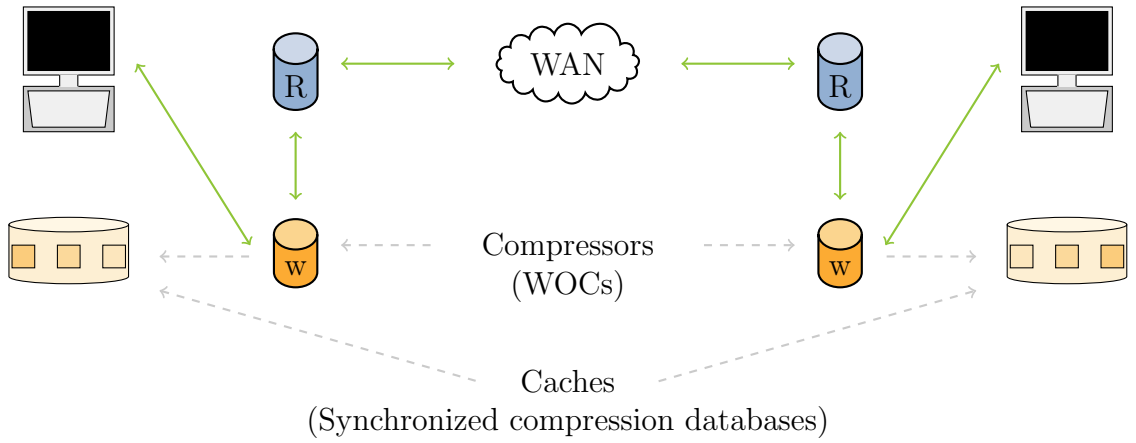


Figure 2.5: We present a similar picture as the one provided by Phan [102]: In the network, the computers symbolize a traffic commodity. The commodity enters and leaves the network via routers (blue). After each router, it is sent through a compressor (WOC), indicated in orange. In these, it could be (de-) compressed, relying on the storage system of the compressors.

For the NDPC problem, we typically consider large scale *backbone* networks or, so-called *wide-area networks* (WAN). Traffic, respectively the commodities, enters such network at a certain *access point* (*router*) and is sent to another router and from there to its destination. In each of these routers, a WAN Optimization Controller (WOC), which is a device to enable a caching scheme as described in Example 2.2, can be installed (see the next subsection for additional information on such WOCs). Then, all traffic which passes the router is sent through the WOC before it is sent to any other router and can hence be compressed or decompressed between two such WOCs. We refer to Figure 2.5 for a visualization of such network.

In the following, we will refer to a WOC as a compressor, respectively a decompressor. For simplicity, we assume that each de-/ compressor has sufficient memory to compress all traffic in the network, i.e., the situation that such device is "full" and can therefore not store any additional data will not occur.

In the NDPC problem, the additionally introduced devices allow to compress and/or to decompress *all* traffic which passes through a node. That is, we assume that all the traffic in the network is aggregated, i.e., we do not consider different time frames, but assume that all the data has to be sent simultaneously. From this perspective, we assume that every (aggregated) commodity $q \in Q$ comes with a ratio $1 < \gamma^q$ which tells us the fraction $(\frac{1}{\gamma^q})$ of the traffic which is unique, i.e., how much of the data remains to be transmitted, if the traffic stream is compressed. This way, a commodity requires d^q units of flow if it is sent uncompressed but only $\frac{d^q}{\gamma^q}$ units of flow if it is sent compressed.

Since compressed traffic requires less capacity than its not compressed counterpart, this compression functionality virtually increases the edge capacities at the expense of

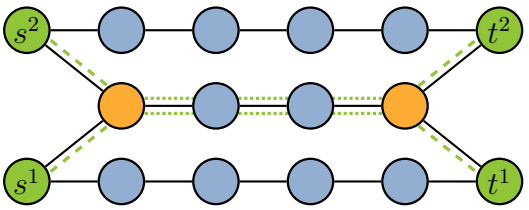


Figure 2.6: Routing and the thereof required compressors and edge-modules in an instance for network design with compression. The solution requires two compressors and seven edge modules.

the additional objective (energy) costs of these devices. Hence, an optimizer has to decide how many of these devices are to be employed and where to locate them, so that more arc capacity cost can be saved as additional compressor costs occur. For a formal definition of the NDPC problem, we refer to Section 2.2.

We assume that each commodity may only be compressed once. Each commodity $q \in Q$ is available at its source s^q uncompressed but can immediately be compressed if a compressor is active there. Correspondingly, the commodity q has to arrive at its target t^q uncompressed, so that it has to be decompressed in t^q at latest. In this case, a decompressor has to be active in t^q. We assume that the same physical device can perform both the compression and the decompression, and that a single device can (de-) compress an unlimited amount of traffic. Hence, it is sufficient to install a single device in any node and, throughout this work, we will refer to such device as *compressor*.

We conclude this section with the following remark, indicating the potential improvement (with respect to energy consumption) of compression in a NDP problem.

Remark 2.2. *Consider the same situation as in Remark 2.1 and recall that the solution of the corresponding NDP problem requires nine capacity modules. Let us assume that both the demanded traffic volume of q^1 and that of q^2 can be compressed by 50%. Then, by employing two compressors (the orange nodes), the traffic can be routed at the expense of the two compressors and only seven capacity modules, see Figure 2.6. Clearly, in case that the two compressors are cheaper than the two capacity modules, the new solution, i.e., that of the* NDPC *problem, is preferable.*

2.1.2 Literature

Green Networking

Recent data confirms that the energy consumption of the Internet and its related Information and Communication Technologies (ICTs) can no longer be ignored, considering its increasing pervasiveness and the importance of the sector on productivity and economic growth. It is estimated that the ICT sector alone is responsible for a percentage which varies between 2% and 10% of the total world energy consumption, as stated by Lubritto et al. [89]. Contrarily to the common belief that sees the Internet and its

related ICTs as environmentally friendly, recent studies show that they are becoming significant contributors to global warming. Indeed, they are responsible for 2% (0,8 Gt CO2) of the annual global greenhouse gas emissions, the so-called carbon footprint, see Forster et al. [58], a number which exceeds the greenhouse gas emissions of the whole aviation sector, for a reference see Marsan et al. [94]. These percentages are likely to grow in the next years to around 1.4 Gt of CO2 by 2020, that is, to approximately 2.8% of the global world emissions as stated by Vereecken et al. [121].

As highlighted by Pickavet et al. [103], the total power consumption due to networking equipment is, with about 25 GW, in the same order of magnitude as the power consumption due to data centers or personal computers. Since the bandwidth requirements of networking applications are doubling every 18 months, see D'Ambrosia [46], the total power consumption due to network equipment will grow as well, reaching, according to a predicted annual growth rate of 12%, a total of 95 GW in 2020, see Vereecken et al. [121]. Clearly, this power consumption is problematic, one the one hand from an environmental perspective and, on the other hand, simply because of economical reasons.

Since the seminal work by Gupta and Singh [67], the research community has started to develop technologies for manufacturing energy-efficient network devices and energy management strategies for reducing the overall energy consumption of communication networks. The field covering these topics is usually referred to as energy-aware (or green) networking. We refer to Bianzino et al. [19], Bolla et al. [22], and Zeadally et al. [127] for three recent surveys on the topic.

Network Design

With respect to green networking, energy management, which refers to network management through energy-aware topology optimization and routing, is one of the central tasks for infrastructure providers. Hereby, energy management relates to the fact that networks are designed and dimensioned to serve the estimated peak traffic demand to avoid network congestion and ensure the quality of the service. This way, network protection mechanisms, i.e., over-dimensioning of the network's capacity, ensure that, even though traffic load varies remarkably over time, the network is usually run largely below network capacity, even during peak hours. Unfortunately, current network device architectures and transmission technologies make their power consumption almost independent of the traffic load or, at least, very high even at a low throughput. As a result, they consume as if they were always fully loaded and consequentially, energy-aware networks are designed to contain as few devices as possible. In this context, a widely adopted approach to improve the network energy performance is to design new hardware components that are capable of adjusting their power consumption according to the traffic load, see, for instance, Bolla et al. [21]. However, a coordinated strategy that reduces the overall energy consumption of the network by switching-off, that is, putting into sleep mode, nodes (routers) and links (line cards), and rerouting the traffic through the remaining portion of the network, is likely to achieve a better performance. Another topic in energy management concerns the routing protocol, as apparently, this

protocol directly influences the utilization of the different devices of the network.

Optimization problems related to network design and routing have been widely investigated during the past twenty years. A survey covering results which appeared up to year 2000 is given by Yuan [126], while we refer to Pióro and Medhi [104] and to Resende and Pardalos [108] for surveys on more recent contributions. Within the context of this work, we refer to the NDP problem as an optimization problem which can take both, the networking infrastructure and the routing paradigm into account. For additional information on the problem, we refer to the Subsection 2.1.1.

Network Design with Compression

The NDPC problem is an extension of the NDP problem. With the introduction of the so called compression devices, the NDPC problem allows to design more energy-efficient networks than the network design problem. To our knowledge, the first algorithmic publication on the NDPC problem was given by Giroire et al. [63]. Afterwards, the topic received great interest within the research community, leading to a series of subsequent publications by a broad range of authors. The published works cover the mathematical foundation of the NDPC problem, see Koster et al. [86], and its computational complexity, see Giroire et al. [62] and Koster and Tieves [84]. Furthermore, there are publications by Coudert et al. [43, 44] covering data uncertainty concepts for NDPC, especially with respect to the Γ-robustness of Bertsimas and Sim [17, 18]. In this context, the author of this thesis contributed to the works of Koster et al. [86], Koster and Tieves [84] and Coudert et al. [43]. Concluding this paragraph, we point the reader to the PhD thesis of Phan [102] in which the NDPC problem is described in great (technical) detail and where, from a practical perspective, various solution approaches are presented.

Technical background

We give a short introduction into the technical background of the compression aspect. At first, we refer to the publication of Chabarek et al. [29] where it is shown that the energy consumption of network components is not proportional to the degree of their usage. That is, a capacity module uses (nearly) the same energy, independently whether its full capacity is utilized or not. Consequentially, the NDP problem and the NDPC problem can be used to model a routing of the commodities on the "least possible" amount of infrastructure, hence saving the energy consumption induced by the omitted components. We refer to Idzikowski [77] and to Fisher et al. [55] for more detailed information on that matter. We further refer to Chabarek et al. [29] and to Anand et al. [8], where the authors estimate that the amount of redundant traffic in a network is in the range of 15-60%, stressing the relevance of the NDPC problem.

For a technical analysis of the compression functionality, we refer to the work of Anand et al. [7]. We point out that the compressors are often referred to as WOC (Wide Area Network Optimization Controller) and that they operate on packet-level in the network, i.e., they can operate protocol independent. Again, a complete survey on the technical

background of the different devices is out of scope of this work. For two examples, we refer to the vendors, [110] and [20]. For a more detailed survey on this matter, we refer to the work of Phan [102].

2.2 Formalizing network design with compression

2.2.1 Notation and definitions

We introduce some additional notation, closely following the notation employed for the NDP problem in the previous chapter. In contrast to the NDP problem, we adopt an undirected graph as the telecommunication network.

We depict the telecommunication network as the undirected graph $G := (V, E)$ where the node set V describes routers and the edge set E represents the possible inter connections between these routers. Let Q be the set of commodities. For $q \in Q$, we denote by $d^q \geq 0$ the volume of the (uncompressed) traffic which has to be routed from the source node s^q to the sink node t^q. If not explicitly stated otherwise, we assume that $|Q| = |V|^2 - |V|$ and $d^q > 0$ for all $q \in Q$, i.e., that, for each node pair $v \neq w \in V$, there is exactly one commodity associated to it. As for the NDP problem, each commodity has to be routed and the total flow traversing an edge $uv \in E$ has to fit in the capacity on this edge. We assume a *splittable* routing scheme. This way, a traffic flow can be interpreted as a set of paths in G, connecting the source and the sink, each of them carrying a fraction of the total traffic volume. The capacity has to be installed in (potentially *multiple*) batches of size k_{uv} where each batch induces a cost of $c_{uv} \geq 0$.

We assume that, for any $q \in Q$, the traffic volume d^q can be compressed by a factor of $\gamma^q \in [1, \infty)$ (the *compression factor*). This way, the volume of a commodity q can be reduced to $\frac{d^q}{\gamma^q}$, if the total flow is compressed, i.e., if it traverses a node where a compressing device is active. Similarly, compressed traffic can be uncompressed again, if it passes an active compressor. Note that, since a commodity has to arrive at its sink uncompressed, it has to be uncompressed at latest in t^q. Of course, it is possible to compress only a certain part of the flow, in particular only that fraction of the commodity which is routed along a certain path in the network. Once a part of the flow is compressed, it can be decompressed if it traverses a node with an active compressor in it. In general, a compressor can be activated in any node $v \in V$. However, if a compressor is active in node $v \in V$, it induces a cost of $c_v \geq 0$, independently of the number of compressed traffic streams. It is not necessary to activate more than a single compressor per node.

In this situation, we are looking for a selection of edge capacities and compressors which allow all commodities to be routed from their source to their sink, with minimum objective cost. We extend the NDP problem, see Definition 1.7, to the definition of the *Network Design Problem with Compression*:

Definition 2.1 (NDPC)**.** *Let an undirected graph $G = (V, E)$, source and target nodes $S, T \subseteq V$, a capacity function*

$$k \ : \ E \ \to \mathbb{Z}_+, \quad uv \mapsto \quad k_{uv}, \tag{2.1a}$$

and two cost functions

$$c_E : E \to \mathbb{R}_+, uv \mapsto c_{uv}, \tag{2.1b}$$
$$c_V : V \to \mathbb{R}_+, u \ \mapsto c_u. \tag{2.1c}$$

be given. Denote Q the collection of (s, t) pairs with $s \in S$ and $t \in T$. For all $q \in Q$, the parameters $d^q \in \mathbb{R}_+$ and $\gamma^q \in [1, \infty)$ are given. The ***Network Design Problem with Compression*** *asks, for all $q \in Q$, for two functions f^q and g^q*

$$f^q : E \to \mathbb{R}_+, \quad uv \mapsto f^q_{uv} \qquad \forall q \in Q, \tag{2.1d}$$
$$g^q : E \to \mathbb{R}_+, \quad uv \mapsto g^q_{uv} \qquad \forall q \in Q, \tag{2.1e}$$

and for two functions x, y indicating the topology of the network

$$x : E \to \mathbb{Z}_+, \quad uv \mapsto x_{uv}, \tag{2.1f}$$
$$y : V \to \{0, 1\}, \ u \mapsto y_u \tag{2.1g}$$

satisfying the constraints

$$\sum_{q \in Q} \left(f^q_{uv} + \frac{1}{\gamma^q} g^q_{uv} + f^q_{vu} + \frac{1}{\gamma^q} g^q_{vu} \right) \leq k_{uv} x_{uv} \qquad \forall uv \in E \tag{2.1h}$$
$$\sum_{v \in \delta(u)} (f^q_{uv} + g^q_{uv}) = \sum_{v \in N(u)} (f^q_{vu} + g^q_{vu}) \qquad \forall q \in Q, u \in V \setminus \{s^q, t^q\} \tag{2.1i}$$
$$\sum_{s^q v \in E} (f^q_{s^q v} + g^q_{s^q v}) = d^q = \sum_{v t^q \in E} (f^q_{v t^q} + g^q_{v t^q}) \qquad \forall q \in Q \tag{2.1j}$$
$$-d^q y_u \leq \sum_{v \in N(u)} (g^q_{uv} - g^q_{vu}) \leq d^q y_u \qquad \forall u \in V, q \in Q \tag{2.1k}$$

such that

$$\sum_{uv \in E} c_{uv} x_{uv} + \sum_{u \in V} c_u y_u \tag{2.1l}$$

is minimized. We abbreviate the Network Design Problem with Compression as NDPC. We say that the tuple $(G, Q, \gamma^q, c_E, k, c_V)$ describes an instance of the NDPC *problem.*

For the NDPC problem, the Objective Cost (2.1l) is given by the installed capacity on the links and by the activated compressors. Constraint (2.1i) describes flow conservation, where for each commodity $q \in Q$, the flow is given by the sum of the corresponding f and g variables. Constraint (2.1h) describes link capacity constraints, subject to the

previously mentioned flows. Hereby, the compressed part of the flow (encoded in g^q) is scaled down by γ^q. Constraint (2.1j) enforces that for any commodity $q \in Q$, the corresponding flow value $val(f^q + g^q)$ equals the demand volume of q. Finally, for any commodity $q \in Q$, Constraint (2.1k) describes the relation between the uncompressed part of the flow f and the compressed part g. Any (inter) change between the traffic value of f and the traffic value of g may only occur, if a compressor is active in node $u \in V$.

We formalize the relation between the NDP problem and the NDPC problem as follows.

Remark 2.3. *Consider an instance of the* NDPC *problem. Assume that $c_u = M$ for all $u \in V$ and assume that M is sufficiently large. In any optimal solution, it is $y_u = 0$ for all $u \in V$ and hence $g^q_{uv} = g^q_{vu} = 0$ for all $q \in Q$ and $uv \in E$. Hence, the problem boils down to the NDP problem in absence of compression.*

Likewise, assume that $c_u = 0$ for all $u \in V$. Then, without loss of generality, in any optimal solution, it is $y_u = 1$ for all $u \in V$ and hence, $f^q_{uv} = f^q_{vu} = 0$ for all $q \in Q$ and $uv \in E$. Therefore, the problem can be reduced to the NDP problem, based on the g variables as its flow variables and $\frac{d^q}{\gamma^q}$ as the demand volumes.

We refer to these NDP problems as the two corresponding NDP problems for a given NDPC instance:

Definition 2.2. *Given a NDP problem, we write NDP_0 for the problem where $y_u = 0$ for all $u \in V$ and NDP_γ for the problem with $y_u = 1$ for all $u \in V$.*

In the course of this work, we will consider the NDPC problem in different variations. Therefore, we derive the following two special cases:

Definition 2.3 (Unit capacity NDPC)**.** *The tuple $(G, Q, \gamma^q, c_E, \mathbf{1}, c_V)$ describes an instance of the network design problem with compression where, independently of the edge $uv \in E$, each edge module provides exactly* ***one unit of capacity****.*

Definition 2.4 (Unit capacity and constant compression rate NDPC)**.** *Given $\gamma \geq 1$, the tuple $(G, Q, \boldsymbol{\gamma}, c_E, \mathbf{1}, c_V)$ describes an instance of the* NDPC *problem where, independently of the edge $uv \in E$, each edge module provides* ***one unit of capacity****, and where, for each commodity $q \in Q$, the* ***compression ratio is constant*** *($\gamma^q = \gamma$ for all $q \in Q$).*

In some works on the NDPC problem, see, e.g., by Coudert et al. [43] and by Koster et al. [86], a variant is considered, where *at most* one edge module can be employed per edge. As for the NDP problem, many different variants of the problem (compare Subsection 1.3.2) are possible. In this work, we focus on the above presented problems.

For any variant of NDPC (as we consider it here), it is immediately clear that:

Corollary 2.1. *Given a* NDPC *instance, there is an optimal solution with either zero or at least two compressors activated.*

PROOF. Let a solution to a NDPC problem be given and assume that $y_u = 1$ for exactly one $u \in V$. The compressed flow can only form a loop in G. Hence, the corresponding solution where all compressed flow is removed and the compressor is deactivated is feasible, as well. This solution is at least as cheap as the previous solution. □

2.2.2 Network design with compression as MILP

As is the case for the NDP problem, see Remark 1.1, a MILP formulation for the NDPC problem can be derived easily. The following remark gives this formulation. The MILP is derived from a formulation presented by Giroire et al. [63] by tightening the big-M coefficients and by modeling the flows as percentages of the total flow value instead of using absolute values. Such improved formulation was first shown by Koster et al. [86].

Remark 2.4. *We denote by $x_{uv} \in \mathbb{Z}_+$ the number of installed capacity modules on the edge $uv \in E$ and by $y_u \in \{0,1\}$ whether a compressor is activated in node $u \in V$. We write f^q_{uv} (f^q_{vu}) for the percentage of (uncompressed) flow of commodity $q \in Q$ being sent from u to v (v to u) on the edge $uv \in E$. Analogue, we write g^q_{uv} (g^q_{vu}) for the percentage of compressed flows. A MILP formulation of the* NDPC *problem is given by*

$$\min \quad \sum_{uv \in E} c_{uv} x_{uv} + \sum_{u \in V} c_u y_u \tag{2.2a}$$

$$\text{s.t.} \quad \sum_{v \in \delta(u)} (f^q_{uv} + g^q_{uv} - f^q_{vu} - g^q_{vu}) = \begin{cases} 1 \text{ if } u = s^q \\ -1 \text{ if } u = t^q \\ 0 \text{ else} \end{cases} \qquad \begin{matrix} \forall u \in V \\ \forall q \in Q \end{matrix} \tag{2.2b}$$

$$\sum_{q \in Q} d^q \left(f^q_{uv} + f^q_{vu} + \frac{1}{\gamma^q} g^q_{uv} + \frac{1}{\gamma^q} g^q_{vu} \right) \leq k_{uv} x_{uv} \qquad \forall uv \in E \tag{2.2c}$$

$$-y_u \leq \sum_{v \in N(u)} (g^q_{uv} - g^q_{vu}) \leq y_u \qquad \begin{matrix} \forall u \in V \\ \forall q \in Q \end{matrix} \tag{2.2d}$$

$$x_{uv} \in \mathbb{Z}_+, y_u \in \{0,1\}, f^q_{uv} \geq 0, g^q_{uv} \geq 0 \tag{2.2e}$$

Note that the MILP constraints are very similar to the ones given in Definition 2.1, with the difference that the f^q and g^q variables do not model absolute flow values but fractions of a flow of a certain commodity $q \in Q$. This way, the Objective Function (2.2a) *corresponds to the network cost as defined in* (2.1l). *Constraint* (2.2b) *describes flow conservation (either compressed or not, 100% of every commodity has to be routed) and model the requirements* (2.1i) *and* (2.1j). *The Constraint* (2.2c) *describes the link capacity constraints as in* (2.1h), *guaranteeing that the total flow on an edge (compressed flow is scaled down by γ^q) may not exceed the installed capacity. Finally, for any commodity $q \in Q$, the inequalities* (2.2d) *describe the relation between the uncompressed flow f and the compressed flow g as in* (2.1k). *Any interchange between the traffic value of f and the traffic value of g may only occur, if a compressor is active in node $u \in V$. We model this by a M ($M = 1$) constraint, i.e., if, for some node $u \in V$, the sum of incoming*

compressed flow is bigger than the sum of outgoing compressed flow (or vice versa), a compressor has to be active in u. Hence, if u does not allow for compression ($y_u = 0$), the node can only forward flow without compression or de-compression.

Clearly, the MILP formulation can be tuned to any of the variants as described in Definition 2.4 and in Definition 2.3 by simply fixing the corresponding parameters in Constraint (2.2c). We define the set of *feasible solutions* for NDPC as follows.

Definition 2.5 (Feasibility region of NDPC)**.** *Let*

$$R := \left\{(x, y, f, g) \in \mathbb{Z}_+^{|E|} \times \{0,1\}^{|V|} \times [0,1]^{2|E||Q|} \times [0,1]^{2|E||Q|} : (2.2b) - (2.2d)\right\} \quad (2.3)$$

be the set of NDPC ***feasible solutions****.*

2.3 Polyhedral results

2.3.1 Definitions and notation

In this section, we investigate the MILP formulation for NDPC given in Remark 2.4. We consider the version of NDPC as described by Definition 2.4, i.e., we assume that all capacities are constant with value one, and that for all commodities $q \in Q$ the compression ratios γ^q are constant.

In general, NDPC can be seen as a composition of two problems, the NDP component (installing edge capacities and compressors) and a routing component (determining a flow for each commodity). On the one hand, the routing part is computationally tractable: If x and y are fixed, the problem can be solved by linear programming. On the other hand, the network design part is a difficult problem (see Theorem 1.3). Hence, we focus on the projection of NDPC onto the integer space (given by the x, y variables). This projection is defined as follows:

Definition 2.6. *With respect to Remark 2.4, we denote the set of (integer) points $(x, y) \in \mathbb{Z}_+^{|R|} \times \{0,1\}^{|V|}$ for which f^q and g^q as in Definition 2.1 exist as*

$$P_I := \left\{(x, y) \in \mathbb{Z}_+^{|E|} \times \{0,1\}^{|V|} \mid \exists f, g \geq 0, \text{ satisfying (2.2b)-(2.2d)}\right\}, \quad (2.4a)$$

with its convex hull $P := conv(P_I)$.

Likewise, we refer to the integer points (with the same property) in the x-space as

$$P_{X_I} := \left\{x \in \mathbb{Z}_+^{|E|} \mid \exists y \in \{0,1\}, f, g \geq 0, \text{ satisfying (2.2b)-(2.2d)}\right\} \quad (2.4b)$$

and to its convex hull as $P_X := conv(P_{X_I})$. With respect to Remark 2.3, we denote the convex hull of NDP_0 and of NDP_γ (in the x-space) as P_{X_0} and P_{X_γ}.

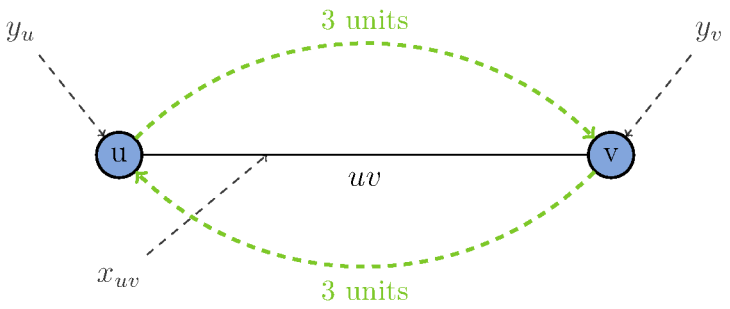


Figure 2.7: A two node NDPC instance.

This way, P_I denotes the fixed (partial) solutions (x^*, y^*), for which feasible assignments to f and g exist, such that a complete solution (x^*, y^*, f^*, g^*) can be obtained. With a slight abuse of notation, we refer to P as the convex hull of NDPC in the following. We begin with a short example:

Example 2.3. *Consider the following two node* NDPC *instance. Let* $V = \{u, v\}$ *and* $E = \{uv\}$ *with* $k_{uv} = 1$. *Further, let* $Q = \{q_1, q_2\}$ *with*

$$s^{q_1} = t^{q_2} = u, \quad s^{q_2} = t^{q_1} = v, \quad \text{and} \quad d^{q_1} = d^{q_2} = 3. \tag{2.5}$$

Finally, let $\gamma = 2$. *See Figure 2.7 for a visualization.*

If no compression takes place, $6 = (3 + 3)$ *units of flow have to be sent over the edge* uv, *i.e.,* $(x_{uv}, y_u, y_v) = (6, 0, 0)$ *is a feasible solution. Clearly,* $(6, 1, 0)$ *and* $(6, 0, 1)$ *are also feasible, even though the activated compressors are not used. If compressors are active in* u *and in* v, *both commodities can be compressed, so that* $(3, 1, 1)$ *is a feasible solution. Since any feasible solution stays feasible if more capacity is installed, we have*

$$P_I := \{t(1, 0, 0) + v \mid t \in \mathbb{Z}_+, v \in \{(3, 1, 1), (6, 1, 0), (6, 0, 1), (6, 0, 0)\}\} \tag{2.6}$$

We remark that P_I and thereby P is unbounded. Note that we assume any NDPC instance to be *non-degenerated*, i.e., we assume that the compression aspect induces a difference in comparison to the NDPC problem, e.g., as in the case that $k_{uv} = 1$ for all $uv \in E$ and that there is $q \in Q$ with $d^q = M$ (M sufficiently large) and $\gamma^q > 1$.

The relation between the x-space of the NDPC problem and its two corresponding NDP problems is as follows.

Corollary 2.2. *Consider a non-degenerated* NDPC *instance. It holds that*

$$(i) \quad P_{X_0} \subset P_X, \qquad (ii) \quad P_X = P_{X_\gamma}. \tag{2.7}$$

Proof. By definition, for $x \in P_{X_0}$, it is $x \in P_X$ and for $x \in P_{X_\gamma}$, it is $x \in P_X$.

(i) As the instance is non-degenerate, there is a solution (x, y, f, g) for NDPC with $y > 0$ for at least two components, such that $(x, 0, f, g)$ is no solution for NDP_0. This way, it is $x \in P_X$ and $x \notin P_{X_0}$.

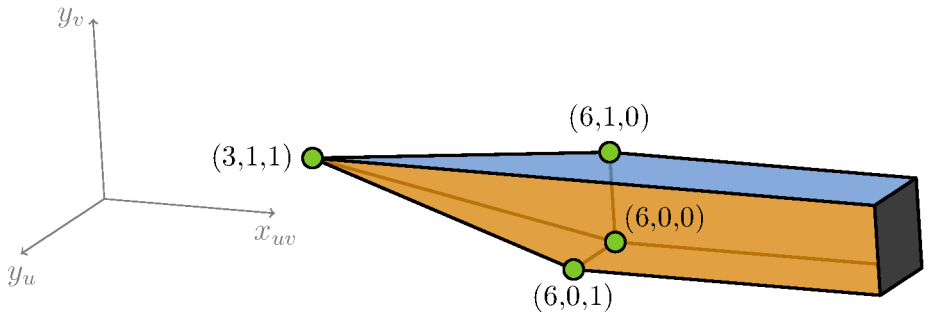


Figure 2.8: The convex hull of the two node NDPC problem from Example 2.3, vertices highlighted. The gray area indicates where the unbounded polyhedron is capped due to illustration purposes.

(ii) Given $x \in P_X$, there exists a solution (x, y, f, g) for the NDPC instance. Then, $(x, 1, f, g)$ is a feasible solution as well and hence, $x \in P_{X_\gamma}$.

□

2.3.2 Dimension and trivial facets

At first, we show that P is full dimensional.

Lemma 2.1 (Dimension). *If G is connected, P is full dimensional, i.e.,*

$$\dim(P) = |E| + |V|. \tag{2.8}$$

Proof. Since G is connected, there exists a spanning tree $T \subseteq E$. A feasible solution can be constructed by defining

$$x^0_{uv} = \begin{cases} M & \text{if } uv \in T \\ 0 & \text{else} \end{cases}, \quad y^0_u = 0 \quad \forall\, u \in V, \tag{2.9}$$

for $M \in \mathbb{Z}_{\geq 0}$ sufficiently large. Denote this solution by $s = (x^0, 0)$. For any unit vector $e_i \in \mathbb{R}^{|V|+|\bar{E}|}$, the vector $s + e_i$ yields a valid solution, since employing an additional edge or compressor is always feasible. There are $|E| + |V|$ potential vectors e_i, such that in total, there are $|E| + |V| + 1$ affinely independent vectors (the difference vectors with respect to s form the identity matrix), i.e., $\dim(P) = |E| + |V|$. □

The premise of connectivity is not very restrictive, since either the problem can be broken down into smaller sub-problems, or it is directly infeasible if there are commodities which are required to be sent across components. We extend the previous example:

Example 2.4. *Consider the two node* NDPC *problem as described in Example 2.3. Apparently, it is $\dim(P) = 3$. For a sketch of P, see Figure 2.8 where the polyhedron is capped by the gray face for illustration purposes.*

With a similar argument as for the dimension, we obtain the trivial facets of P.

Lemma 2.2. *Let G be connected.*

(i) Let $uv \in E$. If $G' := (V, E \setminus \{uv\})$ is connected, $x_{uv} \geq 0$ defines a facet of P.

(ii) For all $u \in V$, the inequality $0 \leq y_u$ and the inequality $y_u \leq 1$ defines a facet of P.

Proof. Since G, respectively G', is connected, we use the construction of Lemma 2.1 and consider a spanning tree T with $uv \notin T$ with the induced solution $(x^0, 0)$. In the notation of Lemma 2.1, there are only $|V| + |E| - 1$ vectors $e_i \in \mathbb{R}^{|V|+|E|}$, such that $s + e_i$ is feasible since one edge, respectively one compressor variable needs to be set to zero/one. Note that a solution is still feasible if one additional compressor is employed, even though its functionality is not used. These are $\dim(P) - 1$ affinely independent solutions fulfilling either $x_{uv} \geq 0$ or $0 \leq y_u$ with equality.

For $y_u \leq 1$, we observe that $\tilde{s} := (x^0, y^0)$, where $y^0_v = 0$ for all $v \in V, v \neq u$ and $y^0_u = 1$ is a feasible solution. In the notation of Lemma 2.1, there are again $|V| - 1 + |E|$ vectors e_i, such that $\tilde{s} + e_i$ is feasible. These are $\dim(P) - 1$ affinely independent solutions. □

The remaining case of uv being a cut-edge is discussed in Subsection 2.3.4, Lemma 2.6, as it requires more complex families of inequalities. We continue with the next subsection, where we describe some universal properties of valid inequalities.

2.3.3 Properties of facets and valid inequalities

We remark that many of the following results hold in a similar manner for the NDP problem. The corresponding results can be found in the works of Agarwal [1, 2], Magnanti et al. [91, 92], and Raack et al. [106]. At first, we show that for any valid inequality (facet) the coefficients of the x, respectively y, variables are always non negative. We start with the coefficients of the edge capacity variables x:

Lemma 2.3. *Let G be connected and let $\alpha \in \mathbb{R}^{|E|}$, $\beta \in \mathbb{R}^{|V|}$ and $\rho \in \mathbb{R}$. If $\alpha x + \beta y \geq \rho$ is valid for P, then $\alpha_{uv} \geq 0$ for all $uv \in E$.*

Proof. Assume the contrary, that is, let $\alpha_{uv} < 0$ for a certain, fixed coefficient uv. Since P is not empty, by Lemma 2.1, we know, that a feasible solution to NDPC exists. Clearly, this solution is still feasible, if additional capacity is installed on uv. For $x_{uv} \to \infty$, we have that $\alpha_{uv} x_{uv} \to -\infty$, such that each fixed ρ is undercut. So, $\alpha x + \beta y \geq \rho$ cuts off a feasible solution. This is a contradiction. □

For the compressor variables y, we present similar results for facet defining inequalities:

Lemma 2.4. *Let G be connected and let $\alpha \in \mathbb{R}^{|E|}_+$, $\beta \in \mathbb{R}^{|V|}$ and $\rho \in \mathbb{R}$. Assume that $\alpha x + \beta y \geq \rho$ defines a facet of P and that it is not equivalent to the inequality $y_u \leq 1$ for some $u \in V$. It holds that $\beta_u \geq 0$ for all $u \in V$.*

Proof. Assume the contrary, that is, let $\beta_u < 0$ for a certain, fixed coefficient u. Reformulating the inequality yields

$$\alpha x + \sum_{\substack{v \in V \\ v \neq u}} \beta_v y_v \geq \rho - \beta_u y_u. \tag{2.10}$$

By assumption, there exists a solution (x^*, y^*) satisfying the inequality with equality and featuring $y_u^* = 0$. Otherwise, all solutions satisfying the inequality with equality obey $y_u = 1$ and the inequality is equivalent to $y_u \leq 1$. This solution is still feasible, if we activate the compressor in u, i.e., if we set $y_u^* := 1$. Inserting into the reformulated Inequality (2.10), it is

$$\rho = \alpha x^* + \sum_{\substack{v \in V \\ v \neq u}} \beta_v y_v^* \geq \rho - \beta_u > \rho. \tag{2.11}$$

This is a contradiction. □

We can combine the preceding lemmata and obtain:

Corollary 2.3. *Let G be connected and let $\alpha \in \mathbb{R}_+^{|E|}$, $\beta \in \mathbb{R}_+^{|V|}$ and $\rho \in \mathbb{R}$. Assume that $\alpha x + \beta y \geq \rho$ defines a facet of P and let the inequality not be equivalent to any of the trivial facets $y_u \geq 0$, $y_u \leq 1$ or $x_{uv} \geq 0$. Then, it is $\rho > 0$.*

Proof. From Lemma 2.3 and Lemma 2.4 we know that the left hand side has to be larger or equal to zero. If $\rho = 0$, the inequality is dominated as it is a linear combination of the inequalities $y_u \geq 0$ and $x_{uv} \geq 0$. □

We conclude this subsection with a theorem, describing the relation between an instance of NDPC and the same instance where a single edge is contracted. In particular, we describe how a problem instance of NDPC can be aggregated into a smaller problem, so that valid inequalities for that smaller problem can be disaggregated in such a way, that the resulting disaggregated inequalities hold for the original problem.

Since we are interested in the feasibility region of NDPC, we neglect the objective function for the remainder of this subsection and define the 1-edge contracted problem:

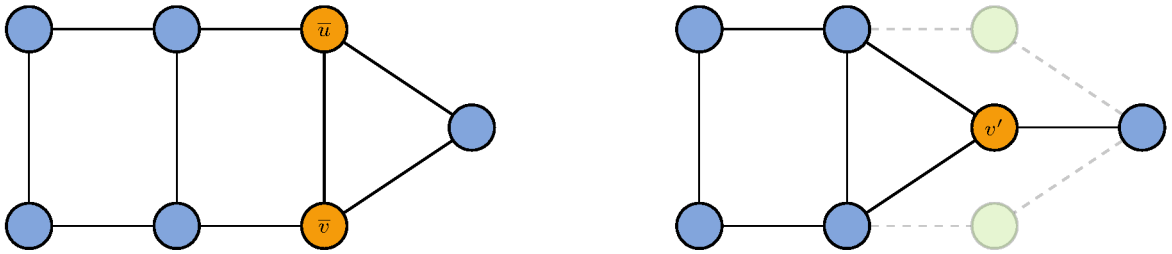
Definition 2.7 (1-edge contracted problem)**.** *Let a* NDPC *instance (G, Q, γ) be given. Let $\overline{uv} \in E$. For the **1-edge contracted problem**, the set of vertices is given by adding a dummy node u' and removing the nodes connected to the contracted edge $\overline{uv}$, i.e.,*

$$V' := V \cup \{u'\} \setminus \{\overline{u}, \overline{v}\}. \tag{2.12a}$$

Similarly, the set of edges of 1-edge contracted problem are adapted to the contraction by

$$\begin{aligned} E' := \quad & E \cup \{uu' \mid u \in V \setminus \{\bar{u}, \bar{v}\} : u\overline{u} \in E \vee u\overline{v} \in E\} \\ & \setminus (\{uv \in E \mid u = \overline{u} \vee u = \overline{v}\} \cup \{\overline{uv}\}). \end{aligned} \tag{2.12b}$$

Let the contracted graph G' be given as $G' := (V', E')$.



(a) "Original" graph $G = (V, E)$ with $\overline{uv} \in E$

(b) 1-edge contracted graph $G' = (V', E')$. The edge $\overline{uv}$ is contracted into the node u'

Figure 2.9: Depiction of the 1-edge contraction of a given graph G.

For any $u \in V'$, $u \neq u'$, let q_1 be the commodity with $s^{q_1} = u$ and $t^{q_1} = \overline{u}$, and let q_2 be the commodity with $s^{q_2} = u$ and $t^{q_2} = \overline{v}$. Introduce a new commodity: q_t^u with $d^{q_t^u} = d^{q_1} + d^{q_2}$, $s^{q_t^u} = u$, and with $t^{q_t^u} = u'$.

Analogously, let q_3 be the commodity with $s^{q_3} = \overline{u}$ and $t^{q_3} = u$, and let q_4 be the commodity with $s^{q_4} = \overline{v}$ and $t^{q_4} = u$. Add the new commodity: q_s^u with $d^{q_s^u} = d^{q_3} + d^{q_4}$, $s^{q_s^u} = u'$, and with $t^{q_s^u} = u$.

For the 1-edge contracted problem, we define the commodities Q' as

$$Q' := Q \setminus \{q \in Q | s^q \in \{\overline{u}, \overline{v}\} \vee t^q \in \{\overline{u}, \overline{v}\}\} \cup \{q_t^u, q_s^u \mid u \in V'\}\,. \tag{2.12c}$$

The corresponding instance (G', Q', γ) defines the 1-edge contracted problem NDPC$'$.

In Figure 2.9, a depiction of a graph G and, given an edge $\overline{uv} \in E$, its 1-edge contracted version G' is visualized. For later reference, we denote the set of nodes, neighboring both node $\overline{u} \in \overline{uv}$ and node $\overline{v} \in \overline{uv}$, as

$$S := \left\{u \in V \setminus \{\overline{u}, \overline{v}\} \middle| u\overline{u} \in E \wedge u\overline{v} \in E\right\}. \tag{2.13}$$

In the following, the convex hull of all feasible integer (x, y) solutions of NDPC$'$ is denoted as P'. Note that $|E'| = |E| - |S| - 1$.

Lemma 2.5. *Consider an instance of* NDPC. *Let the inequality $(\alpha)'x + (\beta)'y \geq \rho$ define a facet of P', which is obtained by contracting the edge $\overline{uv} \in E$.*

Define $\alpha \in \mathbb{R}^{|E|}$ and $\beta \in \mathbb{R}^{|V|}$ as follows: Let

$$\alpha_{uv} = (\alpha_{uv})' \qquad \forall uv \in E : \{u, v\} \cap \{\overline{u}, \overline{v}\} = \emptyset \tag{2.14a}$$
$$\alpha_{u\overline{u}} = (\alpha_{uu'})' \qquad \forall u\overline{u} \in E \tag{2.14b}$$
$$\alpha_{u\overline{v}} = (\alpha_{uu'})' \qquad \forall u\overline{v} \in E \tag{2.14c}$$
$$\alpha_{\overline{uv}} = 0 \tag{2.14d}$$

for the coefficients of the edge variables x, and let

$$\beta_u = (\beta_u)' \qquad \forall \overline{u} \neq u \neq \overline{v} \in V \tag{2.15a}$$
$$\beta_{\overline{u}} = \beta_{u'} \tag{2.15b}$$
$$\beta_{\overline{v}} = \beta_{u'} \tag{2.15c}$$

for the coefficients of the compressor variables y. Then, the inequality $\alpha x + \beta y \geq \rho$ defines a facet of P, if $\rho > 0$, i.e., if $(\alpha)'x + (\beta)'y \geq \rho$ is no trivial facet.

Proof. We structure the proof into three parts. In Part i), we show that $\alpha x+\beta y \geq \rho$ is valid for P. In Part ii) we construct $|V|+|E|$ feasible solutions, fulfilling the inequality with equality, and in Part iii), we show the affine independence of these solutions.

Part i) We show that $\alpha x + \beta y \geq \rho$ is valid for P. Assume the contrary, i.e., assume that there exists a solution $(\tilde{x}, \tilde{y}) \in P$ with $\rho > \alpha\tilde{x} + \beta\tilde{y}$. We employ the definition of the 1-edge contraction and use $(\tilde{x}, \tilde{y})$ to construct a feasible solution $(\tilde{x}', \tilde{y}')$ for NDPC$'$:

$$\tilde{x}'_{uv} = \tilde{x}_{uv}, \quad \forall uv \in E : \{u, v\} \cap \{\overline{u}, \overline{v}\} = \emptyset \tag{2.16a}$$

$$\tilde{x}'_{uu'} = \begin{cases} \tilde{x}_{u\overline{u}} + \tilde{x}_{u\overline{v}} & \text{if} \quad u \in S \\ \tilde{x}_{u\overline{u}} & \text{if} \quad u\overline{u} \in E \text{ and } u\overline{v} \notin E \\ \tilde{x}_{u\overline{v}} & \text{if} \quad u\overline{u} \notin E \text{ and } u\overline{v} \in E \end{cases} \quad \forall u \in V \tag{2.16b}$$

$$\tilde{y}'_{u'} = \max\{\tilde{y}_{\overline{u}}, \tilde{y}_{\overline{v}}\} \tag{2.16c}$$

$$\tilde{y}'_u = \tilde{y}_u \quad \forall u \in V \setminus \{\overline{u}, \overline{v}\}. \tag{2.16d}$$

Without loss of generality, we assume that for $u \in V$, we have $u\overline{u}, u\overline{v} \in E$. Hence, it is

$$\alpha_{u\overline{u}}\tilde{x}_{u\overline{u}} + \alpha_{u\overline{v}}\tilde{x}_{u\overline{v}} = (\alpha_{uu'})'\tilde{x}_{u\overline{u}} + (\alpha_{uu'})'\tilde{x}_{u\overline{v}} = (\alpha_{uu'})'\tilde{x}'_{uu'} \tag{2.17a}$$

$$\beta_{\overline{u}} = (\beta_{u'})' = \beta_{\overline{v}} \tag{2.17b}$$

$$\alpha_{\overline{u}\overline{v}}\tilde{x}_{\overline{u}\overline{v}} = 0. \tag{2.17c}$$

Thus, we obtain

$$\rho > \alpha\tilde{x} + \beta\tilde{y} = (\alpha)'\tilde{x}' + (\beta)'\tilde{y}' - \beta_{u'}\tilde{y}'_{u'} + \beta_{\overline{u}}\tilde{y}_{\overline{u}} + \beta_{\overline{v}}\tilde{y}_{\overline{v}} \geq \rho \tag{2.18}$$

where the last inequality holds because $\tilde{y}'_{u'} = \max\{\tilde{y}_{\overline{u}}, \tilde{y}_{\overline{v}}\} \leq \tilde{y}_{\overline{u}}+\tilde{y}_{\overline{v}}$ as well as (2.17a) and $\beta_{\overline{u}}, \beta_{\overline{v}} \geq 0$ for non trivial facets. By Lemma 2.4 and the requirement that $(\alpha)'x+(\beta)'y \geq \rho$ is no trivial facet, we know that $\beta_{\overline{u}}, \beta_{\overline{v}} \geq 0$. We obtain a contradiction which implies that $\alpha x + \beta y \geq \rho$ is valid for P.

Part ii) We show that $\alpha x + \beta y \geq \rho$ is facet defining for P, i.e., we construct $|V| + |E|$ solutions fulfilling the inequality with equality. Since $(\alpha)'x + (\beta)'y \geq \rho$ defines a facet of P', there are

$$p_0 := |E'| + |V'| = |E| - |S| - 1 + |V| - 1 = |E| - |S| + |V| - 2 \tag{2.19}$$

affinely independent integer solutions satisfying the inequality with equality. Denote these solutions by $s_1, \ldots s_{p_0}$. Note that for each $u \in S$, we can assume that there exists a solution with $x_{uu'} \neq 0$, since otherwise, $(\alpha)'x + (\beta)'y = x_{uu'} \geq 0$ is a trivial facet. From these solutions, we will construct $|E| + |V|$ affinely independent integer solutions, satisfying $\alpha x + \beta y = \rho$.

For this purpose, we expand the solutions s_j for $j = 1, \dots, p_0$ as follows. Every value of an edge-variable x_{uv} in P', $u \neq u' \neq v$ is directly copied to its corresponding variable in P. Every value of an edge variable connecting to u' ($x_{uu'}$) is copied to $x_{u\overline{u}}$ if $u\overline{u}$ exists or to $x_{u\overline{v}}$ if not (one of the edges always exists by construction). If both edges exist, we chose one of them to carry the value such that the other is equal to zero. $x_{\overline{uv}}(= M)$ is set to a sufficiently high value.

All compressor variables/values are copied to their corresponding variables and we set $y_{\overline{u}} = y_{u'}$ and $y_{\overline{v}} = 0$.

We show the feasibility of this solution: By definition of the shrinking process all demands can be sustained as there is enough additional capacity to route the commodities $q \in Q$ with $s^q, t^q \in \{\overline{u}, \overline{v}\}$ directly on $\overline{uv}$. To obtain a feasible solution, only a re-routing, potentially back and forth if flow needs to be decompressed in $\overline{u}$, between $\overline{u}$ and $\overline{v}$ is necessary. Since $x_{\overline{uv}}$ yields enough capacity, this is always feasible. This solution satisfies $\alpha x + \beta y = \rho$. In total, we obtain p_0 integer, affinely independent solutions.

Since either $x_{u\overline{u}}$ or $x_{u\overline{v}}$ is zero in the above solutions, $|S|$ many different solutions can be obtained by shifting the value of $x_{u\overline{u}}$ to $x_{u\overline{v}}$ and eventually re-routing on $x_{\overline{uv}}$ if necessary. We call the additional solutions $(s_{p_0+1}, \dots s_{p_0+|S|})$.

Two further solutions can be constructed as follows. Given an arbitrary solution of the ones constructed before, a new solution is obtained by adding one unit of capacity on $x_{\overline{uv}}$, i.e., $x_{\overline{uv}} = M + 1$ ($s_{p_0+|S|+1}$). Furthermore, since $(\alpha')x + (\beta')y \neq y_{u'} \geq 0$, a solution with $y_{u'} \neq 0$ exists. As described above, this solution can be adapted to hold for P. In the adapted solution, we set $y_{\overline{v}} = y_{u'}$ and $y_{\overline{u}} = 0$. This solution is feasible as, again, re-routing on $x_{\overline{uv}}$ is possible. Denote this solution by $s_{p_0+|S|+2}$.

Part iii) We proof the affine independence of these solutions, that is, we show that the unique solution of

$$\sum_{i=1}^{|V|+|E|} \gamma_i s_i = 0, \quad \text{with} \quad \sum_{i=1}^{|V|+|E|} \gamma_i = 0 \tag{2.20}$$

is $\gamma_i = 0$ for all $i = 1, \dots, |V| + |E|$. Since the solutions s_{p_0+i}, for $i = 1, \dots, |S|$, respectively $s_{p_0+|S|+2}$, have unique non zero entries on $x_{u\overline{v}}$ or $y_{\overline{v}}$ variables, their coefficients γ_i need to be zero. We focus on the row given by $x_{\overline{uv}}$. Inserting the solution values yields

$$M \sum_{i=1}^{|V|+|E|} \gamma_i + \gamma_{p_0+|S|+1} = 0. \tag{2.21}$$

Since $\sum_{i=1}^{|V|+|E|} \gamma_i = 0$ is required to hold, this implies $\gamma_{p_0+|S|+1} = 0$.

By assumption, the (remaining) vectors s_1 to s_{p_0} contain p_0 affinely independent subvectors. Hence, all γ_i need to be zero, such that the $|E| + |V|$ vectors s_i are affinely independent. Thus, $\alpha x + \beta y \geq \rho$ defines a facet of P.

For $\rho = 0$, $\alpha x + \beta y \geq \rho$ can be obtained as linear combination of the non-negativity

constraints and can therefore not be facet defining. If $(\alpha)'x + (\beta)'y \geq \rho$ resembles to $y_u \leq 1$, the aggregated inequality is either $y_u \leq 1$ (again) or invalid if $u = u'$ was contracted from $\overline{uv}$ as the de-aggregated inequality is $y_{\overline{u}} + y_{\overline{v}} \leq 1$. □

We can apply Lemma 2.5 inductively and obtain

Corollary 2.4 (k-partition)**.** *Let V be partitioned into $k \in \mathbb{Z}_+$ connected components $S_1, \ldots, S_k$. For $\rho > 0$, let*

$$(\alpha)'x + (\beta)'y \geq \rho \tag{2.22}$$

be facet defining for the problem obtained by iteratively employing Definition 2.7 until each component consists of one node only. In this case, the disaggregated inequality $\alpha x + \beta y \geq \rho$ is, according to Definition 2.7, facet defining as well.

The immediate strength of this result is that a problem can be contracted into smaller variants. For these smaller problems, structural information like e.g., facets can potentially be found easier and then, such results can directly be transfered back to the original problem.

2.3.4 Cutset and extended cutset inequalities

Cutset Inequalities play a crucial role for the NDP problem, which is testified by various works in the literature, see for instance the works of Agarwal [1, 2], Magnanti et al. [91, 92] and of Raack et al. [106]. Many of these results can be carried over or can be extended to hold for the NDPC problem, as well. As indicated by the name, they formulate necessary conditions on edge capacities to allow to route traffic from one side of a cut in the network to the other side.

To describe Cutset Inequalities for NDPC, we require some further notation:

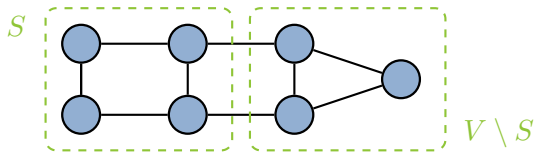
Definition 2.8. *Given two aubsets $S, \overline{S} \subset V$ with $|S| \geq 1$, $|\bar{S}| \geq 1$ and $S \cap \bar{S} = \emptyset$, the **total demand**, which needs to be routed **on the cut** (between S and $\overline{S}$), is denoted by*

$$Q^{S,\overline{S}} := \sum_{\substack{q \in Q \\ s^q \in S \\ t^q \in \overline{S}}} d^q + \sum_{\substack{q \in Q \\ t^q \in S \\ s^q \in \overline{S}}} d^q. \tag{2.23a}$$

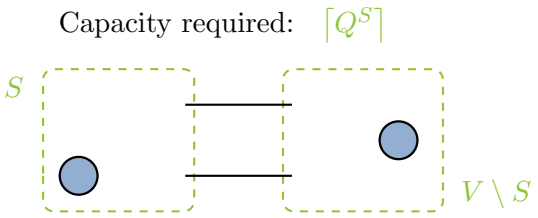
In case $\overline{S} = V \setminus S$, define $Q^S := Q^{S,\overline{S}}$. Let $\delta(S)$ denote all edges on the cut, that is, let

$$\delta(S) := \left\{ uv \in E \mid u \in S \wedge v \in \overline{S} \right\}. \tag{2.23b}$$

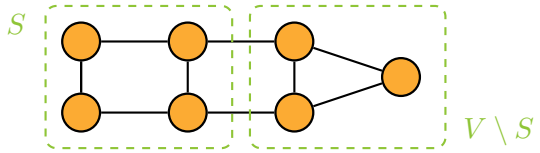
For the NDP problem, the Cutset Inequalities state that the capacity installed on the edges in $\delta(S)$ has to be larger or equal to the total amount of flow which must traverse these edges. Clearly, if $\delta(S) = \emptyset$, the problem is infeasible, except if $Q^S = 0$ in which case we decompose the problem into two smaller ones. Hence, we assume that $|\delta(S)| \geq 1$ holds for the remaining section.



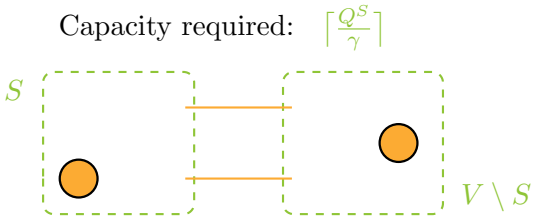
(a) NDP_0, no active compressors.



(b) The capacity requirement induced by S without any active compressors. No traffic (on the cut edges) can be compressed.



(c) NDP_γ, all compressors active.



(d) The capacity requirement induced by S and at least two active compressors. All traffic (on the cut edges) can be compressed

Figure 2.10: NDPC over a cut (green) induced by the node-set S. Blue nodes indicate inactive compressors, orange nodes imply active (de-) compression functionality.

Example 2.5. *Let an* NDPC *instance and a cut $\delta(S)$ be given. Consider the two corresponding NDP problems. For NDP_0, a visualization of such Cutset Inequality can be found in Figure 2.10 (a) and Figure 2.10 (b). Since no traffic is compressed, there have to be at least $\alpha_0 := \lceil Q^S \rceil$ units of capacity available on $\delta(S)$.*

For NDP_γ, a visualization of a Cutset Inequality can be found in Figure 2.10 (c) and Figure 2.10 (d). In this case, all flow can be compressed such that

$$\alpha_\gamma := \left\lceil \frac{Q^S}{\gamma} \right\rceil \tag{2.24}$$

units of capacity are required. In principle, two active compressors (one in S and one in $\overline{S}$) are already sufficient for a decreased capacity requirement on $\delta(S)$. In this case, all the flows between S and $\overline{S}$ can be routed through these two nodes so that they can be compressed. Of course, this comes at the cost of a comparably higher capacity requirement on the edges within S and $\overline{S}$.

All following results of this section depend on the values $\lceil Q^S \rceil$ and $\lceil \frac{Q^S}{\gamma} \rceil$, describing the amount of traffic which has to be sent across the cut, depending on active or inactive compression. If NDPC is not considered in the variant of Definition 2.4, the following remark explains how the results can be adapted to the more general variants.

Remark 2.5. *We consider the case of k_{uv} not being constant to one for all $uv \in E$. If k_{uv} is constant, i.e., $k_{uv} = k_0$ for all $uv \in E$, the two values have to be adapted by*

$$a_0 := \left\lceil \frac{Q^S}{k_0} \right\rceil \quad \text{and} \quad \alpha_\gamma := \left\lceil \frac{Q^S}{\gamma k_0} \right\rceil. \tag{2.25}$$

In the general case, k_{uv} has to appear as the coefficient of the x_{uv} variable (which, in the other cases, will be equal to one), and the two values remain unchanged. Note that this may require additional rounding when considering the Chvátal-Gomory procedure (we refer to Nemhauser and Wolsey [98] for a description of said procedure). Further, there is no guarantee that the resulting inequalities will be facet defining, see Section 1.3.2 for more details. As a consequence, the following inequalities are still valid but, in general, not facet defining any more.

If the compression ratio γ^q is not constant the value $\lceil \frac{Q^S}{\gamma} \rceil$ has to be replaced by

$$\left\lceil \sum_{\substack{q \in Q \\ s^q \in S \\ t^q \in \overline{S}}} \frac{d^q}{\gamma^q} + \sum_{\substack{q \in Q \\ t^q \in S \\ s^q \in \overline{S}}} \frac{d^q}{\gamma^q} \right\rceil, \tag{2.26}$$

and the value $\lceil Q^S \rceil$ for the uncompressed traffic remains unchanged.

We state the Cutset Inequalities for NDPC with constant compression ratio γ.

Lemma 2.6 (Cutset Inequalities)**.** *Let $S, \overline{S} \subseteq V$ be a partition of V. The inequality*

$$\sum_{uv \in \delta(S)} x_{uv} \geq \left\lceil \frac{Q^S}{\gamma} \right\rceil \tag{2.27}$$

is valid for NDPC*. We refer to this inequality as the* ***Cutset Inequality****.*

Proof. The inequality can be obtained by summing up the Capacity Inequality (2.2c) for all edges $uv \in \delta(S, \overline{S})$ on the cut. We obtain

$$\sum_{uv \in \delta(S)} x_{uv} \geq \sum_{uv \in \delta(S)} \sum_{q \in Q} d^q \Big(f^q_{uv} + f^q_{vu} + \frac{1}{\gamma} g^q_{uv} + \frac{1}{\gamma} g^q_{vu} \Big) \tag{2.28a}$$

$$\geq \sum_{q \in Q} \frac{d^q}{\gamma} \sum_{uv \in \delta(S)} \Big(f^q_{uv} + f^q_{vu} + g^q_{uv} + g^q_{vu} \Big) \tag{2.28b}$$

$$\geq \sum_{\substack{q \in Q \\ s^q \in S \\ t^q \in \overline{S}}} \frac{d^q}{\gamma} + \sum_{\substack{q \in Q \\ t^q \in S \\ s^q \in \overline{S}}} \frac{d^q}{\gamma} = \frac{Q^S}{\gamma} \tag{2.28c}$$

since Constraint (2.2b) imposes that, for any $q \in Q$, the sum of f^q and g^q is at least one across any cut (with s^q and t^q on different sides). Rounding up of the right hand side yields the inequality, showing that the Cutset Inequalities are rank one Chvátal-Gomory cuts (obtained by the Chvátal-Gomory procedure, see Nemhauser and Wolsey [98]). □

While we will prove later (see Lemma 2.8) that Cutset Inequalities are facet defining, they are blind to the existence of compressors. Though the coefficients of the y variables might be zero, the Cutset Inequality implicitly assumes that all compressors are active ($y_u = 1$). Hence, the flow volume on the right hand side is scaled down. In case that no compressors are active, the inequality can be tightened (see Figure 2.10) for a visualization) and we obtain the *Extended Cutset Inequalities*.

Lemma 2.7 (Extended Cutset Ineq.)**.** *Let $S, \overline{S} \subset V$ be a partition of V. The inequality*

$$\left(\left\lceil Q^S \right\rceil - \left\lceil \frac{Q^S}{\gamma} \right\rceil\right) \sum_{u \in S} y_u + \sum_{uv \in \delta(S)} x_{uv} \geq \left\lceil Q^S \right\rceil \tag{2.29}$$

is valid for NDPC*. We refer to this inequality as the* ***Extended Cutset Inequality****.*

Proof. Let S be given. As the sum of the y variables is integer, we consider two cases:
Case 1: Assume that $y_u = 0$ for all $u \in S$. Then, Constraint (2.29) equals

$$\sum_{uv \in \delta(S)} x_{uv} \geq \left\lceil Q^S \right\rceil, \tag{2.30}$$

which is equivalent to the Cutset Inequality in absence of compressors, i.e., to the case where no flows can be compressed.
Case 2: Assume that $\sum_{u \in S} y_u \geq 1$. We obtain

$$\left(\left\lceil Q^S \right\rceil - \left\lceil \frac{Q^S}{\gamma} \right\rceil\right) \sum_{u \in S} y_u + \sum_{uv \in \delta(S)} x_{uv} \tag{2.31a}$$

$$\geq \left(\left\lceil Q^S \right\rceil - \left\lceil \frac{Q^S}{\gamma} \right\rceil\right) + \sum_{uv \in \delta(S)} x_{uv} \geq \left\lceil Q^S \right\rceil \tag{2.31b}$$

which holds, because of the Cutset Inequality (2.27). □

We point out that the above proof also shows that the Extended Cutset Inequalities are dominated by the standard Cutset Inequalities if $\sum_{u \in S} y_u \geq 1$. We further remark that, for fixed S, one Cutset Inequality but two Extended Cutset Inequalities can be derived, associated to S and to $\bar{S}$. To prove that both inequalities, (2.27) and (2.29), are facet defining, we employ a lifting argument. Assume that an instance of the NDPC problem and a cut $\delta(S)$ is given. Let S and $\overline{S}$ be connected and consider the two corresponding problems NDP_0 and NDP_γ. For both problems, a Cutset Inequality can be defined (see below) and such inequality is known to be facet defining for the corresponding problem

(see Magnanti et al. [92]).

$$NDP_0: \quad y_u = 1, \quad \forall\, u \in V$$
$$\text{Cutset Inequality:} \quad \sum_{uv \in \delta(S)} x_{uv} \geq \left\lceil \frac{Q^S}{\gamma} \right\rceil \tag{2.32}$$
$$NDP_\gamma: \quad y_u = 0, \quad \forall\, u \in V$$
$$\text{Cutset Inequality:} \quad \sum_{uv \in \delta(S)} x_{uv} \geq \left\lceil Q^S \right\rceil \tag{2.33}$$

Therefore, both inequalities can be used as a starting point for lifting procedures which will give a facet defining inequality if all restrictions are lifted (compare Nemhauser and Wolsey [98]). This is carried out in the following two lemmata. Note that both lifting procedures may also serve as a proof of validity of the Cutset Inequality (2.27) and of the Extended Cutset Inequality (2.29).

Lemma 2.8 (Down-lifting Cutset Inequalities)**.** *The Cutset Inequality* (2.27) *defines a facet of* P*, if* S *and* $\overline{S}$ *are connected and* $\min\left\{|S|, |\overline{S}|\right\} > 1$.

Proof. Assume that both, S and $\overline{S}$, contain more than one node and are connected. We use Inequality (2.32) as starting point for the lifting procedure. By Magnanti et al. [92] the Cutset Inequality (2.27), respectively (2.32), defines a facet of

$$P^0 := P \cap \left\{ (x, y) \in \mathbb{Z}_+^{|E|} \times \{0,1\}^{|V|} \,:\, y_u = 1 \,\forall u \in V \right\}. \tag{2.34}$$

For the i^{th} lifting step, which is lifting a variable $y_{\overline{u}}$ for $\overline{u} \in V$, we define

$$L^i := L^{i-1} \cup \{y_{\overline{u}}\}, \quad L^0 := \emptyset \tag{2.35a}$$
$$P^i := P \cap \left\{ (x, y) \in \mathbb{Z}_+^{|E|} \times \{0,1\}^{|V|} \,:\, y_u = 1 \,\forall u \in V \setminus L^i \right\}. \tag{2.35b}$$

This way L^i contains the indices of already lifted/free variables y and P^i is the convex hull of all integer solutions of NDPC, intersected with the hyperplane where all non-lifted variables (except $y_{\overline{u}}$) are fixed to one.

During the lifting procedure, we always down-lift y_u for all $u \in V$. For the first lifting step $i = 1$, we start with an arbitrary vertex $\overline{u} \in V$. Lifting $y_{\overline{u}}$, we are looking for a

maximal $\beta_{\overline{u}}$, such that for all $(x, y) \in P^1$ the following holds:

$$\beta_{\overline{u}} y_{\overline{u}} + \sum_{uv \in \delta(S,\overline{S})} x_{uv} \geq \left\lceil \frac{Q^S}{\gamma} \right\rceil + \beta_{\overline{u}}. \tag{2.36a}$$

$$\Leftrightarrow \quad \beta_{\overline{u}}(y_{\overline{u}} - 1) \geq \left\lceil \frac{Q^S}{\gamma} \right\rceil - \sum_{uv \in \delta(S,\overline{S})} x_{uv}. \tag{2.36b}$$

$$\Leftrightarrow \quad \begin{cases} \beta_{\overline{u}} \leq \displaystyle\sum_{uv \in \delta(S,\overline{S})} x_{uv} - \left\lceil \frac{Q^S}{\gamma} \right\rceil, & \text{if } y_{\overline{u}} = 0 \\ 0 \geq \left\lceil \frac{Q^S}{\gamma} \right\rceil - \displaystyle\sum_{uv \in \delta(S,\overline{S})} x_{uv}, & \text{if } y_{\overline{u}} = 1. \end{cases} \tag{2.36c}$$

The latter inequality is always fulfilled, since the Inequality (2.32) was valid. The first inequality states, that $\beta_{\overline{u}}$ can be chosen as the optimal solution value of the following MILP. For later reference, the MILP is denoted for an arbitrary lifting step i for $i = 1, \ldots, |V|$:

$$\beta_{\overline{u}} := \min \quad \sum_{uv \in \delta(S,\overline{S})} x_{uv} - \left\lceil \frac{Q^S}{\gamma} \right\rceil \tag{2.37a}$$

$$\text{s.t.} \quad y_{\overline{u}} = 0 \tag{2.37b}$$

$$y_u = 1 \qquad \forall\, u \in V \setminus L^i \tag{2.37c}$$

$$y_u \in \{0, 1\} \qquad \forall\, u \neq \overline{u} \in L^i \tag{2.37d}$$

$$(x, y) \in P \tag{2.37e}$$

Since this is the first lifting step, and $\min\left\{|S|, |\overline{S}|\right\} > 1$, an active compressor is (still) available in both S and $\overline{S}$, and all flows between S and $\overline{S}$ can be compressed. Hence, the optimal solution with $y_{\overline{u}} = 0$, is given by $\sum_{uv \in \delta(S,\overline{S})} x_{uv} = \lceil \frac{Q^S}{\gamma} \rceil$ as the minimal capacity requirement on the cut. Consequently, it is $\beta_{\overline{u}} = 0$, such that the inequality does not change, but $y_{\overline{u}}$ is free.

The same behavior occurs in the following lifting steps as well. In each lifting step, exactly one $y_{\overline{u}}$ is set to zero. All other y_u are either fixed to one or free ($u \in L^{i-1}$). Inductively, each y_u with $u \in L^i$ has an objective coefficient of zero in the lifting problem such that the above MILP occurs in all lifting steps $i = 1, \ldots |V|$. This way, at least one compressor is always free to use in each of the two partitions, compression is always possible and the objective is only determined by the necessary edge capacity. So, at every lifting step, a solution $(x, y) \in P^i$ exists with $\sum_{uv \in \delta(S,\overline{S})} x_{uv} = \lceil \frac{Q^S}{\gamma} \rceil$ such that each lifting coefficient is zero.

Since the lifting process yields the Cutset Inequality (2.27), it defines a facet. □

We point out, that the lifting procedure is sequence independent. Additionally, under some (rather strict) restrictions, the above proof can be used to determine which facets from the NDP problem carry over to the NDPC problem as well.

Corollary 2.5. *Let $\alpha x \geq \rho$ be a facet of $P \cap \{(x,y) \in P \mid y_u = 1 \,\forall\, u \in V\}$. If for all $\overline{u} \in V$ the solution value of*

$$\begin{aligned} \min \quad & \alpha x - \rho && \text{(2.38a)} \\ \text{s.t} \quad & y_{\overline{u}} = 0 && \text{(2.38b)} \\ & (x,y) \in P \cap \{(x,y) \in P \mid y_u = 1 \,\forall\, \overline{u} \neq u \in V\} && \text{(2.38c)} \end{aligned}$$

is zero, $\alpha x \geq \rho$ yields a facet of P as well.

Proof. By Lemma 2.3, it is $\alpha \geq 0$. Consider the next lifting step. By the definition of NDPC, setting $y_u = 0$ for any $u \in V$ only increases the components of x. Without loss of generality, an optimal solution of the above program obeys $y_u = 1$ for all but the currently lifted variable, reproducing the same setting as in (2.38a)–(2.38c). By assumption, the lifting coefficient is zero. The claim follows inductively. □

Even though the assumptions of the corollary are rather strict, it can be applied to numerous classes of inequalities for NDP_γ. Consider in the following example:

Example 2.6. *Consider a* NDPC *instance and the corresponding NDP_γ instance. Let V be the disjoint union of S_1, S_2 and S_3, with $|S_i| > 1$ for $i = 1, \ldots, 3$ and let all S_i be connected. Similar to Definition 2.8, we denote the traffic which has to be sent between each of the sets S_i and S_j for $i \neq j$, with $i, j = 1, 2, 3$ as Q^{ij}, respectively $\frac{Q^{ij}}{\gamma}$ for the compressed traffic. By Magnanti et al. [91], the three partition inequality*

$$\sum_{\substack{u \in S_1 \\ v \in S_2}} x_{uv} + \sum_{\substack{u \in S_1 \\ v \in S_3}} x_{uv} + \sum_{\substack{u \in S_2 \\ v \in S_3}} x_{uv} \geq \left\lceil \frac{\left\lceil \frac{Q^{12}}{\gamma} \right\rceil + \left\lceil \frac{Q^{13}}{\gamma} \right\rceil + \left\lceil \frac{Q^{23}}{\gamma} \right\rceil}{2} \right\rceil \tag{2.39}$$

is facet defining for $\{(x,y) \in P \mid y_u = 1 \forall u \in V\}$ if $\lceil \frac{Q^{12}}{\gamma} \rceil + \lceil \frac{Q^{13}}{\gamma} \rceil + \lceil \frac{Q^{23}}{\gamma} \rceil$ is odd. Since $|S_i| > 1$, all traffic can still be compressed if one compressor is turned off such that we can inductively lift all y variables, each obtaining a lifting coefficient equal to zero. Thus, Constraint (2.39) *is valid and facet defining for the* NDPC *problem as well.*

Clearly, the same argument also holds for other families of inequalities e.g., for four partition based facets as described by Agarwal [1, 2].

We show that, the conditions in Lemma 2.8 are also necessary. At first, consider the case that S, respectively $\bar{S}$, is not connected.

Corollary 2.6. *If S or $\bar{S}$ is not connected, the Cutset Inequality is dominated.*

Proof. Without loss of generality, assume that $\bar{S}$ is not connected. The Cutset Inequality is dominated by the sum of the two Cutset Inequalities of the components: Let $\overline{S}$ be the disjoint union of $\overline{S_1}$ and $\overline{S_2}$. Summing the two induced Cutset Inequalities:

$$\sum_{uv \in \delta(S, \overline{S_1})} x_{uv} \geq \left\lceil \frac{Q^{\overline{S_1}, S \cup \overline{S_2}}}{\gamma} \right\rceil, \qquad \sum_{uv \in \delta(S, \overline{S_2})} x_{uv} \geq \left\lceil \frac{Q^{\overline{S_2}, S \cup \overline{S_1}}}{\gamma} \right\rceil \tag{2.40}$$

yields the inequality

$$\sum_{uv\in\delta(S,\overline{S})} x_{uv} \geq \left\lceil \frac{Q^{\overline{S_1},S\cup\overline{S_2}}}{\gamma} \right\rceil + \left\lceil \frac{Q^{\overline{S_2},S\cup\overline{S_1}}}{\gamma} \right\rceil$$
$$\geq \left\lceil \frac{Q^{\overline{S_1},S}}{\gamma} \right\rceil + \left\lceil \frac{Q^{\overline{S_2},S}}{\gamma} \right\rceil \geq \left\lceil \frac{Q^{\overline{S},S}}{\gamma} \right\rceil = \left\lceil \frac{Q^{S}}{\gamma} \right\rceil. \tag{2.41}$$

□

We consider the case that either of the two components consists of a single node.

Corollary 2.7. *The Cutset Inequality is dominated if*

$$\min\{|S|, |\overline{S}|\} = 1 \qquad \text{and} \qquad \left\lceil \frac{Q^S}{\gamma} \right\rceil \neq \lceil Q^S \rceil. \tag{2.42}$$

Proof. If $|S| = 1$ or $|\overline{S}| = 1$ and $\lceil \frac{Q^S}{\gamma} \rceil \neq \lceil Q^S \rceil$, the lifting procedure leads to the Extended Cutset Inequality as it is shown in Lemma 2.9. Without loss of generality, let $S = \{u\}$. The Cutset Inequality can be obtained by summing the Extended Cutset Inequality and $-(\lceil Q^S \rceil - \lceil \frac{Q^S}{\gamma} \rceil)$ times the trivial inequality $y_u \leq 1$. In particular, it is

$$\sum_{uv\in\delta(S)} x_{uv} \geq \lceil Q^S \rceil \tag{2.43a}$$
$$\Leftrightarrow \sum_{uv\in\delta(S)} x_{uv} + \left(\left\lceil Q^S \right\rceil - \left\lceil \frac{Q^S}{\gamma} \right\rceil \right) y_u \geq \left\lceil \frac{Q^S}{\gamma} \right\rceil \tag{2.43b}$$
$$+ \left[- \left(\left\lceil Q^S \right\rceil - \left\lceil \frac{Q^S}{\gamma} \right\rceil \right) y_u \geq - \left(\left\lceil Q^S \right\rceil - \left\lceil \frac{Q^S}{\gamma} \right\rceil \right) \right]. \tag{2.43c}$$

□

Since the MILP formulation of NDPC, respectively its LP relaxation, contains flow variables, the Cutset Inequality is not necessarily facet defining outside of the projection P. We discuss the strength of the Cutset Inequalities in the following remark:

Remark 2.6. *By Lemma 2.8, the Cutset Inequality describes a facet of P. Since the MILP formulation of* NDPC*, respectively its LP relaxation, contains flow variables, the same does not necessarily hold outside of the projection P. It is easy to give an example where, if $\frac{Q^S}{\gamma}$ is non-integer, the Cutset Inequality induces an improvement over the LP bound, e.g., consider a single edge network with a fractional commodity.*

On the contrary, if $\frac{Q^S}{\gamma}$ is integer, the lifting procedure did not start from a facet, such that it does not necessarily yield a facet. As shown in the proof of Lemma 2.6, in this case the Cutset Inequality is dominated by a linear combination of the capacity constraints.

We now show that the Extended Cutset Inequalities are facet defining for P:

Lemma 2.9 (Up-lifting Cutset Inequalities.)**.** *If S and $\overline{S}$ are connected, the Extended Cutset Inequality* (2.29) *defines a facet of P.*

Proof. Let S and $\overline{S}$ both be connected components. Using Inequality (2.33) as starting point for the lifting procedure Magnanti et al. [92] state that the Cutset Inequality (2.27), respectively (2.33), defines a facet of

$$P \cap \left\{ (x,y) \in \mathbb{Z}_+^{|E|} \times \{0,1\}^{|V|} \,:\, y_u = 0 \,\forall u \in V \right\}. \tag{2.44}$$

We employ the notation of Lemma 2.8. Up-Lifting the first variable $\overline{u} \in \overline{S}$ into the inequality, we are looking for a (minimal) $\beta_{\overline{u}}$, such that for all $(x,y) \in P^1$ it holds that:

$$\beta_{\overline{u}} y_{\overline{u}} + \sum_{uv \in \delta(S,\overline{S})} x_{uv} \geq \left\lceil Q^S \right\rceil. \tag{2.45}$$

The lifting problem writes

$$\begin{aligned}
\beta_{\overline{u}} := \max \quad & \left\lceil Q^S \right\rceil - \sum_{uv \in \delta(S,\overline{S})} x_{uv} && && \text{(2.46a)}\\
\text{s.t} \quad & y_{\overline{u}} = 1 && && \text{(2.46b)}\\
& y_u = 0 && \forall\, u \neq \overline{u} \in V \setminus L^i && \text{(2.46c)}\\
& y_u \in \{0,1\} && \forall\, u \in L && \text{(2.46d)}\\
& (x,y) \in P. && && \text{(2.46e)}
\end{aligned}$$

We obtain $\beta_{\overline{u}} = 0$, since without any compressors in S, we have

$$\sum_{uv \in \delta(S,\overline{S})} x_{uv} \geq \left\lceil Q^S \right\rceil. \tag{2.47}$$

The same holds for all following lifting steps when lifting a variable y_u, $u \in \overline{S}$ into the constraint. However, lifting y_u with $u \in S$ yields

$$\beta_u = \left(\left\lceil Q^S \right\rceil - \left\lceil \frac{Q^S}{\gamma} \right\rceil \right), \tag{2.48}$$

since there exist solutions with $y_u = y_{\overline{u}} = 1$. This way, minimizing the sum over all edge variables while exploiting connectivity in the components, we obtain

$$\sum_{uv \in \delta(S,\overline{S})} x_{uv} = \left\lceil \frac{Q^S}{\gamma} \right\rceil \tag{2.49}$$

in the best case. Again, the same also holds for all following lifting steps for $u \in S$. In each step, we have the compressors in $\overline{S}$ "for free" as well as the currently lifted one. To this end, the Extended Cutset Inequality (2.29) is obtained, i.e., it defines a facet. □

Note that the first variable y_u always gets a coefficient of zero. In the following lifting steps, independent of the order of lifting, all y_u on the same side of the cut as the first variable get a zero coefficient as well. All y_u on the other side get the coefficients of the extended cutset inequalities. This way, the lifting is almost sequence-independent. Only the first vertex determines whether S or $\overline{S}$ has non-zero coefficients in the Extended Cutset Inequality (2.29).

We remark that in the case $\lceil Q^S \rceil = \lceil \frac{Q^S}{\gamma} \rceil$, the Extended Cutset Inequality collapses to the Cutset Inequality. In this case, the Cutset Inequality is facet defining already under the connectivity restrictions. As for the Cutset Inequalities, the connectivity conditions are also necessary as is discussed in the following corollaries.

Corollary 2.8. *Let $\bar{S} = \bar{S}_1 \cup \bar{S}_2$ with $\bar{S}_1 \cap \bar{S}_2 = \emptyset$ and $\delta(\bar{S}_1, \bar{S}_2) = \emptyset$. The Extended Cutset Inequality*

$$\sum_{uv \in \delta(S)} x_{uv} + \left(\left\lceil Q^{S,\bar{S}_1} + Q^{S,\bar{S}_2} \right\rceil - \left\lceil \frac{Q^{S,\bar{S}_1} + Q^{S,\bar{S}_2}}{\gamma} \right\rceil \right) \sum_{u \in \bar{S}} y_u \geq \left\lceil Q^{S,\bar{S}_1} + Q^{S,\bar{S}_2} \right\rceil \tag{2.50}$$

is dominated if $\bar{S}$ is not connected.

Proof. Let $\overline{S}$ be the disjoint union of $\bar{S}_1$ and $\bar{S}_2$ with $\delta(\bar{S}_1, \bar{S}_2) = \emptyset$. Recall that in any case, the Extended Cutset Inequality is dominated if $\sum_{u \in \bar{S}} y_u \geq 1$. Summing two Extended Cutset Inequalities from the cuts $(S \cup \bar{S}_1, \bar{S}_2)$ and $(S \cup \bar{S}_2, \bar{S}_1)$, we obtain

$$\begin{aligned}
&\sum_{uv \in \delta(S,\bar{S}_1)} x_{uv} + \sum_{uv \in \delta(S,\bar{S}_2)} x_{uv} \\
&+ \left(\left\lceil Q^{S \cup \bar{S}_2, \bar{S}_1} \right\rceil - \left\lceil \frac{Q^{S \cup \bar{S}_2, \bar{S}_1}}{\gamma} \right\rceil \right) \sum_{u \in \bar{S}_1} y_u && (2.51) \\
&+ \left(\left\lceil Q^{S \cup \bar{S}_1, \bar{S}_2} \right\rceil - \left\lceil \frac{Q^{S \cup \bar{S}_1, \bar{S}_2}}{\gamma} \right\rceil \right) \sum_{u \in \bar{S}_2} y_u && (2.52) \\
&\geq \left\lceil Q^{S \cup \bar{S}_2, \bar{S}_1} \right\rceil + \left\lceil Q^{S \cup \bar{S}_1, \bar{S}_2} \right\rceil. && (2.53)
\end{aligned}$$

For $\alpha := \sum_{u \in \bar{S}} y_u$ with $0 \leq \alpha \leq 1$ it is

$$\begin{aligned}
&\left(\left\lceil Q^{S \cup \bar{S}_2, \bar{S}_1} \right\rceil - \left\lceil \frac{Q^{S \cup \bar{S}_2, \bar{S}_1}}{\gamma} \right\rceil \right) \sum_{u \in \bar{S}_1} y_u + \left(\left\lceil Q^{S \cup \bar{S}_1, \bar{S}_2} \right\rceil - \left\lceil \frac{Q^{S \cup \bar{S}_1, \bar{S}_2}}{\gamma} \right\rceil \right) \sum_{u \in \bar{S}_2} y_u \\
&\leq \left(\left\lceil Q^{S \cup \bar{S}_2, \bar{S}_1} \right\rceil - \left\lceil \frac{Q^{S \cup \bar{S}_2, \bar{S}_1}}{\gamma} \right\rceil \right) \alpha + \left(\left\lceil Q^{S \cup \bar{S}_1, \bar{S}_2} \right\rceil - \left\lceil \frac{Q^{S \cup \bar{S}_1, \bar{S}_2}}{\gamma} \right\rceil \right) \alpha. && (2.54)
\end{aligned}$$

Hence, after subtraction of Summand (2.51) and of Summand (2.52), the left hand side

of Constraint (2.53) has to be larger or equal to

$$\begin{aligned}
&(1-\alpha)\left\lceil Q^{S\cup\bar{S}_2,\bar{S}_1}\right\rceil + (1-\alpha)\left\lceil Q^{S\cup\bar{S}_1,\bar{S}_2}\right\rceil \\
&+\alpha\left\lceil\frac{Q^{S\cup\bar{S}_1,\bar{S}_2}}{\gamma}\right\rceil + \alpha\left\lceil\frac{Q^{S\cup\bar{S}_2,\bar{S}_1}}{\gamma}\right\rceil \\
&\geq (1-\alpha)\left\lceil Q^{S,\bar{S}_1}+Q^{S,\bar{S}_2}\right\rceil + \alpha\left\lceil\frac{Q^{S,\bar{S}_1}+Q^{S,\bar{S}_2}}{\gamma}\right\rceil. \qquad (2.55)
\end{aligned}$$

I.e., the Extended Cutset Inequality (2.50) is dominated. □

The proof that the other Extended Cutset Inequality with S not connected is also dominated will be given in the next subsection, in Corollary 2.10.

As for the standard Cutset Inequality, we discuss the strength of the Extended Cutset Inequality with respect to the MILP Formulation (2.2b)–(2.2e). Since, in this formulation, respectively in its LP relaxation, the flow variables are present, the inequality is not necessarily facet defining outside of the projection P. At first, we give an example where the Extended Cutset Inequalities strengthen the formulation.

Remark 2.7. *As is the case for the standard Cutset Inequality, it is easy to give an example where, if Q^S is fractional, the Extended Cutset Inequality tightens the MILP formulation. In the other case, if $Q^S \in \mathbb{Z}_+$, the formulation can still be improved if $\frac{Q^S}{\gamma}$ is fractional. For an example, consider the two-node* NDPC *problem (see Figure 2.7), where 8 units of traffic have to be sent between the nodes, where $\gamma = 3$, and where the capacity is constant to one. Writing (x, y_1, y_2) for any feasible point of the LP relaxation, $(\frac{8}{3}, 1, 1)$ is feasible for the relaxation. As soon as any of the two Extended Cutset Inequalities is added, said point is cut away but $(3, \frac{15}{16}, 1)$, respectively $(3, 1, \frac{15}{16})$, is still valid. In this solution, 3 units of capacity are installed such that 7.5 units of traffic can be sent compressed and 0.5 units are sent uncompressed. In this solution, it is not required to "fully" use one of the compressors such that y_1 is fractional. If both Extended Cutset Inequalities are added, these points are also cut away.*

We point out that in this case, even though the inequality was obtained by lifting from a dominated inequality, it still strengthens the formulation.

However, if $Q^S \in \mathbb{Z}_+$ and $\frac{Q^S}{\gamma} \in \mathbb{Z}_+$, the Extended Cutset Inequality is dominated as is shown in the next corollary.

Corollary 2.9. *If $Q^S \in \mathbb{Z}_+$ and $\frac{Q^S}{\gamma} \in \mathbb{Z}_+$, the Extended Cutset Inequality* (2.29) *is implied in the linear relaxation of Formulation* (2.2b)–(2.2e).

Proof. Let, a cut $(S, \bar{S})$ be given. At first, consider the Extended Cutset Inequality

$$\sum_{uv\in\delta(S)} x_{uv} + \left(\left\lceil Q^S\right\rceil - \left\lceil\frac{Q^S}{\gamma}\right\rceil\right)\sum_{u\in S} y_u \geq \left\lceil Q^S\right\rceil. \qquad (2.56)$$

Denote by $\bar{Q}$ the set of commodities $q \in Q$ for which $s^q \in S$ and $t^q \in \bar{S}$. The construction for the commodities $q \in Q$ with $s^q \in \bar{S}$ and $t^q \in S$ is analogue, such that, without loss of generality, we neglect these. We consider a fixed, valid solution (x, y, f, g) for the linear relaxation of the Formulation (2.2b)–(2.2e). With respect to this solution, let

$$\bar{g}^q := \max \left\{ 0, \sum_{\substack{uv \in \delta(S) \\ u \in S}} (g^q_{uv} - g^q_{vu}) \right\} \qquad \forall\, q \in \bar{Q}. \tag{2.57}$$

From the Flow Conservation Constraint (2.2b), we derive the following bounds:

$$\sum_{\substack{uv \in \delta(S) \\ u \in S}} f^q_{uv} \geq 1 \qquad \forall q \in \bar{Q} : \bar{g}^q = 0, \tag{2.58a}$$

$$\sum_{\substack{uv \in \delta(S) \\ u \in S}} f^q_{uv} \geq 1 - \bar{g}^q \qquad \forall q \in \bar{Q} : 0 < \bar{g}^q < 1, \tag{2.58b}$$

$$\sum_{\substack{uv \in \delta(S) \\ u \in S}} g^q_{uv} \geq \bar{g}^q \qquad \forall q \in \bar{Q} : 0 < \bar{g}^q < 1, \tag{2.58c}$$

$$\sum_{\substack{uv \in \delta(S) \\ u \in S}} g^q_{uv} \geq 1 \qquad \forall q \in \bar{Q} : \bar{g}^q \geq 1. \tag{2.58d}$$

Now, consider the sum of i) the capacity constraints on all edges of the cut and ii) for each commodity $q \in \bar{Q}$ and for all $u \in S$ the $\sum_{q \in \bar{Q}} (d^q - \frac{d^q}{\gamma})$ times the constraint $y_u \geq \sum_{v \in N(u)} (g^q_{uv} - g^q_{vu})$. For the afore mentioned solution, it is

$$\begin{aligned} &\sum_{uv \in \delta(S)} x_{uv} + \left(\sum_{q \in \bar{Q}} \left(d^q - \frac{d^q}{\gamma} \right) \right) \sum_{u \in S} y_u \\ \geq &\sum_{q \in Q} \sum_{uv \in \delta(S)} d^q \left(f^q_{uv} + f^q_{vu} + \frac{1}{\gamma} g^q_{uv} + \frac{1}{\gamma} g^q_{vu} \right) \\ &+ \left(\sum_{q \in \bar{Q}} \left(d^q - \frac{d^q}{\gamma} \right) \right) \sum_{\substack{uv \in \delta(S) \\ u \in S}} (g^q_{uv} - g^q_{vu}) \,. \end{aligned} \tag{2.59}$$

Omitting some of the positive terms, we obtain

$$\begin{aligned} \geq &\sum_{\substack{q \in \bar{Q} \\ \bar{g}^q = 0}} \sum_{\substack{uv \in \delta(S) \\ u \in S}} d^q f^q_{uv} + \sum_{\substack{q \in \bar{Q} \\ 0 < g^q < 1}} \sum_{\substack{uv \in \delta(S) \\ u \in S}} d^q \left(f^q_{uv} + \frac{1}{\gamma} g^q_{uv} \right) + \sum_{\substack{q \in \bar{Q} \\ \bar{g}^q \geq 1}} \sum_{\substack{uv \in \delta(S) \\ u \in S}} d^q \frac{1}{\gamma} g^q_{uv} \\ &+ \left(\sum_{q \in \bar{Q}} \left(d^q - \frac{d^q}{\gamma} \right) \right) \sum_{\substack{uv \in \delta(S) \\ u \in S}} (g^q_{uv} - g^q_{vu}) \,. \end{aligned} \tag{2.60}$$

Employing the Bounds (2.58a) – (2.58d), it is

$$\begin{aligned}
&\geq \sum_{\substack{q\in\bar{Q}\\ \bar{g}^q=0}} d^q + \sum_{\substack{q\in\bar{Q}\\ 0<\bar{g}^q<1}} \left(d^q(1-\bar{g}^q) + \frac{d^q}{\gamma}\bar{g}^q\right) + \sum_{\substack{q\in\bar{Q}\\ \bar{g}^q\geq 1}} \frac{d^q}{\gamma} \\
&\quad + \left(\sum_{\substack{q\in\bar{Q}\\ \bar{g}^q=0}} \left(d^q - \frac{d^q}{\gamma}\right)\right)\bar{g}^q + \left(\sum_{\substack{q\in\bar{Q}\\ \bar{0}<g^q<1}} \left(d^q - \frac{d^q}{\gamma}\right)\right)\bar{g}^q + \left(\sum_{\substack{q\in\bar{Q}\\ \bar{g}^q\geq 1}} \left(d^q - \frac{d^q}{\gamma}\right)\right) \qquad (2.61)\\
&= \sum_{q\in\bar{Q}} d^q = Q^S. \qquad (2.62)
\end{aligned}$$

All in all, we have

$$\sum_{uv\in\delta(S)} x_{uv} + \left(Q^S + \frac{Q^S}{\gamma}\right)\sum_{u\in S} y_u \geq Q^S. \qquad (2.63)$$

Since $Q^S \in \mathbb{Z}_+$ and $\frac{Q^S}{\gamma} \in \mathbb{Z}_+$, this is equivalent to the Extended Cutset Inequality (2.56) which is hence implied for the fixed (x, y, f, g). Since the solution was arbitrary, the Extend Cutset Inequality is, in general, implied.

For the other Extended Cutset Inequality, the proof is analogue. We conclude by pointing out that in the here discussed case, the Extended Cutset Inequality is of Chvátal-Gomory rank one, as is the standard Cutset Inequality. □

We conclude the subsection with a remark inspired by Magnanti et al. [92].

Remark 2.8. *Consider the two-node* NDPC *problem as given in Example 2.3 but with general demand data and general objective costs. Let* $D := d^{q_1} + d^{q_2}$, $\lceil D\rceil \neq \lceil \frac{D}{\gamma}\rceil$ *and let* D *and* $\frac{D}{\gamma}$ *be non-integral. For this problem, we can give a complete description of* P*. That is, we can solve the problem with the following LP.*

$$\begin{aligned}
\min \quad & c_{uv}x_{uv} + c_u y_u + c_v y_v && (2.64a)\\
\text{s.t.} \quad & x_{uv} + \left(\left\lceil D\right\rceil - \left\lceil \frac{D}{\gamma}\right\rceil\right) y_u \geq \left\lceil D\right\rceil && (2.64b)\\
& x_{uv} + \left(\left\lceil D\right\rceil - \left\lceil \frac{D}{\gamma}\right\rceil\right) y_v \geq \left\lceil D\right\rceil && (2.64c)\\
& y_u \leq 1 && (2.64d)\\
& y_v \leq 1 && (2.64e)\\
& y_u, y_v \geq 0, x_{uv} \in \mathbb{R} && (2.64f)
\end{aligned}$$

PROOF. The dual LP writes as

$$\max \quad \left\lceil D \right\rceil \Pi_1 + \left\lceil D \right\rceil \Pi_2 - \Phi_1 - \Phi_2 \tag{2.65a}$$

$$\text{s.t.} \quad \Pi_1 + \Pi_2 = c_{uv} \tag{2.65b}$$

$$\left(\left\lceil D \right\rceil - \left\lceil \frac{D}{\gamma} \right\rceil \right) \Pi_1 - \Phi_1 \leq c_u \tag{2.65c}$$

$$\left(\left\lceil D \right\rceil - \left\lceil \frac{D}{\gamma} \right\rceil \right) \Pi_2 - \Phi_2 \leq c_v \tag{2.65d}$$

$$\Pi_1, \Pi_2 \geq 0, \Phi_1, \Phi_2 \geq 0. \tag{2.65e}$$

We construct an (integer) primal solution and show that a solution with equal objective exists for the dual problem. Consider the following two cases:

Case I: Assume

$$\left\lceil D \right\rceil c_{uv} \geq \left\lceil \frac{D}{\gamma} \right\rceil c_{uv} + c_u + c_v \Leftrightarrow \frac{\left(\left\lceil D \right\rceil - \left\lceil \frac{D}{\gamma} \right\rceil \right)}{c_u + c_v} c_{uv} \geq 1. \tag{2.66}$$

Then, $x_{uv} = \lceil \frac{D}{\gamma} \rceil$ and $y_u = y_v = 1$ is optimal for the primal problem with objective value $\lceil \frac{D}{\gamma} \rceil c_{uv} + c_u + c_v$. Let $\Pi_1 := \frac{c_u}{c_u + c_v} c_{uv}$ and $\Pi_2 := \frac{c_v}{c_u + c_v} c_{uv}$ and let

$$\Phi_1 := \left(\left\lceil D \right\rceil - \left\lceil \frac{D}{\gamma} \right\rceil \right) \frac{c_u}{c_u + c_v} c_{uv} - c_u \tag{2.67a}$$

$$\Phi_2 := \left(\left\lceil D \right\rceil - \left\lceil \frac{D}{\gamma} \right\rceil \right) \frac{c_v}{c_u + c_v} c_{uv} - c_v. \tag{2.67b}$$

By construction, it is $\Pi_1 + \Pi_2 = c_{uv}$ and by assumption it is $\Phi_1, \Phi_2 \geq 0$. Therefore, (Π, Φ) is feasible for the dual. For the objective, it holds that

$$\begin{aligned} &\left\lceil D \right\rceil c_{uv} - \left(\left\lceil D \right\rceil - \left\lceil \frac{D}{\gamma} \right\rceil \right) \frac{c_u}{c_u + c_v} c_{uv} + c_u - \left(\left\lceil D \right\rceil - \left\lceil \frac{D}{\gamma} \right\rceil \right) \frac{c_v}{c_u + c_v} c_{uv} + c_v \\ &= \left(\left\lceil D \right\rceil - \left\lceil \frac{D}{\gamma} \right\rceil \right) + c_u + c_v. \end{aligned} \tag{2.68}$$

Case II: Assume

$$\left\lceil D \right\rceil c_{uv} < \left\lceil \frac{D}{\gamma} \right\rceil c_{uv} + c_u + c_v \Leftrightarrow \frac{\left(\left\lceil D \right\rceil - \left\lceil \frac{D}{\gamma} \right\rceil \right)}{c_u + c_v} c_{uv} < 1. \tag{2.69}$$

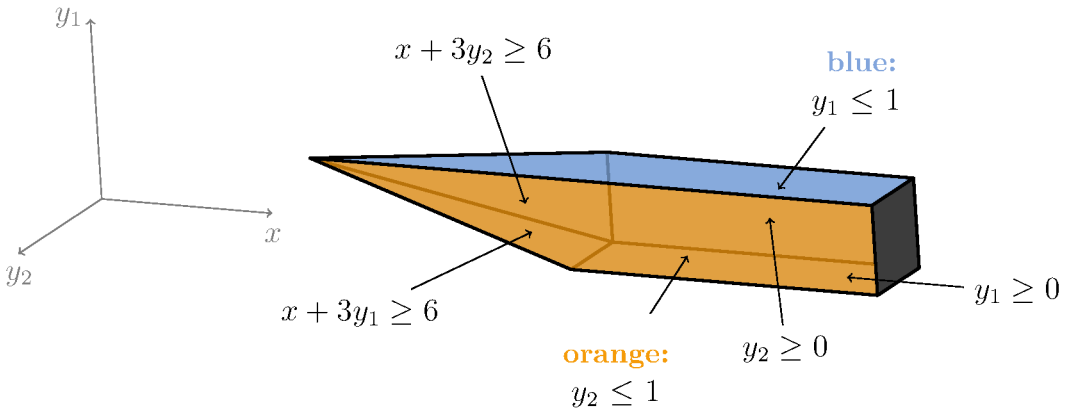


Figure 2.11: Convex hull of the two node NDPC problem from Example 2.3, inequalities highlighted. The gray area indicates where the unbounded polyhedron is capped due to illustration purposes.

Then, $x_{uv} = \lceil D \rceil$ and $y_u = y_v = 0$ is optimal for the primal problem with objective value $\lceil D \rceil c_{uv}$. Again, let $\Pi_1 := \frac{c_u}{c_u + c_v} c_{uv}$ and $\Pi_2 := \frac{c_v}{c_u + c_v} c_{uv}$. By assumption, it is

$$\left(\left\lceil D \right\rceil - \left\lceil \frac{D}{\gamma} \right\rceil \right) \frac{c_u}{c_u + c_v} c_{uv} \leq c_u \tag{2.70a}$$

$$\left(\left\lceil D \right\rceil - \left\lceil \frac{D}{\gamma} \right\rceil \right) \frac{c_v}{c_u + c_v} c_{uv} \leq c_v. \tag{2.70b}$$

This way, the solution (Π, Φ) with $\Phi_1 = \Phi_2 = 0$ is feasible for the dual problem and has an objective value of $\lceil D \rceil c_{uv}$. □

We point out that the convex hull of the two node example does not contain any Cutset Inequalities. The Cutset Inequality is $x_{uv} \geq \lceil \frac{D}{\gamma} \rceil$ ($x_{uv} \geq 3$) but, in this case, it is dominated, see Corollary 2.7. Instead, the polyhedron consists of the trivial inequalities and the two Extended Cutset Inequalities. Clearly, the inequalities coincide with the Polyhedron shown in Example 2.4, see Figure 2.11. We conclude with a remark:

Remark 2.9. *With respect to Lemma 2.5, we remark that the Extended Cutset Inequalities can be seen as disaggregated inequalities induced by a corresponding two node problem, whereas this is not the case for the standard Cutset Inequalities.*

2.3.5 Three node path instances

As we have observed in the previous subsection, (Extended-) Cutset Inequalities, respectively the two-node problems, play an important role for understanding the projection P of the NDPC formulation. In this context, the connectivity requirements of the sides of the considered cuts are necessary to obtain strong inequalities. In this subsection, we exemplary discuss three-node path networks as one basic case where we can find a cut

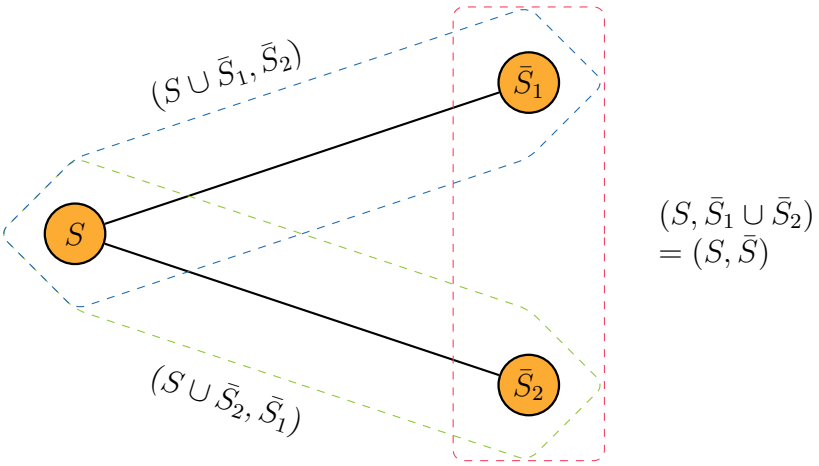


Figure 2.12: A three-node path instances, respectively a three partition of a graph where two of the elements are not connected. The "connected" cuts are indicated in green and blue, the cut with a non-connected side is marked in red.

whose side is not connected. Note that due to Lemma 2.5, we can equivalently discuss a three partition $(S, \bar{S}_1, \bar{S}_2)$ of the graph of an NDPC instance $G = (V, E)$, where

$$V = S \cup \bar{S}_1 \cup \bar{S}_2, \tag{2.71a}$$
$$S \cap \bar{S}_1 = S \cap \bar{S}_2 = \bar{S}_1 \cap \bar{S}_2 = \emptyset, \tag{2.71b}$$
$$|\delta(S \cup \bar{S}_1, \bar{S}_2)| > 0, \quad |\delta(S \cup \bar{S}_2, \bar{S}_1)| > 0, \quad |\delta(\bar{S}_2, \bar{S}_1)| = 0, \tag{2.71c}$$
$$\text{and } S, \bar{S}_1, \bar{S}_2 \text{ are connected.} \tag{2.71d}$$

For convenience sake, we write $\bar{S} := \bar{S}_1 \cup \bar{S}_2$. A sketch of such instance is given in Figure 2.12. In this setting, there are three different cuts in the network, each inducing two Extended- and one (standard) Cutset Inequality, such that, in total, nine valid inequalities can be derived for the projection P. We enlist these inequalities, their type, and their strength in Table 2.1. We derive additional classes of valid inequalities, especially for the case where one side of a cut (here $\bar{S}$) is not connected:

Lemma 2.10. *Consider an* NDPC *instance for which a 3-partition* $(S, \bar{S}_1, \bar{S}_2)$ *of* V *with the properties* (2.71a)–(2.71d) *exists. The following three inequalities hold for* NDPC*:*

$$\sum_{uv \in \delta(S, \bar{S}_1)} x_{uv} + \sum_{uv \in \delta(S, \bar{S}_2)} x_{uv} + \Bigg(\underbrace{\left\lceil Q^{S,\bar{S}_1} + \frac{Q^{\bar{S}_2,\bar{S}_1}}{\gamma} + Q^{S,\bar{S}_2} + \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma} \right\rceil}_{:=z_0} - \left\lceil \frac{Q^{S,\bar{S}_1} + Q^{\bar{S}_2,\bar{S}_1}}{\gamma} \right\rceil - \left\lceil \frac{Q^{S,\bar{S}_2} + Q^{\bar{S}_1,\bar{S}_2}}{\gamma} \right\rceil \Bigg) \sum_{y \in S} y_u$$
$$\geq \underbrace{\left\lceil Q^{S,\bar{S}_1} + \frac{Q^{\bar{S}_2,\bar{S}_1}}{\gamma} + Q^{S,\bar{S}_2} + \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma} \right\rceil}_{:=z_0}. \tag{2.72a}$$

(2.72b)

Table 2.1: A list of all nine (Extended-) Cutset Inequalities for a three-node path NDPC instance with respect to the projection P. The three inequalities where one side of the cut is not connected are dominated, the others are facet defining. If one side of the cut consists of a single node, the standard Cutset Inequality is dominated (denoted by *).

	Cut	Type	y-Variables	Strength
1	$(S \cup \bar{S}_1, \bar{S}_2)$	Std.	–	Facet*
2	$(S \cup \bar{S}_1, \bar{S}_2)$	Ext.	$S \cup \bar{S}_1$	Facet
3	$(S \cup \bar{S}_1, \bar{S}_2)$	Ext.	$\bar{S}_2$	Facet
4	$(S \cup \bar{S}_2, \bar{S}_1)$	Std.	–	Facet*
5	$(S \cup \bar{S}_2, \bar{S}_1)$	Ext.	$S \cup \bar{S}_2$	Facet
6	$(S \cup \bar{S}_2, \bar{S}_1)$	Ext.	$\bar{S}_1$	Facet
7	$(S, \bar{S}_1 \cup \bar{S}_2) = (S, \bar{S})$	Std.	–	Dominated
8	$(S, \bar{S}_1 \cup \bar{S}_2) = (S, \bar{S})$	Ext.	S	Dominated
9	$(S, \bar{S}_1 \cup \bar{S}_2) = (S, \bar{S})$	Ext.	$\bar{S}_1 \cup \bar{S}_2 = \bar{S}$	Dominated

$$
\begin{aligned}
&\sum_{uv \in \delta(S,\bar{S}_1)} x_{uv} + \sum_{uv \in \delta(S,\bar{S}_2)} x_{uv} \\
&+ \left(\left\lceil Q^{S,\bar{S}_1} + Q^{\bar{S}_2,\bar{S}_1} \right\rceil + \left\lceil Q^{S,\bar{S}_2} + Q^{\bar{S}_1,\bar{S}_2} \right\rceil - \underbrace{\left\lceil Q^{S,\bar{S}_1} + \frac{Q^{\bar{S}_2,\bar{S}_1}}{\gamma} + Q^{S,\bar{S}_2} + \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma} \right\rceil}_{:=z_0} \right) \sum_{y \in \bar{S}_i} y_u \\
&+ \left(\left\lceil Q^{S,\bar{S}_1} + Q^{\bar{S}_2,\bar{S}_1} \right\rceil + \left\lceil Q^{S,\bar{S}_2} + Q^{\bar{S}_1,\bar{S}_2} \right\rceil - \left\lceil \frac{Q^{S,\bar{S}_1} + Q^{\bar{S}_2,\bar{S}_1}}{\gamma} \right\rceil - \left\lceil \frac{Q^{S,\bar{S}_2} + Q^{\bar{S}_1,\bar{S}_2}}{\gamma} \right\rceil \right) \sum_{y \in S} y_u \\
&\geq \left\lceil Q^{S,\bar{S}_1} + Q^{\bar{S}_2,\bar{S}_1} \right\rceil + \left\lceil Q^{S,\bar{S}_2} + Q^{\bar{S}_1,\bar{S}_2} \right\rceil \qquad \forall\, i = 1, 2. \qquad (2.72c)
\end{aligned}
$$

Proof.

Inequality (2.72a)**:** If $\sum_{y \in S} y_u = 0$, the aggregated commodities $Q^{S,\bar{S}_1}$ and $Q^{S,\bar{S}_1}$ have to traverse at least one edge uncompressed. $Q^{\bar{S}_1,\bar{S}_2}$ has to traverse at least two edges but can potentially be compressed. The right hand side of the inequality states the sum of these traffic values and the left hand side describes a lower bound on the overall capacity in the network. In the other case, if $\sum_{y \in S} y_u \geq 1$, the inequality is dominated by the sum of the Cutset Inequalities induced by $(S \cup \bar{S}_1, \bar{S}_2)$ and $(S \cup \bar{S}_2, \bar{S}_1)$.

Inequality (2.72c)**:** For fixed $i \in 1, 2$, consider Inequality (2.72c). For $\sum_{u \in S} y_u \in \mathbb{Z}_{\geq 1}$ the inequality is dominated by the sum of Cutset Inequalities for the cuts $(S \cup S_1, S_2)$ and $(S \cup S_2, S_1)$ and thus trivially valid. Now, assume that $\sum_{u \in S} y_u = 0$.

For $\sum_{u \in \bar{S}_i} y_u = 0$, the inequality holds as sum of Cutset Inequalities corresponding to the cuts $(S \cup S_1, S_2)$ and $(S \cup S_2, S_1)$ in absence of compression. If $\sum_{u \in \bar{S}_i} y_u \geq 1$, the inequality is dominated by Inequality (2.72a) and thus valid. □

We conclude the proof that the Extended Cutset Inequalities with the compressor variables y on the connected side S are dominated if $\bar{S}$ is not connected.

Corollary 2.10. *Consider an* NDPC *instance for which a three-partition* $(S, \bar{S}_1, \bar{S}_2)$ *of* V *with the properties* (2.71a)–(2.71a) *exists. The Extended Cutset Inequality*

$$\sum_{uv\in\delta(S)} x_{uv} + \left(\left\lceil Q^{S,\bar{S}_1} + Q^{S,\bar{S}_2}\right\rceil - \left\lceil \frac{Q^{S,\bar{S}_1} + Q^{S,\bar{S}_2}}{\gamma}\right\rceil\right)\sum_{u\in S} y_u \geq \left\lceil Q^{S,\bar{S}_1} + Q^{S,\bar{S}_2}\right\rceil, \quad (2.73)$$

cf. Table 2.1 number 8, is dominated.

Proof. The proof is similar to the one of Corollary 2.8. Let $\overline{S}$ be the disjoint union of $\bar{S}_1$ and $\bar{S}_2$ with $\delta(\bar{S}_1, \bar{S}_2) = \emptyset$. Recall that in any case, the Extended Cutset Inequality is dominated if $\sum_{u\in\bar{S}} y_u \geq 1$. Consider Inequality (2.72a):

$$\begin{aligned}
&\sum_{uv\in\delta(S,\bar{S}_1)} x_{uv} + \sum_{uv\in\delta(S,\bar{S}_2)} x_{uv} \\
&+ \left(\left\lceil Q^{S,\bar{S}_1} + \frac{Q^{\bar{S}_2,\bar{S}_1}}{\gamma} + Q^{S,\bar{S}_2} + \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma}\right\rceil - \left\lceil \frac{Q^{S,\bar{S}_1} + Q^{\bar{S}_2,\bar{S}_1} + Q^{S,\bar{S}_2} + Q^{\bar{S}_1,\bar{S}_2}}{\gamma}\right\rceil\right)\sum_{y\in S} y_u \\
&\geq \left\lceil Q^{S,\bar{S}_1} + \frac{Q^{\bar{S}_2,\bar{S}_1}}{\gamma} + Q^{S,\bar{S}_2} + \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma}\right\rceil. \qquad (2.74)
\end{aligned}$$

Let $\alpha := \sum_{u\in S} y_u$ with $0 \leq \alpha \leq 1$. After subtraction of the y's, the left-hand side of Inequality (2.72a) is larger or equal than

$$\begin{aligned}
&(1-\alpha)\left\lceil Q^{S,\bar{S}_1} + \frac{Q^{\bar{S}_2,\bar{S}_1}}{\gamma} + Q^{S,\bar{S}_2} + \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma}\right\rceil + \alpha\left\lceil \frac{Q^{S,\bar{S}_1} + Q^{\bar{S}_2,\bar{S}_1} + Q^{S,\bar{S}_2} + Q^{\bar{S}_1,\bar{S}_2}}{\gamma}\right\rceil \\
&\geq (1-\alpha)\left\lceil Q^{S,\bar{S}_1} + Q^{S,\bar{S}_2}\right\rceil + \alpha\left\lceil \frac{Q^{S,\bar{S}_1} + Q^{\bar{S}_2,\bar{S}_1}}{\gamma}\right\rceil. \qquad (2.75)
\end{aligned}$$

This shows that the Extended Cutset Inequality is dominated. □

Apparently, in Lemma 2.10, the parameter z_0 describes the minimal amount of traffic which has to be routed on the edges of $\delta(S, \bar{S})$ when there is compression in $\bar{S}_1$ and $\bar{S}_2$ but not in S. In the above lemma, $\lceil Q^{S,\bar{S}_1} + \frac{Q^{\bar{S}_2,\bar{S}_1}}{\gamma} + Q^{S,\bar{S}_2} + \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma}\rceil$ yields a lower bound on this value. In other words, the inequality $\sum_{uv\in\delta(S)} x_{uv} \geq z_0$ is valid for the problem, if $y_u = 1$ for all $u \in \bar{S}$ and $y_u = 0$ for all $u \in S$. Similar as it is done in the proofs of Lemma 2.8 and Lemma 2.9, the inequalities can also be derived by a lifting procedure.

This bound on z_0 is not always tight, as shown in the next example.

Example 2.7. *Consider an* NDPC *instance for which a three-partition* $(S, \bar{S}_1, \bar{S}_2)$ *of* V *with the properties* (2.71a)–(2.71d) *exists and let* $Q^{S,\bar{S}_1} = Q^{S,\bar{S}_2} = \frac{1}{3}$ *and* $Q^{\bar{S}_1,\bar{S}_2} = 1$. *In this case, it is* $z_0 = 2$ *but, since on any edge at least two units of capacity need to be available,* $z_0 := 3 = \lceil Q^{S,\bar{S}_1} + \frac{Q^{\bar{S}_2,\bar{S}_1}}{\gamma}\rceil + \lceil Q^{S,\bar{S}_2} + \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma}\rceil$ *is a stronger parameter choice.*

However, there are cases where the choice $z_0 = \lceil Q^{S,\bar{S}_1} + \frac{Q^{\bar{S}_2,\bar{S}_1}}{\gamma} + Q^{S,\bar{S}_2} + \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma} \rceil$ *is tight. Consider, e.g.,*

$$Q^{S,\bar{S}_1} = \frac{1}{2}, \quad Q^{S,\bar{S}_2} = \frac{1}{5}, \quad Q^{\bar{S}_1,\bar{S}_2} = \frac{9}{5} \text{ and } \gamma = 2. \tag{2.76}$$

Routing $Q^{S,\bar{S}_1}$ *direct,* $Q^{S,\bar{S}_2}$ *to* $\bar{S}_1$ *and then compressed back to* S *and to* $\bar{S}_2$*, and* $Q^{\bar{S}_1,\bar{S}_2}$ *compressed to* S *and then to* $\bar{S}_2$ *consumes exactly* $z_0 = (\frac{9}{10}+\frac{1}{2}+\frac{1}{5}+\frac{1}{10})+\frac{9}{10}+\frac{1}{10}) = 2+1 = 3$ *units of capacity but* $\lceil Q^{S,\bar{S}_1} + \frac{Q^{\bar{S}_2,\bar{S}_1}}{\gamma} \rceil + \lceil Q^{S,\bar{S}_2} + \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma} \rceil = \lceil \frac{9}{10} + \frac{1}{2} \rceil + \lceil \frac{9}{10} + \frac{1}{5} \rceil = 2+2 = 4$.

We generalize this example to the following corollary:

Corollary 2.11. *Consider an* NDPC *instance for which a three-partition* $(S, \bar{S}_1, \bar{S}_2)$ *of* V *with the properties* (2.71a)–(2.71d) *exists. In Inequality* (2.72a) *and in Inequality* (2.72c)*, the strongest choice of* z_0 *is given by*

$$z_0 := \min \left\{ \begin{array}{l} \left\lceil \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma} + Q^{S,\bar{S}_1} \right\rceil + \left\lceil \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma} + Q^{S,\bar{S}_2} \right\rceil, \\ \left\lceil \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma} + \frac{Q^{S,\bar{S}_1}}{\gamma} \right\rceil + \left\lceil \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma} + Q^{S,\bar{S}_2} + Q^{S,\bar{S}_1} + \frac{Q^{S,\bar{S}_1}}{\gamma} \right\rceil, \\ \left\lceil \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma} + Q^{S,\bar{S}_1} + Q^{S,\bar{S}_2} + \frac{Q^{S,\bar{S}_2}}{\gamma} \right\rceil + \left\lceil \frac{Q^{\bar{S}_1,\bar{S}_2}}{\gamma} + \frac{Q^{S,\bar{S}_1}}{\gamma} \right\rceil \end{array} \right\}. \tag{2.77}$$

Proof. When considering the minimal amount of traffic which has to be routed on the edges of $\delta(S, \bar{S})$, we can, without loss of generality, restrict to the following situation:

1. $Q^{\bar{S}_1,\bar{S}_2}$ is always sent compressed from $\bar{S}_1$ to S to $\bar{S}_2$.
2. $\lfloor Q^{S,\bar{S}_i} \rfloor$ is always sent uncompressed, directly from S to $\bar{S}_i$, for $i = 1, 2$.
3. One of the remaining commodities $Q^{S,\bar{S}_i} - \lfloor Q^{S,\bar{S}_i} \rfloor$ is routed directly from S to $\bar{S}_i$.

The minimum in (2.77) describes capacity consumption of the three remaining possibilities: $Q^{S,\bar{S}_i} - \lfloor Q^{S,\bar{S}_i} \rfloor$ is also routed directly, or one of the two commodities, say $Q^{S,\bar{S}_1}$, is not routed directly but from S to $\bar{S}_2$ to $\bar{S}_1$. □

We conclude the subsection with a comparison to the standard NDP problem. For both, NDP_0 and NDP_γ, the complete description of P for 3-node path instances is given by the Cutset Inequalities induced by the cuts $(S \cup \bar{S}_1, \bar{S}_2)$ and $(S \cup \bar{S}_2, \bar{S}_1)$. For the NDPC problem such complete description is not known, even when considering the (Extended) Cutset Inequalities and the Inequalities (2.72a) and (2.72c) (12 in total) as is shown in detail in the following example. One particular reason for this is, that in the corresponding lifting procedures, it is difficult to determine lifting coefficients "closed form" since these coefficients are problem data specific and, with respect to different lifting steps, may also depend on the previously derived coefficients. One example for this is the arguable more complex structure of z_0 in comparison to the traffic demands, respectively in the lifting coefficients, occurring in the two node case.

Table 2.2: The complete description of the three-node path instance as of Example 2.8, derived by Polymake [60].

Nb.	Inequality	Type
1.	$x_{S,\bar{S}_1} + y_S + y_{\bar{S}_1} \geq 2$	Ext. Cutset $(S \cup \bar{S}_2, \bar{S}_1)$
2.	$x_{S,\bar{S}_1} + y_{\bar{S}_2} \geq 2$	Ext. Cutset $(S \cup \bar{S}_2, \bar{S}_1)$
3.	$x_{S,\bar{S}_2} + 3y_S + 3y_{\bar{S}_2} \geq 5$	Ext. Cutset $(S \cup \bar{S}_1, \bar{S}_2)$
4.	$x_{S,\bar{S}_2} + 3y_{\bar{S}_1} \geq 5$	Ext. Cutset $(S \cup \bar{S}_1, \bar{S}_2)$
5.	$x_{S,\bar{S}_1} + x_{S,\bar{S}_2} + 2y_S + 2y_{\bar{S}_1} \geq 7$	
6.	$x_{S,\bar{S}_1} + x_{S,\bar{S}_2} + 2y_S + y_{\bar{S}_1} + y_{\bar{S}_2} \geq 7$	
7.	$x_{S,\bar{S}_1} + x_{S,\bar{S}_2} + 3y_S + 2y_{\bar{S}_2} \geq 7$	
8.	$x_{S,\bar{S}_1} + 2x_{S,\bar{S}_2} + 2y_S + 3y_{\bar{S}_1} \geq 9$	
9.	$2x_{S,\bar{S}_1} + x_{S,\bar{S}_2} + 3y_S + 4y_{\bar{S}_1} \geq 12$	
10.	$2x_{S,\bar{S}_1} + x_{S,\bar{S}_2} + 3y_S + 3y_{\bar{S}_1} + y_{\bar{S}_2} \geq 12$	
11.	$2x_{S,\bar{S}_1} + x_{S,\bar{S}_2} + 6y_S + 4y_{\bar{S}_2} \geq 12$	
12.	$0 \leq y \leq 1$	Trivial inequalities

Example 2.8. *Consider an three-node path instances where* $|S| = |\bar{S}_1| = |\bar{S}_2| = 1$ *with* $Q^{S,\bar{S}_1} = 2.9$, $Q^{S,\bar{S}_2} = 0.3$ *and* $Q^{\bar{S}_1,\bar{S}_2} = 1.7$. *Let* $\gamma = 3$ *and assume that compressors and edge capacities have unit cost. Then, Inequality* (2.72a) *and Inequality* (2.72c) *read as*

$$x_{S,\bar{S}_1} + x_{S,\bar{S}_2} + 2y_S \geq 5, \tag{2.78a}$$

$$x_{S,\bar{S}_1} + x_{S,\bar{S}_2} + 2y_S + 4y_{\bar{S}_1} \geq 7, \qquad x_{S,\bar{S}_1} + x_{S,\bar{S}_2} + 2y_S + 4y_{\bar{S}_2} \geq 7. \tag{2.78b}$$

Denoting $(x_{S,\bar{S}_1}, x_{S,\bar{S}_2}, y_S, y_{\bar{S}_1}, y_{\bar{S}_2})$ *a solution of the linear relaxation of this* NDPC *problem, we obtain the following solution values (rounded to two digits):*

1. *Without any cuts:*
   5.20 *by* $(1.53, 0.67, 1, 1, 1)$.

2. *With (standard) Cutset Inequalities:*
   5.45 *by* $(2, 1, 0.85, 0.85, 0.75)$.

3. *With Cutset & Extended Inequalities:*
   5.69 *by* $(2, 1, 0.69, 1, 1)$.

4. *With Cutset & Extended Inequalities,* (2.72a), (2.72c), *and* z_0 *as in Lemma 2.10:*
   5.72 *by* $(2, 1.56, 0.72, 1, 0.44)$.

5. *With Cutset & Extended Inequalities,* (2.72a), (2.72c), *and* z_0 *as in Corollary 2.11:*
   5.80 *by* $(2, 1.8, 0.8, 2, 0.2)$.

6. *Optimal integer solution:*
   6 $(2, 2, 1, 1, 0)$.

For completeness sake, we remark that the complete description of P as derived by Polymake, see Gawrilow and Joswig [60], is given in Table 2.2. In this case, neither Inequality (2.72a) *nor Inequality* (2.72c) *is facet defining.*

2.3.6 Separating (extended) cutset inequalities

Employing all (Extended-) Cutset Inequalities for all possible cuts in the NDPC formulation is not a realistic option. In the following, we present a straightforward approach for separating these inequalities via an MILP. This approach generalizes the separation of Cutset Inequalities as in the work of Raack et al. [106]. Again, we point out that we focus on the case with constant edge capacities. For the general case, the MILP has to be adapted, i.e., by, in the objective, multiplying each x variable with its capacity. In this case, the separated inequalities have non-constant coefficients for the x variables, e.g., the Cutset Inequalities have the same structure as the ones in Example (1.12).

We employ the following variables: For each $u \neq v \in V$, let $z_{uv} \in \{0,1\}$ denote, whether u and v are in separate sides of the cut. Further, let $d, d_\gamma \in \mathbb{Z}_{\geq 0}$ represent the values $\lceil Q^S \rceil$ and $\lceil \frac{Q^S}{\gamma} \rceil$, respectively. For every node $u \in V$, $\alpha_u \in \{0,1\}$ denotes whether u is in S or in $\overline{S}$. Finally, given a sufficiently large constant $M \in \mathbb{N}$, for $k = 0, \dots, M$, and $u \in V$, let $\alpha_u^k \in \{0,1\}$ be equal to one if and only if $\lceil Q^S \rceil - \lceil \frac{Q^S}{\gamma} \rceil = k$ and u is in S. This way, α_u^k is an enumeration of all possible coefficients of the potential compressor variable coefficients in the Extended Cutset Inequality. Consequentially, we have that

$$\sum_{k=1}^{M} k y_u^* \alpha_u^k = \left(\left\lceil Q^S \right\rceil - \left\lceil \frac{Q^S}{\gamma} \right\rceil \right) \sum_{u \in V} y_u^* \alpha_u = \left(\left\lceil Q^S \right\rceil - \left\lceil \frac{Q^S}{\gamma} \right\rceil \right) \sum_{u \in S} y_u^*. \tag{2.79}$$

A violated Extended Cutset Inequality can be found via an extension of the triangle-formulation for the cut-polytope presented by Barahona and Mahjoub [13]:

$$\min \quad \sum_{k=1}^{M} k y_u^* \alpha_u^k + \sum_{uv \in E} x_{uv}^* z_{uv} - d \tag{2.80a}$$

$$\text{s.t.} \quad -1 + \epsilon \leq \sum_{q \in Q} d^q z_{s^q t^q} - d \leq 0 \tag{2.80b}$$

$$-1 + \epsilon \leq \sum_{q \in Q} \frac{d^q}{\gamma^q} z_{s^q t^q} - d_\gamma \leq 0 \tag{2.80c}$$

$$z_{uv} + z_{wv} + z_{wu} \leq 2 \qquad \forall\ \{u, v, w\} \subset V \tag{2.80d}$$
$$z_{uv} + z_{wv} \geq z_{wu} \qquad \forall\ \{u, v, w\} \subset V \tag{2.80e}$$
$$\alpha_u + \alpha_v \leq 2 - z_{uv} \qquad \forall\ u, v \in V, u \neq v \tag{2.80f}$$
$$\alpha_u + \alpha_v \geq z_{uv} \qquad \forall\ u, v \in V, u \neq v \tag{2.80g}$$
$$\alpha_u^k \leq \alpha_u \qquad \forall\ u \in V, k = 1, \cdots, M \tag{2.80h}$$
$$\sum_{k=1}^{M} \alpha_u^k = \alpha_u \qquad \forall\ u \in V \tag{2.80i}$$
$$M(1 - \alpha_u) + \sum_{k=1}^{M} k\alpha_u^k \geq d - d_\gamma \ \forall\ u \in V \tag{2.80j}$$
$$z_{uv}, \alpha_u, \alpha_u^k \in \{0, 1\}, d, d_\gamma \in \mathbb{Z}_{\geq 0} \tag{2.80k}$$

In this model, Inequality (2.80d) and Inequality (2.80e) establish a feasible cut and Inequality (2.80b) and Inequality (2.80c) recognize the rounded traffic across this cut. Inequalities (2.80f) and (2.80g) determine which nodes are within one side (S) of the cut, and the inequalities (2.80i)– (2.80j) choose the correct α_u^k values for each α_u.

If the resulting objective value is strictly smaller than zero, the optimal solution of the corresponding MILP describes a partition of V, violating an Extended Cutset Inequality (the z_{uv} correspond to the edge variables and the α_u to the compressors variables in S). If the contrary holds, none of Extended Cutset Inequalities is violated.

This program can be adapted to separate Cutset Inequalities (2.27) by omitting the variables d, α_v, α_v^k, adapting the objective, and restricting to Constraint (2.80d), Constraint (2.80e), and Constraint (2.80c). In this case, we obtain:

$$\min \sum_{uv \in E} x_{uv}^* z_{uv} - d \tag{2.81a}$$
$$\text{s.t. } (2.80\text{d}), (2.80\text{e}), (2.80\text{c}) \tag{2.81b}$$
$$z_{uv}, d_\gamma \in \mathbb{Z}_{\geq 0}. \tag{2.81c}$$

2.4 Computational complexity

In practical evaluations, see Koster et al. [86] and also in Section 2.6, we observe that NDPC is much harder to solve than NDP. A short indication for this is given in Table 2.3. In this table, we show the CPU time required for solving two instances, once without compression (NDP) and once with compression (NDPC). The significant increase in computation time induced by the compression aspect motivated our research on the theoretical side: Is NDPC really "more difficult" than classical NDP?

By Definition 2.1, NDPC is a generalization of the NDP problem, see Remark 2.3. Since NDP is $\mathcal{NP}$-hard, see Theorem 1.3, we have the following straight forward result:

Table 2.3: Solution times when introducing compression aspects to NDP. An excerpt from Table 2.5 for instances with *average* traffic load. Characteristics of the instances are shown in the first part. The time limit is 3600 seconds.

Instance	ABILENE16	GERMANY17
$\|V\|$	12	17
$\|E\|$	15	26
$\|Q\|$	66	136
CPU time to optimality:		
NDP	0.17 s	5.52 s
NDPC	13.20 s	time-limit reached

Corollary 2.12. *Network design with compression (*NDPC*) is $\mathcal{NP}$-hard.*

In the remainder of this section, we answer the above question more precisely by investigating the compressor activation/placement aspect, in detail. That is, we show that even in cases where NDP is easy, the NDPC problem remains $\mathcal{NP}$-hard due to the compression functionality. For this purpose, we consider the NDPC problem as described by Definition 2.1. In addition, we present a pseudo-polynomial algorithm for NDPC on trees and an even more restricted polynomial time solvable case.

2.4.1 The compressor placement problem

In this subsection, we focus on subproblems where the routing of the commodities is fixed to a single path, explicitly or implicitly (by graph structure). For the resulting NDPC problem, only the decision on the optimal compressor placement remains. Recall that under these restrictions, the corresponding NDP problem is in $\mathcal{P}$, by Lemma 1.2. However, we define the restricted NDPC problem as the *Compressor Placement Problem*.

Definition 2.9 (Compressor Placement Problem)**.** *The* ***Compressor Placement Problem*** *(CPP) is a* NDPC *problem (G, Q, γ, c_V, c_E), where for every $q \in Q$, a single s^q-t^q path P^q is given. For each commodity $q \in Q$, the complete traffic volume d^q has to be sent along this path.*

Note that, for any given/fixed compressor placement, CPP is in $\mathcal{P}$. We can adapt the MILP formulation for NDPC to obtain a MILP formulation for CPP.

Remark 2.10. *Derived from Remark 2.4, we present an adapted mixed integer linear*

program to model CPP*:*

$$\min \quad \sum_{uv \in E} c_{uv} x_{uv} + \sum_{u \in V} c_u y_u \tag{2.82a}$$

$$\text{s.t.} \quad f^q_{uv} + g^q_{uv} = 1 \qquad \forall\, q \in Q, \forall\, uv \in P^q \tag{2.82b}$$

$$\sum_{\substack{q \in Q: \\ uv \in P^q}} d^q \left(f^q_{uv} + \frac{1}{\gamma^q} g^q_{uv} \right) \leq k_{uv} x_{uv} \qquad \forall\, uv \in E \tag{2.82c}$$

$$-y_u \leq f^q_{vw} - f^q_{uv} \leq y_u \qquad \begin{array}{l} \forall\, q \in Q, \forall\, u \neq s^q, t^q, \\ uv, uw \in P^q \end{array} \tag{2.82d}$$

$$f^q_{s^q u} \geq 1 - y_{s^q} \qquad \forall\, q \in Q, s^q u \in P^q \tag{2.82e}$$

$$f^q_{ut^q} \geq 1 - y_{t^q} \qquad \forall\, q \in Q, ut^q \in P^q \tag{2.82f}$$

$$x_{uv} \in \mathbb{Z}_+,\ y_u \in \{0,1\},\ f^q_{uv} \geq 0, g^q_{uv} \geq 0 \tag{2.82g}$$

In this formulation, flow conservation boils down to the choice between f and g for every edge on a routing path, see Constraint (2.82b)*. At the same time, the Capacity Constraint* (2.82c) *is shortened. Inequalities* (2.82d) *assert that, between consecutive edges uv and uw on path P^q, the relation between f and g may only change if the compressor at node u is activated. Finally, Constraint* (2.82e) *and Constraint* (2.82f) *state that all flow leaving the source/arriving at the target must be uncompressed if no compressor is active there.*

We remark that, for a more compact model, Constraint (2.82b) can be used to substitute g in Constraint (2.82c) and hence, can be omitted. Note that the variables f and g can also be restricted to binary variables since, if some traffic for a commodity q is compressed in an optimal solution, the solution where all the flow for that commodity is compressed is valid as well and does not consume any more capacity.

Backtracking to the NDP problem, we directly have:

Corollary 2.13. *Let a* CPP *instance and a fixed compressor placement y^* be given. An optimal solution (x, y^*, f, g) of the* CPP *problem can be found in polynomial time.*

Proof. We employ the same construction as in the proof of Lemma 1.2. Consider a commodity $q \in Q$ and an edge uv in the routing path of q. The commodity q can be sent compressed if a compressor is active on q's routing path in or before node u and if a compressor is active in or after node v. If the commodity is compressed, it consumes $\frac{d^q}{\gamma^q}$ units of capacity. Otherwise, it takes d^q units. The result follows as in Lemma 1.2 by adding up the capacity consumption of every edge and rounding up the result. □

Once more backtracking to NDP, we can derive a first approximation result. Therefore, we need the following proposition:

Proposition 2.1. *For $z, \lambda \in \mathbb{R}_+$, it holds that*

$$\left\lceil \frac{z}{\lambda} \right\rceil \geq \frac{\lceil z \rceil}{\lambda + 1}. \tag{2.83}$$

PROOF. For $z \in \mathbb{R}_+$, we show the inequality $\left\lceil \frac{z}{\lambda} \right\rceil \geq \frac{\lceil z \rceil}{\lceil \lambda \rceil}$ by

$$\left\lceil \frac{z}{\lambda} \right\rceil \lceil \lambda \rceil - \lceil z \rceil \geq \left\lceil z \frac{\lceil \lambda \rceil}{\lambda} \right\rceil - \lceil z \rceil \geq \lceil z \rceil - \lceil z \rceil = 0. \tag{2.84}$$

Since $\left\lceil \frac{z}{\lambda} \right\rceil \geq \frac{\lceil z \rceil}{\lceil \lambda \rceil}$ implies $\left\lceil \frac{z}{\lambda} \right\rceil \geq \frac{\lceil z \rceil}{\lambda+1}$, the proof is concluded. □

Then, we have:

Lemma 2.11. *Let* $\gamma = \max_{q \in Q} \gamma^q$. CPP *can be* $(\gamma + 1)$*-approximated in polynomial time.*

PROOF. Consider the algorithm which does not activate any compressor ($y_u = 0$ for all $u \in V$) and then solves the resulting NDP instance. By Lemma 1.2, this can be done in polynomial time. We show that the solution $ALG = (x, 0)$ of this algorithm is not worse than $\gamma + 1$ times the optimal solution $OPT = (x^*, y^*)$. Since the routing is fixed, for every edge uv, the uncompressed flow volume on a link is fixed and denoted by d_{uv} in the following. Hence, it holds that

$$\frac{c(ALG)}{c(OPT)} = \frac{\sum_{uv \in E} c_{uv} x_{uv}}{\sum_{uv \in E} c_{uv} x^*_{uv} + \sum_{u \in V} c_u y^*_u} \leq \frac{\sum_{uv \in E} c_{uv} \lceil d_{uv} \rceil}{\sum_{uv \in E} c_{uv} \left\lceil \frac{d_{uv}}{\gamma} \right\rceil} \leq_{(*)} \frac{\sum_{uv \in E} c_{uv} \lceil d_{uv} \rceil}{\frac{1}{\gamma+1} \sum_{uv \in E} c_{uv} \lceil d_{uv} \rceil} \tag{2.85}$$

which equals $\gamma + 1$. Hereby, (*) holds by Proposition 2.1. □

Lemma 2.11 stresses the relation between NDP and NDPC. Given a fixed simple (routing-) path, the difference between the two solution values is bounded by the compression factor plus one. Since $\gamma > 1$, this emphasizes the importance of the compression aspect and its potential efficiency gain for the network. But, as shown in the following, this gain is paid for by an increased computational difficulty of the problem itself.

2.4.2 Theoretical difficulty

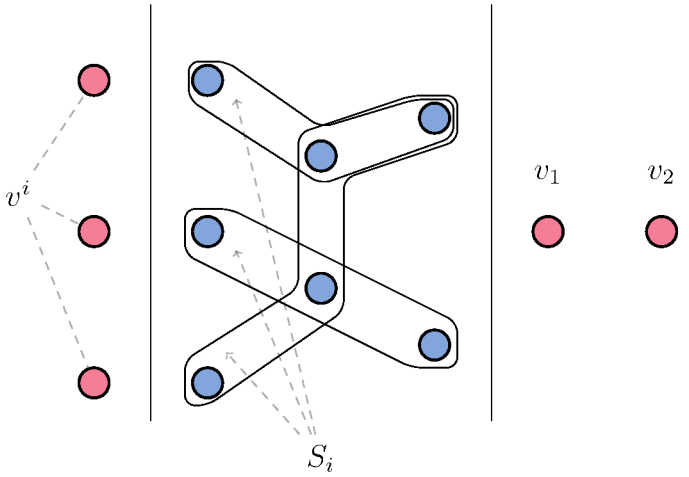
In this subsection, we show that CPP is strongly $\mathcal{NP}$-hard. In order to do so, we show a reduction from the *Hitting Set Problem* (HSP). Recall the definition of HSP:

Definition 2.10 (Hitting Set Problem)**.** *Let a set* U *be given and let* $n \in \mathbb{Z}_+$. *Further, let a collection of subsets of* U *be given as* $\mathcal{S} := \{S_i \subseteq U \mid i = 1, \ldots, n\}$ *and let* $k \in \mathbb{Z}_+$. *The* ***Hitting Set Problem*** *(HSP) asks for a subset* $H \subseteq U$ *with*

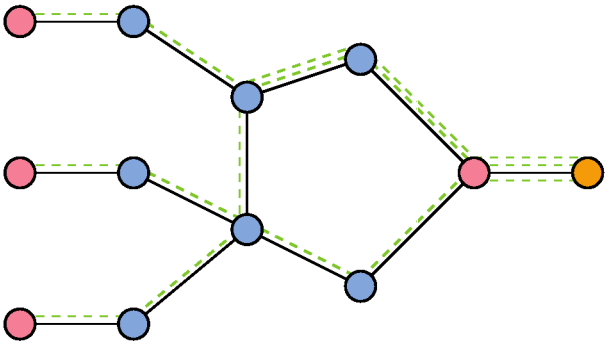
$$|H| \leq k \tag{2.86a}$$
$$H \cap S_i \neq \emptyset \qquad \forall i = 1 \ldots, n \tag{2.86b}$$

or for a proof that such set does not exist.

By Karp [82], it is known that



(a) An extended HSP instance. Blue nodes correspond to the Universe U, red indicate the additional elements.



(b) An extended HSP instance with additional commodities. The red nodes indicate expensive compressors with $c_u = M$, the orange nodes compressor is for free ($c_u = 0$).

Figure 2.13: An extended Hitting Set Problem (HSP) instance with $n = 3$ and the corresponding NDPC/ CPP instance.

Theorem 2.1 (Karp [82])**.** *The Hitting Set Problem is strongly $\mathcal{NP}$-hard.*

Relying on this result, we prove that CPP is also strongly $\mathcal{NP}$-hard.

Theorem 2.2. CPP *is strongly $\mathcal{NP}$-hard.*

Proof. We show the claim by reduction from HSP. Let an instance of HSP be given. We transform this instance into a CPP instance as follows: Let $G = (V, E)$ be a graph with $V = U \cup \{v^i \mid i = 1, \dots, n\} \cup \{v_1, v_2\}$. For each $S_i \in \mathcal{S}$, we fix an arbitrary order on the nodes, such that $S_i = \{v^i_1, \dots, v^i_{|S_i|}\}$. See Figure 2.13 (a) for a sketch of such extended HSP instance.

We add the edges $\{v^i, v^i_1\}$ and $\{v^i_l, v^i_{l+1}\}$ for $l = 1, \dots, |S_i| - 1$, resembling to a path through the nodes of S_i in the above defined sequence (parallel edges are omitted). Further, we add the edges $\{v^i_{|S_i|}, v_1\}$ for all $i = 1, \dots, n$ and the edge $\{v_1, v_2\}$ to E.

For the set of commodities, we assume that every node v^i, $i = 1, \dots, n$, sends flow of volume $\frac{2}{n}$ to node v_2, which has to be routed along the path defined by S_i. We define the capacity of all edges to be one and the compression rate to be $\gamma^q = 2$ for all $q \in Q$. For a sufficient big $M \in \mathbb{Z}_+$, let the objective costs be

$$c_{uv} = \begin{cases} k+1 & uv = \{v_1, v_2\} \\ 0 & else, \end{cases} \quad \text{and} \quad c_v = \begin{cases} 0 & v = v_2 \\ M & v = v_1 \vee v = v^i, i = 1, \dots, n \\ 1 & v \in U. \end{cases} \tag{2.87}$$

By construction, a solution to CPP costs $2k+2$ units if no compressor is deployed since the only cost inducing edge $\{v_1, v_2\}$ bears a load of $n\frac{2}{n} = 2$ units. In the following, if any compressors are deployed, we assume that the compressor in v_2 is always deployed (but not actively counted for the number of total active compressors) and that the compressors in v^i, $i = 1, \dots, n$ and v_1 are never active. See Figure 2.13 (b).

If $k_0 \geq k+1$ compressors are deployed, a feasible solution of CPP costs at least (if all flows can be compressed) $k + 1 + k + 1 = 2k + 2$ units. Thus it is not cheaper than the solution where no compressor is active.

Assuming that no hitting set H of size k exists, any solution which employs $k_0 \leq k$ compressors leaves at least one node set $S_{\bar{i}}$ without an active compressor, otherwise, we have a hitting set of size $k_0 \leq k$ given by the activated compressors. Because of the fixed routing, the flow leaving node $v^{\bar{i}}$ can not be compressed before crossing edge (v_1, v_2). Hence, the minimum flow on this edge amounts to $1 + \frac{1}{n}$ units, such that at least two units of capacity are required, increasing the total cost of such solution to at least $k_0 + 2(k+1) > 2k + 2$.

On the contrary, assuming that U admits a hitting set of size $k_0 \leq k$, a solution to CPP is given by activating compressors on all the nodes corresponding to the hitting set. Hence, all flows can be compressed before reaching edge (v_1, v_2), such that the optimal solution costs at most $k_0 + k + 1 < 2k + 2$.

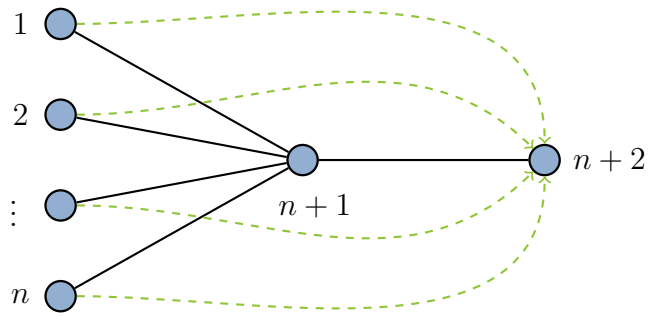


Figure 2.14: CPP on a star. Commodities indicated in green.

In total, an optimal solution to the above described CPP instance shows that either no hitting set of size at most k exists (solution value is larger or equal to $2k+2$) or a hitting set exists (solution value is strictly smaller than $2k+2$) and is given by the elements which correspond to the nodes of activated compressors. □

2.4.3 Special cases

In this subsection, we investigate some special cases of CPP. We show that even on simple graphs (stars), the problem is still (weakly) $\mathcal{NP}$-hard. However, we present a pseudo-polynomial algorithm for CPP on tree instances (including stars) and we conclude this subsection with a polynomial time algorithm for CPP on path instances.

CPP in star networks: a weakly $\mathcal{NP}$-hard case

We show that CPP is (weakly) $\mathcal{NP}$-hard, even on so-called star-instances.

Definition 2.11 (Star-Instances)**.** *Let $n \in \mathbb{Z}_+$ be given. Construct a graph G with $|V| = n+2$ nodes as follows. We refer to the first n nodes as supply nodes, to node $n+1$ as center, and to node $n+2$ as sink. Let $E = \{\{i, n+1\} | i = 1, \dots, n\} \cup \{\{n+1, n+2\}\}$ consist of all connections between the first n nodes and the center node plus the additional edge between the center and the sink.*

Commodities $Q = \{(i, n+2) \mid i = 1, \dots, n\}$ do only exist between the first n nodes and the target node $n+2$. Each commodity q has to be routed directly via $(s^q, n+1, n+2)$. We refer to such instance as ***Star-Instance****.*

See Figure 2.14 for a sketch of a star-instance. The dashed, green lines indicate the commodities. In a star instance, all commodities are routed directly via the central node $n+1$.

Theorem 2.3. CPP *on star-instances is weakly $\mathcal{NP}$-hard.*

PROOF. We show a reduction from the Knapsack Problem (KP). Let a KP instance be given by a set of items $N = \{1, \dots, n\}$. Let $c_i \in \mathbb{Z}_+$ denote the profit and $a_i \in \mathbb{Z}_+$ the weight of item $i \in N$. The KP budget is $B \in \mathbb{Z}_+$. Without loss of generality, we assume

$$\max_{i \in N} a_i \leq B \quad \text{and} \quad \sum_{i \in N} a_i > B. \tag{2.88}$$

We define the corresponding CPP star instance as follows:

Each item $i \in N$ corresponds to one supply node u, we add a central node $n+1$ and a sink $n+2$ with edges as defined in Definition 2.11. For $M > \sum_{i \in N} c_i$, we define

$$c_u := \begin{cases} c_u & u \in N \\ 0 & u = n+2 \\ \infty & u = n+1 \end{cases}, \quad c_{uv} := \begin{cases} 0 & v \in N, u = n+1 \\ M & v = n+1, u = n+2, \end{cases} \tag{2.89}$$

as capacity/compressor costs. We choose any $\gamma > 1$ and for all $i \in N$, we define a commodity q_i with $d^{q_i} = a_i$, $s^{q_i} = i$, and $t^{q_i} = n+2$. No other commodities exist. Finally, we choose the (constant) capacity of all edges to be

$$\frac{1}{\gamma} \sum_{i \in N} a_i + \left(1 - \frac{1}{\gamma}\right) B. \tag{2.90}$$

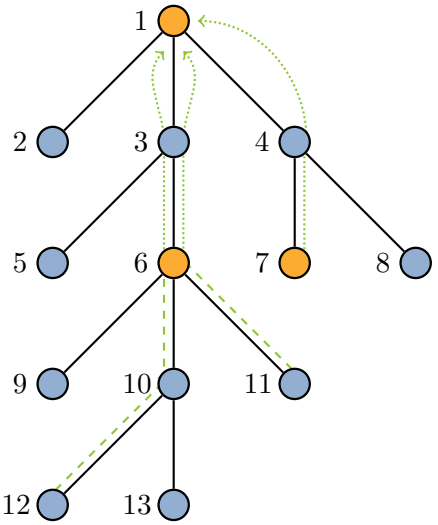
A feasible *baseline* solution of CPP is given by $y_u = 1$ for all $u \in N \cup \{n+2\}$ and $y_{n+1} = 0$. The capacity requirement on $\{u, n+1\}$ is irrelevant, since the cost of the capacities is zero. $x_{n+1,n+2} = 1$ is sufficient, since all commodities can be compressed and hence, the total flow on this edge is $\frac{1}{\gamma} \sum_{i \in N} a_i$, which even leaves a spare capacity of $(1 - \frac{1}{\gamma})B$. Clearly, an optimal solution to CPP is given by the maximal improvements with respect to the baseline solution and any cheaper solution can only be obtained by turning off compressors in the nodes corresponding to N. The extra amount of flow which is added by removing a compressor in node u is $a_u(1 - \frac{1}{\gamma})$. So, the maximal improvements directly correspond to the maximum KP value, i.e., an optimal solution solution of CPP yields an optimal solution of KP and vice versa. □

Compressor placement in tree networks: a pseudo polynomial algorithm

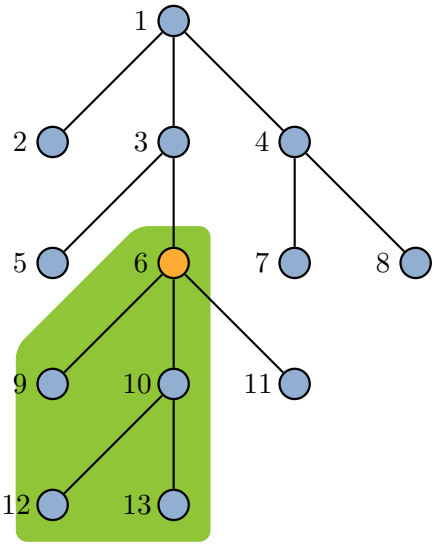
We give a pseudo-polynomial algorithm for CPP on trees, similar to the one proposed by Flippo et al. [56] for network expansion problems and inspired by the work of Johnson and Niemi [80]. Therefore, we derive a series of recursive evaluation formula for the optimal cost of a solution in a (sub) tree and show how these can be evaluated.

Definition 2.12 (Tree-Instances)**.** *Let $G = (V, E)$, $|V| = n$ be a tree and let the nodes be labeled increasingly by breadth first search, starting at the root node $u = 1$. Each edge yields a capacity $k_{uv} \in \mathbb{Z}_+$ per installed unit.*

Commodities $Q = \{(u, 1) \mid u = 2, \dots, n\}$ do only exist between source nodes $s^q \neq 1$ and the target $t^q = 1$. The compression rates are constant, i.e., $\gamma^q := \gamma \in (0, 1)$ for all



(a) Example of a tree instance with three commodities. The routing of the commodities is indicated in green. Orange nodes indicate active compressors, the dotted lines show compressed traffic and dashed ones the uncompressed traffic.



(b) In the tree instance, the subtree $[6, 2]$ is highlighted (green). The subtree is induced by node 6 and its first two children (9 and 10).

Figure 2.15: CPP on a tree instance. All commodities are sent upwards to the root.

$q \in Q$. We assume that all commodity volumes are integer, i.e., that $d^q \in \mathbb{Z}_+$. Each commodity q has to be routed directly on the unique path from s^q to the root $t^q = 1$. We refer to such instance as ***Tree-Instance***.

Note that since every star instance of CPP (see the preceding subsection) corresponds to a tree instance, we directly obtain that CPP is at least weakly $\mathcal{NP}$-hard on trees.

We give a short example for such tree-instance.

Example 2.9. *Consider the tree instance with $|V| = n = 13$ in Figure 2.15.(a). In this instance, three commodities are defined and their routing is indicated by green lines. Dashed lines refer to uncompressed flow, dotted lines refer to compressed flows. In the depicted instance, three compressors (orange) are employed.*

We define the following notation:

Definition 2.13. *Let a tree instance be given. For $u \in V$ and $k \in \mathbb{Z}_+$, we write*

- *$[u, k] \subseteq G$ as the subgraph induced by the node u together with its first k children. $[u, 0]$ is the subgraph consisting only of node u (without any edges).*
- *$p(u) \in V$ as u's predecessor ($u \neq 1$).*

- $s(u,k) \in V$ *for* u*'s* k^{th} *sibling.*
- $a(u) \in \mathbb{Z}_+$ *as the number of children of node* u*. We employ the short form* $a(u,k) := a(s(u,k))$*.*
- $d([u,k])$ *as the traffic volume induced by the subgraph* $[u,k]$*, i.e., the total flow requirement of all the commodities with source* $s^q \in [u,k]$*:*

$$d([u,k]) := \sum_{\substack{q \in Q \\ s^q \in [u,k]}} d^q \tag{2.91}$$

- $d^u = d^q$ *as the flow volume which has to be routed for commodity* $q \in Q$ *with* $s^q = u$*. That is, we identify each commodity with its source node.*

We visualize the notation with the following example.

Example 2.10. *We continue Example 2.9. Consider Figure 2.15.(b). The green area highlights the subtree* $[6,2]$*, given by node 6 and its first two siblings. In the figure, the predecessor of node* 6 *is* $p(6) = 3$ *and its second sibling is* $s(6,2) = 10$*. Furthermore, it is* $a(6) = 3$ *and* $a(6,2) = 2$ *and the flow volume originating from* $[6,2]$*, i.e.,* $d([6,2])$ *is the flow originating from node 12 (there are no commodities associated to 6, 10, and 13), compare Figure 2.15.(a).*

Based on this notation, we define three cost functions, operating on subgraphs $[u,k]$ and referring to the cost of a feasible solution. In detail, these costs refer to the necessary compressors/capacities, which have to be installed in $[u,k]$ to route the traffic volume of all the commodities from source $s^q \in [u,k]$ via u to the common target. Note that there is no outgoing edge for $u = 1$. Additionally, the functions depend on the parameter $f \in \mathbb{Z}_+$, which describes the amount of uncompressed flow on edge $(u, p(u))$.

Definition 2.14. *Let* $\mathcal{G} := \bigcup_{u \in V} \bigcup_{k=0,\dots,a(u)} [u,k]$ *be the collection of all subtrees* $[u,k]$*. For* $[u,k] \subseteq G$ *and* $f \in \mathbb{Z}_+$*, we define*

$$\mathcal{C}_f : \mathcal{G} \to \mathbb{R}_+, [u,k] \mapsto \mathcal{C}\left([u,k], f\right), \tag{2.92a}$$
$$\mathcal{D}_f : \mathcal{G} \to \mathbb{R}_+, [u,k] \mapsto \mathcal{D}\left([u,k], f\right), \tag{2.92b}$$
$$\mathcal{N}_f : \mathcal{G} \to \mathbb{R}_+, [u,k] \mapsto \mathcal{N}\left([u,k], f\right), \tag{2.92c}$$

as the cost induced by $([u,k], f)$ *in an optimal solution, given* $\mathcal{C}$*ompression, respectively* $\mathcal{D}$*ecompression (*$y_u = 1$*), or* $\mathcal{N}$*either compression nor decompression (*$y_u = 0$*), in* u *and an uncompressed flow of* f *units originating in* $[u,k]$ *and leaving the subtree in* u*.*

Note that these costs do *not* contain the capacity which needs to be installed on the edge between u and it's predecessor $p(u)$. By this definition, we have:

Lemma 2.12. *Given a tree instance, the optimal solution value of* CPP *is given by*

$$\min\left\{\mathcal{D}\left([1,a(1)],0\right), \mathcal{N}\left([1,a(1)],0\right)\right\}. \tag{2.93}$$

PROOF. In the root node, no compression takes place and no flow passes through. Both functions give the remaining possibilities for the objective cost. □

In an optimal solution for any node $u \in V$, we only have to compute $\mathcal{C}([u,k],f)$ for $f=0$, since all flow will be compressed in u as a compressor is activated there. Similar, if decompression is assumed to take place in $i \neq 1$, no flow can remain compressed and hence, $\mathcal{D}([u,k],f)$ only has to be computed for $f = d([u,a(u)])$.

In preparation of the pseudo-polynomial algorithm, we give recursive evaluation formula for these cost functions. Therefore, we define starting values for all $[u,0]$. Note that for a singleton $\mathcal{N}([u,0],f)$ without compression/decompression, the outgoing flow value has to be the total volume of the commodity originating in that node.

Lemma 2.13. *For $u \in V \setminus \{1\}$ and $f \in \mathbb{Z}_+$, it is*

$$\mathcal{C}([u,0],0) = c_u, \tag{2.94a}$$
$$\mathcal{D}([u,0],d([u,a(u)])) = c_u, \tag{2.94b}$$
$$\mathcal{N}([u,0],d^u) = 0. \tag{2.94c}$$

PROOF. Since the singletons $[u,0]$ include no edges, no cost of edge capacities have to be accounted for in $[u,0]$. Depending on whether de-/compression is activated, the cost of such singleton is either zero or the cost of the compressor in u. □

For each of the cost functions, we give a recursion:

Lemma 2.14 (Recursion $\mathcal{C}$ompression)**.** *Let $x_0 := d([s(u,k),a(u,k)])$. For every node $u \neq 1$ and $k = 1,\dots,a(u)$, it is*

$$\begin{aligned}\mathcal{C}([u,k],0) = \mathcal{C}([u,k-1],0) \\ + \min\left\{ \begin{array}{l} \mathcal{C}([s(u,k),a(u,k)],0) + c_{s(u,k),u}\left\lceil \frac{x_0}{\gamma k_{s(u,k),u}} \right\rceil, \\ \min\limits_{x \in \{d^{s(u,k)},\dots,x_0\}} \left\{ \begin{array}{l} \mathcal{N}([s(u,k),a(u,k)],x) \\ + c_{s(u,k),u}\left\lceil \frac{x}{k_{s(u,k),u}} + \frac{x_0-x}{\gamma k_{s(u,k),u}} \right\rceil \end{array} \right\} \end{array} \right\}. \end{aligned} \tag{2.95}$$

PROOF. Since the outgoing flow f of u is fixed (and flow conservation holds in u), the minimal cost of a subtree $[u,k]$ with a compressor in u can be split into the minimal cost of the subtree $[u,k-1]$ (with a compressor in u), plus the minimal cost of the remaining subtree induced by sibling $s(u,k)$, namely $[s(u,k),a(u,k)]$, and the cost of the capacity on edge $(s(u,k),u)$.

The minimal cost and the routing of the remaining subtree depends on the (cheapest) action which can take place there ($\mathcal{C}$ or $\mathcal{N}$ only, since $\mathcal{D}$ecompressing makes no sense if the traffic is, again, compressed one level higher). In both cases, this cost has to be evaluated depending on the amount of uncompressed flow x which leaves $s(u,k)$. Clearly, this flow can maximally amount to the complete traffic demand $d([s(u,k),a(u,k)])$ of the subtree. At least, it amounts to the traffic induced by $s(u,k)$. Since all commodities are integer, the flow can only take values in $d^{s(u,k)},\dots,x_0$. □

Lemma 2.15 (Recursion $\mathcal{D}$ecompression)**.** *Let* $x_0 := d([s(u,k), a(u,k)])$*. For every node* $u \neq 1$ *and* $k = 1, \ldots, a(u)$*, it is*

$$\begin{aligned}\mathcal{D}\left([u,k], d([u,a(u)])\right) &= \mathcal{D}\left([u,a(u)], d([u,k-1])\right) \\ &+ \min\left\{\begin{array}{l} \mathcal{C}\left([s(u,k),a(u,k)],0\right) + c_{s(u,k),u}\left\lceil \frac{x_0}{\gamma k_{s(u,k),u}}\right\rceil, \\ \min\limits_{x\in\left\{d^{s(u,k)},\ldots,x_0\right\}}\left\{\begin{array}{l}\mathcal{N}\left([s(u,k),a(u,k)],x\right) \\ +c_{s(u,k),u}\left\lceil \frac{x}{k_{s(u,k),u}} + \frac{x_0-x}{\gamma k_{s(u,k),u}}\right\rceil\end{array}\right\}\end{array}\right\}. \end{aligned} \tag{2.96}$$

Proof. The proof is analogue to the one of Lemma+2.14. Given $\mathcal{D}$ecompression in u, the cost cannot be improved by decompressing at $s(u,k)$, as well. □

Note that $\mathcal{C}\left([u,k],0\right) = \mathcal{D}\left([u,k], d([u,a(u)])\right)$ by these formula. However, we do distinguish them for a correct computation in the case without (de-) compression.

Lemma 2.16 (Recursion $\mathcal{N}$either compression nor decompression)**.** *Define* $x_0 := d([s(u,k),a(u,k)])$*. For* $u \neq 1$*,* $k = 1, \ldots, a(u)$*, and for* $f = d^u \ldots, d([u,k])$*, it is*

$$\begin{aligned}&\mathcal{N}\left([u,k],f\right) = \\ &\min\left\{\begin{array}{l} \mathcal{N}\left([u,k-1],f\right) + \mathcal{C}\left([s(u,k),a(u,k)],0\right) + c_{s(u,k),u}\left\lceil \frac{x_0}{\gamma k_{s(u,k),u}}\right\rceil, \\ \mathcal{N}\left([u,k-1],f-x_0\right) + \mathcal{D}\left([s(u,k),a(u,k)],x_0\right) + c_{s(u,k),u}\left\lceil \frac{x_0}{k_{s(u,k),u}}\right\rceil, \\ \min\limits_{x\in\left\{d^{s(u,k)},\ldots,f-d^u\right\}}\left\{\begin{array}{c}\mathcal{N}\left([u,k-1],f-x\right) + \mathcal{N}\left([s(u,k),a(u,k)],x\right) \\ +c_{s(u,k),u}\left\lceil \frac{x}{k_{s(u,k),u}} + \frac{x_0-x}{\gamma k_{s(u,k),u}}\right\rceil\end{array}\right\}\end{array}\right\}. \end{aligned} \tag{2.97}$$

Proof. The proof is analogue to the proofs of Lemma 2.14 and 2.15. However, given $\mathcal{N}$o (de-) compression in u, all three actions are possible in $s(u,k)$. As a result, given action $\mathcal{N}$ in $s(u,k)$, the uncompressed flow f has to be divided into an x and an $f-x$ flow between the subproblems $[u,k-1]$ and $[s(u,k),a(u,k)]$. □

Denoting $\triangle := \max\limits_{q\in Q} d^q$, the recursive formula can be evaluated in:

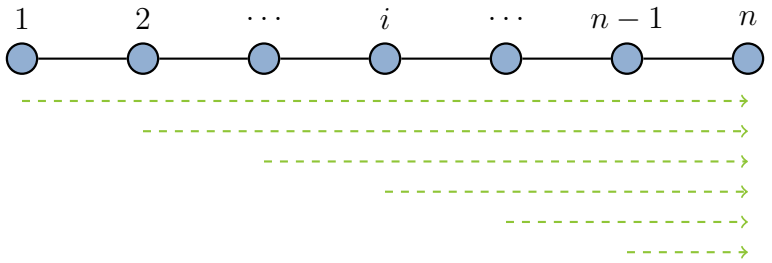
Theorem 2.4. NDPC *on trees can be solved in* $O(n^3\triangle^2)$*.*

Proof. Lemma 2.12 to Lemma 2.16 give the recursions. These can be computed bottom up. Since the underlying graph is a tree, the amount of subtrees $[u,k]$ which have to be evaluated is $\sum_{u\in V} deg(u) = 2n-2$, together with an amount of $n\triangle$ potential f values ($f \leq \sum_{q\in Q} d^q \leq n\triangle$).

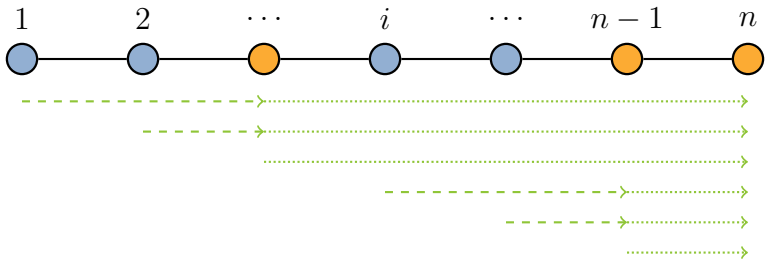
Per subtree $[u,k]$ and value f, the computing costs of $\mathcal{C}$, $\mathcal{D}$, and $\mathcal{N}$ are in $O(n\triangle)$. Together, this yields a runtime of $O(n^3\triangle^2)$. □

Compressor placement in path networks: a polynomial time algorithm

We show that CPP is "easy" in the sense of the existence of a polynomial time algorithm for the special case that G is a path and some further restrictions apply.



(a) Given a solution without compression, all flows (green) are routed uncompressed as is indicated by the dashed lines.



(b) Observe a solution with three active compressors (orange). The compressed flows are indicated by dotted lines.

Figure 2.16: CPP on a path instance.

Definition 2.15 (Path Instances)**.** *Let $G = (V, E)$ be a path, i.e., for $n \in \mathbb{Z}_+$, let $V = \{1, \dots, n\}$ and let $E = \{\{u, u+1\} \,|u \in 1, \dots, n-1\}$. Assume that each edge has a capacity of 1 unit. Define $d \in \mathbb{Z}_+$ and $\gamma \in \mathbb{Z}_+$, such that $\frac{d}{\gamma} \in \mathbb{Z}_+$.*

Constant commodities exist for sources nodes $u \neq n$ to the common target n, i.e., $Q = \{1, \dots, n-1\}$ with $d^q = d$, $s^q = q$, $t^q = n$ for all $q \in Q$. The compression rates are constant, i.e., $\gamma^q = \gamma \in (0, 1)$ for all $q \in Q$. The cost of a compressor and the cost of edge capacities are both constant. Each commodity has to be routed directly, i.e., the commodity q with source s^q has to be routed on $(s^q, s^q + 1, \dots, n)$. We refer to such instance as ***Tree-Instance****.*

We give a short example of such a path instance.

Example 2.11. *See Figure 2.16 for a sketch of a path instance, without any compressors (a) and with three active compressors (b). The green lines indicate where and which commodity (compressed - dotted or uncompressed - dashed) has to be routed.*

At first, we present a general result. Therefore, we consider a path instance as described in Definition 2.15, with the additional degree of freedom that the restrictions on k_{uv}, d^q, γ^q, c_{uv} and c_u are lifted. As the following theorem shows, such path instance can be solved in polynomial time.

Theorem 2.5. *Consider a* CPP *where $G = (V, E)$ is an undirected, simple path. Let $V := \{1, \dots, n\}$ be the sequence of vertices as occurring in the sequence of the path. For all $q \in Q$, let $s^q < t^q$.* CPP *on such instances can be solved in polynomial time.*

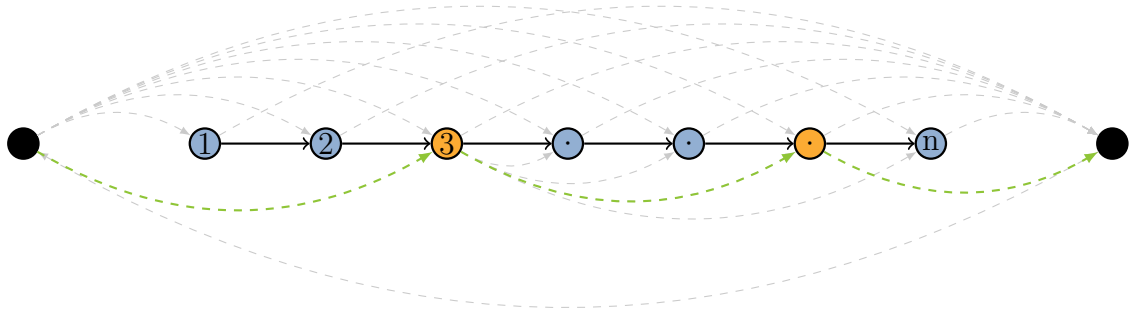


Figure 2.17: CPP on a path instance modeled as shortest path problem. A $0-(n+1)$-path, i.e., $0-3-(n-1)-(n+1)$, is highlighted in green. It corresponds to installing compressors in the nodes 3 and $n-1$ (orange).

Proof. Without loss of generality, we assume that for $u < v$ with $u, v \in V$, there is a single commodity to be routed from u to v. We construct a directed graph $G' = (V', A')$ on which the CPP problem can be expressed as shortest path problem. Let G' be defined as follows: We append a vertex to the start and the end of the original graph, i.e., let $V' := V \cup \{\{0\} \cup \{n+1\}\}$. Correspondingly, let the arc set A' be defined as:

$$\begin{aligned} A' := \quad & \{(0,u) \mid u \in V\} \\ \cup \quad & \{(u,v) \mid u,v \in V, u < v\} \\ \cup \quad & \{(u,n+1) \mid u \in V\} \\ \cup \quad & \{(0,n+1)\}\,. \end{aligned} \tag{2.98}$$

The arcs have the following interpretation: The arcs $(0,u)$ originating in vertex 0 indicate that in the CPP problem, the *first* compressor is active in node $u \in V$. Similar, the arcs $(u,n+1)$ ending in $n+1$ state that the *last* compressor is active at node $u \in V$. The intermediate arcs (u,v) for $u,v \in V$ imply that one compressor is active in both, in node u and in node v but no compressor is active in any node $w \in V$ with $u < w < v$. Finally, the arc $(0,n+1)$ states that no compressors are used at all. See Figure 2.17 for a visualization of the construction.

On the arcs $uv \in A'$, we define a cost function p_{uv} with respect to the cost induced in the CPP instance by the corresponding compressor placements. We employ some additional notation: Let $E_u := \bigcup_{i=1}^{u-1}\{(i,i+1)\}$ be the edges in the path instance *before* node u, let $E_{uv} := \bigcup_{i=u}^{v-1}\{(i,i+1)\}$ be the edges in the path instance *between* node u and node v, and let $E^u := \bigcup_{i=u}^{n-1}\{(i,i+1)\}$ be the edges in the path instance *after* node u.

Consider the arc $(0,u) \in A'$. Let p_{0u} be the minimal cost of the capacity required on the edges E_u if all commodities $q \in Q$ with $s^q < u$ are routed uncompressed (from s^q) to node $\min\{t^q, u\}$ plus the cost of activating a compressor in u.

Similarly, let p_{uv} be the sum of the (minimal) cost of:

1. The capacity required on the edges E_{uv} if all commodities $q \in Q$ with $s^q \leq u$ and

$v \leq t^q$ are routed compressed (from node u) to node v,

2. the capacity required on the edges E_{uv} if all commodities $q \in Q$ with ($s^q \leq u$ and $u \leq t^q < v$) and with ($u < s^q \leq v$ and $t^q \geq v$) are routed uncompressed (from node $\max\{u, s^q\}$) to node $\min\{t^q, v\}$,

3. the cost of activating a compressor in v.

For the arc $(v, n+1) \in A'$, let $p_{v,n+1}$ be the minimal cost of the capacity required on the edges E^v if all commodities $q \in Q$ with $t^q > v$ are routed uncompressed (from $\max\{v, s^q\}$) to node t^q. Finally, let $p_{0,n+1}$ be the cost obtained by the corresponding NDP problem if no compressor is active. Note that all these costs can be computed in polynomial time, see Corollary 2.13.

An optimal solution of the CPP problem is given by the compressor placement as derived from the optimal solution of the shortest $0-(n+1)$-path problem in G', since each solution of CPP, i.e., the compressor placement, the corresponding traffic flows, and its cost, can be identified by an $0-(n+1)$-path (and its cost) and vice versa. □

We exploit the special structure of the path instances to derive a more "formulaic" solution approach which does not rely on a solution of a shortest path problem. The final result of this approach can be found in Theorem 2.6. We introduce some notation:

Definition 2.16. *Let a path instance be given. We write*

- $[n, k]$ *for such instance where we impose that* $\sum_{u \in V} y_u = k$ *for a fixed* $k \in \mathbb{Z}_+$.
- $p_i \in \{1, \ldots, n\}$ *as the position of compressor* $i \in 1, \ldots, k$. *It is* $p_i = u$ *for a unique* $u \in V$ *with* $y_u = 1$. *Without loss of generality, it is* $p_1 < p_2 < \cdots < p_k$, *i.e.,*

$$\sum_{\substack{v \in V: \\ v \leq u}} y_v = i - 1. \tag{2.99}$$

- $s_{p_1,\ldots,p_k}$ *as the cost of an optimal solution with* k *compressors in* $p_1, \ldots, p_k$.

Obviously, the (optimal) solution value of a $[n, k]$ instance does only depend on the position of the active compressors.

Example 2.12. *We continue Example 2.11. Figure 2.16.(b) depicts an* $[n, 3]$ *instance with a fixed compressor placement (orange).*

In the following, we derive cost functions for $[n, k]$ in Lemma 2.18 and in Lemma 2.20. These functions always omit the costs of the k compressors, because they only induce a constant offset. Since compressed flow always has to be decompressed the solutions for $(n, 0)$ and $(n, 1)$ are equivalent and can be computed similar as presented in Lemma 1.2. We exploit that the cost of compressors is independent of the node, and start with:

Lemma 2.17. *Without loss of generality, given an optimal solution for* $[n, k]$, $k \geq 2$, *it is* $p_k = n$. *I.e., the last compressor is always be placed at the last node.*

PROOF. Assume an optimal solution with $p_k < n$. Since all commodities end in n, all of the compressed traffic is decompressed at node p_k and no traffic is compressed and further (since it cannot be decompressed anymore). The solution stays feasible when the compressor p_k is moved to n, because the flow stays compressed longer and thus, consumes less capacity. The adapted solution cannot be more expensive, since it requires the same number of compressors but less capacity between the nodes p_k and n. Hence, without loss of generality, it is $p_k = n$. □

We show how optimal (integer) placements for the compressors can be obtained for $[n, 2]$ (Lemma 2.18 and Lemma 2.19). This is subsequently extended to the general case $[n, k]$ with $k > 2$, resulting in Theorem 2.6. The section is concluded by a polynomial time algorithm for $[n, k]$ in Corollary 2.15.

Lemma 2.18. *Consider $[n, 2]$. Let one compressor be installed in p_1 and the other one in node $p_2 = n$. The lowest cost $c_{p_1,p_2} = c_{p_1,n}$ of the corresponding solution is given by*

$$c_{p_1,n} := \sum_{k=1}^{p_1-1} kd + p_1 \frac{d}{\gamma} (n - p_1) + \sum_{k=p_1+1}^{n-1} (k - p_1)\, d. \tag{2.100}$$

PROOF. The lowest cost is obtained by compressing as many flows as possible (i.e., the total incoming flow) at the first compressor (at node p_1) up to node n. The therefore required capacity is given by the first part of the summation which describes the total flow of all commodities from all nodes before p_1 on the subsequent edges up to node p_1. From p_1 onwards, all the prior commodities and the commodity of p_1 can be sent compressed to the target, such that the necessary capacity to route it to n is described by the second summand. The third summand yields the remaining capacity requirement, i.e., the of all flow of all commodities starting after p_1, which are sent uncompressed all their way to node n. □

These costs do only depend on the position of the first compressor p_1. We derive an optimal (integer) placement for $[n, 2]$.

Lemma 2.19. *For $[n, 2]$ an optimal solution of* CPP *is $p_1 = \lfloor \frac{n}{2} \rfloor$ and $p_2 = n$.*

PROOF. By Lemma 2.18 it is

$$c_{p_1,n} = p_1^2 \left(d - \frac{d}{\gamma} \right) - p_1 n \left(d - \frac{d}{\gamma} \right). \tag{2.101}$$

This function takes its unique minimum in $p_1 = \frac{n}{2}$. Since the function $c_{p_1,n}$ is quadratic, if n is odd, both $\lceil \frac{n}{2} \rceil$ and $\lfloor \frac{n}{2} \rfloor$ are optimal (integer) placements. □

In the following, we extend this result for $k > 2$. At first, we extend the cost formula to the case, were any number of compressor positions is given:

Lemma 2.20. *Let $[n, k]$ be given, together with the corresponding compressor positions p_i, $i = 1, \ldots, k$ with $p_0 := 0 < p_1 < \ldots < p_i < \ldots < p_k = n$. The lowest cost $c_{p_1,\ldots,p_k}$ of the corresponding solution is given by*

$$c_{p_1,\ldots,p_k} := \sum_{i=1}^{k} \left(\sum_{k=p_{i-1}+1}^{p_i-1} (k - p_{i-1})\, d + (p_i - p_{i-1})\, \frac{d}{\gamma}\, (n - p_i) \right). \tag{2.102}$$

Proof. We prove Statement (2.102) by induction over k. Lemma 2.18 yields the statement for $k = 2$. Let Statement (2.102) hold for a fixed $k_0 \in \mathbb{N}_{\geq 2}$, we show that it holds for $k_0 + 1$. By assumption, when deactivating the compressor at p_{k_0}, the costs are

$$\begin{aligned} c_{p_1,\ldots,p_{k_0-1}} = &\sum_{i=1}^{k_0-1} \left(\sum_{k=p_{i-1}+1}^{p_i-1} (k - p_{i-1})\, d + (p_i - p_{i-1})\, \frac{d}{\gamma}\, (n - p_i) \right) \\ &+ \sum_{k=p_{k_0-1}+1}^{n-1} (k - p_{k_0-1})\, d. \end{aligned} \tag{2.103}$$

If the first part of the sum is neglected, the second part describes a path instance on $n - p_{k_0-1}$ nodes with two active compressors. If the compressor at p_{k_0} is activated, we use Lemma 2.18 to replace the second part with the cost, when two compressors (at p_{k_0} and $p_{k_0+1} = n$) are present at that sub instance, i.e., by

$$\sum_{k=p_{k_0-1}+1}^{p_{k_0}-1} (k - p_{k_0-1})\, d + (p_{k_0} - p_{k_0-1})\, \frac{d}{\gamma}\, (n - p_{k_0}) + \sum_{k=p_{k_0}+1}^{n-1} (k - p_{k_0})\, d \tag{2.104a}$$

$$= \sum_{i=k_0}^{k_0+1} \left(\sum_{k=p_{i-1}+1}^{p_i-1} (k - p_{i-1}) d + (p_i - p_{i-1}) \frac{d}{\gamma} (n - p_i) \right). \tag{2.104b}$$

Substituting in $c_{p_1,\ldots,p_k}$ concludes the proof. □

To determine the optimal compressor placements, we give some helpful lemmata:

Lemma 2.21. *The minimum of $c_{p_1,\ldots,p_k}$ satisfies $p_i = \frac{i}{i+1} p_{i+1}$ for $i = 1, \ldots, k-1$.*

Proof. Given k, we prove this by induction over i. Let $i = 1$. For the derivative in direction of p_1, it suffices to restrict to

$$c_{p_1,p_2} := \sum_{k=1}^{p_1-1} kd + \frac{p_1(n-p_1)d}{\gamma} + \sum_{k=p_1+1}^{p_2-1} (k-p_1)d - \frac{p_1(n-p_2)d}{\gamma} \tag{2.105a}$$

$$\begin{aligned} =&(p_1-1)p_1\frac{d}{2} + \frac{dn}{\gamma}p_1 - \frac{dn}{\gamma}p_1^2 + (p_2-1)p_2\frac{d}{2} - (p_2-1)p_1 d \\ &- p_1(p_1+1)\frac{d}{2} + p_1^2 d - \frac{dn}{\gamma}p_1 + p_1 p_2 \frac{d}{\gamma} \end{aligned} \tag{2.105b}$$

$$= p_1^2\left(d - \frac{d}{\gamma}\right) - p_1 p_2 \left(d - \frac{d}{\gamma}\right) + (p_2 - 1) p_2 \frac{d}{2}. \tag{2.105c}$$

Taking the derivative in p_1, we obtain

$$c'_{p_1,p_2} = 2p_1\left(d-\frac{d}{\gamma}\right) - p_2\left(d-\frac{d}{\gamma}\right) \stackrel{!}{=} 0 \tag{2.106a}$$

$$\Leftrightarrow \quad p_1 = \frac{1}{2}p_2. \tag{2.106b}$$

Assuming that the statement holds for a fixed i, we show that it also holds for $i+1$, i.e.,

$$p_{i+1} = \frac{i+1}{i+2}p_{i+2}. \tag{2.107}$$

As above, it suffices to restrict to

$$\begin{aligned} c_{p_{i+1},p_{i+2}} &= \sum_{k=p_i+1}^{p_{i+1}-1} (k-p_i)\,d + (p_{i+1}-p_i)\,(n-p_{i+1})\,\frac{d}{\gamma} \\ &\quad + \sum_{k=p_{i+1}+1}^{p_{i+1}-1} (k-p_{i+1})\,d - p_{i+1}\,(n-p_{i+2})\,\frac{d}{\gamma} && (2.108a) \\ &= (p_{i+1}-1)\,p_{i+1}\frac{d}{2} - (p_{i+1}-1)\,p_i d && (2.108b) \\ &\quad - p_i\,(p_i+1)\,\frac{d}{2} + p_i^2 d + p_{i+1}\,(n-p_{i+1})\,\frac{d}{\gamma} \\ &\quad - p_i\,(n-p_{i+1})\,\frac{d}{\gamma}\,(p_{i+2}-1)\,p_{i+2}\frac{d}{2} \\ &\quad - (p_{i+2}-1)\,p_{i+1}d - p_{i+1}\,(p_{i+1}+1)\,\frac{d}{2} \\ &\quad + p_{i+1}^2 - \frac{dn}{\gamma}p_{i+1} + p_{i+1}p_{i+2}\frac{d}{\gamma} && (2.108c) \end{aligned}$$

$$\begin{aligned} &= -p_i + p_{i+1}d + p_{i+1}n\frac{d}{\gamma} - p_{i+1}^2\frac{d}{\gamma} \\ &\quad + p_i p_{i+1}\frac{d}{\gamma} - p_{i+2}p_{i+1}d + p_{i+1}^2 d \\ &\quad - \frac{dn}{\gamma}p_{i+1} + p_{i+1}p_{i+2}\frac{d}{\gamma} + +p_{i+2}^2\frac{d}{2} - p_{i+2}\frac{d}{2}. && (2.108d) \end{aligned}$$

Taking the derivative in direction of p_{i+1} yields

$$c'_{p_{i+1},p_{i+2}} = -p_i\left(d-\frac{d}{\gamma}\right) + 2p_{i+1}\left(d-\frac{d}{\gamma}\right) - p_{i+2}\left(d-\frac{d}{\gamma}\right), \tag{2.109}$$

and by assumption

$$(2.109) = -\frac{i}{i+1}p_{i+1}\left(d-\frac{d}{\gamma}\right)+2p_{i+1}\left(d-\frac{d}{\gamma}\right)-p_{i+2}\left(d-\frac{d}{\gamma}\right)\stackrel{!}{=}0 \tag{2.110a}$$

$$\Leftrightarrow \quad \frac{i+2}{i+1}p_{i+1}\left(d-\frac{d}{\gamma}\right)=p_{i+2}\left(d-\frac{d}{\gamma}\right) \tag{2.110b}$$

$$\Leftrightarrow \quad p_{i+1}=\frac{i+1}{i+2}p_{i+2}. \tag{2.110c}$$

Note, that $p_k = n$, such that for $i+2=k$, p_{i+2} is replaced by n in the above. The induction principle concludes the proof. □

Iteratively, Lemma 2.21 translates to

Corollary 2.14. *$c_{p_1,\dots,p_k}$ has a unique minimum at $p_i = \frac{i}{k}n$, $i = 1,\dots,k$.*

Proof. By Lemma 2.21, we have

$$p_i=\frac{i}{i+1}p_{i+1}=\frac{i}{i+1}\frac{i+1}{i+2}\cdots\frac{k-1}{k}p_k=\frac{i}{k}n. \tag{2.111}$$

□

To obtain an optimal integer solution for $[n,k]$, simple rounding is sufficient:

Lemma 2.22. *Let $[n,k]$ be given. For every compressor $i = 1,\dots,k$, the optimal placement p_i is $\lfloor\frac{in}{k}\rfloor$ or $\lceil\frac{in}{k}\rceil$.*

Proof. By Corollary 2.14 the costs function has its unique minimum at the fractional values of $\frac{in}{k}$. Since the cost function is monotonously increasing from the global minimum, the optimal integer solution has to be at one of the neighboring integers. □

Since, there are exponentially many possibilities to round, we describe one way of rounding for obtaining an optimal solution. In the following, assume, without loss of generality, that $\frac{n}{k}$ is non-integer, otherwise no rounding is necessary and Corollary 2.14 gives the optimal positions.

Lemma 2.23. *Consider $[n,k]$, $k \geq 2$ and a compressor placement p_i. For a fixed $i \in \{1,\dots,k-1\}$ let p_{i-1}, p_i and p_{i+1} such that the distance between the compressors $s_1 := p_i - p_{i-1} \neq p_{i+1} - p_i =: s_2$ are not equal (if $k=2$, $p_{i-1} := 0$). Then, the solution obtained by the compressor placement $p_{i-1}, p_{i-1}+(p_{i+1}-p_i), p_{i+1}$, that is, reversing/exchanging the lengths of the uncompressed path (s_1,s_2) to (s_2,s_1), yields the same costs.*

Proof. Without loss of generality, all traffic before p_{i-1} and after p_{i+1} in both placements can be neglected since its capacity requirements/compressor placements do not

differ. Further, we assume that $p_{i-1} = 0$. We subtract both costs from each other:

$$\sum_{k=1}^{p_i} kd + p_i\,(p_{i+1} - p_i)\,\frac{d}{\gamma} + \sum_{k=p_i+1}^{p_{i+1}-1} (k - p_i)\,d$$
$$- \sum_{k=1}^{p_{i+1}-p_i} kd + (p_{i+1} - p_i)\,p_i\frac{d}{\gamma} + \sum_{k=p_{i+1}-p_i+1}^{p_{i+1}-1} (k - p_{i+1} + p_i)\,d \tag{2.112a}$$
$$= \sum_{k=1}^{s_1} kd + s_1 s_2 \frac{d}{\gamma} + \sum_{k=s_1+1}^{s_1+s_2-1} (k - s_1)\,d$$
$$- \sum_{k=1}^{s_2} kd + s_1 s_2 \frac{d}{\gamma} + \sum_{k=s_2+1}^{s_1+s_2-1} (k - s_2)\,d \tag{2.112b}$$
$$= \sum_{k=s_2}^{s_1-1} kd + \sum_{k=1}^{s_1+s_2-1} (k - s_1)\,d - \sum_{k=1}^{s_1} (k - s_1)\,d$$
$$- \sum_{k=1}^{s_1+s_2-1} (k - s_s)\,d - \sum_{k=1}^{s_s} (k - s_s)\,d \tag{2.112c}$$
$$= \sum_{k=s_2}^{s_1-1} kd - (s_1 + s_2 - 1)\,(s_1 - s_2) + s_1 s_1 d - s_2 s_2 d - \sum_{k=s_1+2}^{s_1} kd = 0. \tag{2.112d}$$

□

Lemma 2.24. *For $[n, k]$, with $\frac{n}{k}$ non-integer, let an optimal compressor placement be given. For any $i = 0, \ldots, k-1$ and $s := p_{i+1} - p_i$ it holds that $\left\lfloor \frac{n}{k} \right\rfloor \leq s \leq \left\lceil \frac{n}{k} \right\rceil$.*

Proof. Assume that there exists a compressor placement with an $s \geq \left\lceil \frac{n}{k} \right\rceil + 1$, that is, let $p_{i+1} = p_i + \left\lceil \frac{n}{k} \right\rceil + 1$. Since

$$n = k\frac{n}{k} < k \left\lceil \frac{n}{k} \right\rceil, \tag{2.113}$$

there has to be an $\overline{s}$ with $\overline{s} \leq \left\lceil \frac{n}{k} \right\rceil - 1$ as the distance between two compressors $\overline{i}$ and $\overline{i} + 1$. By the above corollary we can assume that s and $\overline{s}$ are neighboring each other, i.e., let $\overline{i} := i + 1$. It is $\overline{s} < s$, but by Corollary 2.19, p_{i+1} has to be at $p_i + \frac{s+\overline{s}}{2} < p_{i+1}$, a contradiction to the optimality of the assignment.

The contradiction for $s \leq \left\lfloor \frac{n}{k} \right\rfloor - 1$ is obtained analogously. □

Since Lemma 2.24 describes the possible step-length/distance between compressors, we can derive the following lemma.

Lemma 2.25. *In an optimal compressor placement $[n, k]$, there are $b := n \mod k$ steps of length $\left\lceil \frac{n}{k} \right\rceil$ and $a := k \left\lceil \frac{n}{k} \right\rceil - n$ steps of length $\left\lfloor \frac{n}{k} \right\rfloor$.*

PROOF. By the above lemma, $\left\lfloor \frac{n}{k} \right\rfloor$ and $\left\lceil \frac{n}{k} \right\rceil$ are the possible step-lengths. Thus, the relation of these length has to satisfy

$$a \left\lfloor \frac{n}{k} \right\rfloor + b \left\lceil \frac{n}{k} \right\rceil = n \quad \textit{and} \quad a + b = k. \tag{2.114}$$

Inserting $b = k - a$ into the first inequality yields

$$a \left\lfloor \frac{n}{k} \right\rfloor + \left(k \left\lfloor \frac{n}{k} \right\rfloor + k - a \left\lfloor \frac{n}{k} \right\rfloor - a \right) = a \tag{2.115a}$$

$$\Leftrightarrow \quad a = k \left\lceil \frac{n}{k} \right\rceil - n. \tag{2.115b}$$

Hence, it is

$$b = k - k \left\lceil \frac{n}{k} \right\rceil + n = n - k \left\lfloor \frac{n}{k} \right\rfloor = n \mod k. \tag{2.116}$$

□

With the help this lemma, we determine the optimal compressor placements.

Theorem 2.6. *Given $[n, k]$ with $k \geq 2$, an optimal compressor placement is given by $p_i = \left\lfloor \frac{in}{k} \right\rfloor$ for $i = 1, \ldots, k$.*

PROOF. Always rounding down yields step-lengths of size $\left\lfloor \frac{n}{k} \right\rfloor$ and $\left\lceil \frac{n}{k} \right\rceil$. Since there are k steps, adding up to n nodes, this solution satisfies Lemma 2.25 and is optimal. □

We exploit this result to conclude that the path instances can be solved by a polynomial time algorithm:

Corollary 2.15. *The path instances can be solved in polynomial time.*

PROOF. For $k = 2, \ldots, n$, $k \neq 1$ we evaluate the costs by an optimal compressor placement as described in Theorem 2.6. The overall optimal is given by the minimum of the $n - 1$ costs (adding the previously omitted constant costs of k compressors) and the costs where no compressor is activated at all. □

We point out that, since the number of active compressors is fixed, the optimal compressor placement for $[n, k]$ is independent from the actual value of γ. However, it is clear that the optimal solution value and the number of active compressors in this solution will depend on that value.

2.5 Addressing the case of data uncertainty

In this section, we focus on the NDPC problem subject to data uncertainty. That is, in many practical applications, it is reasonable to assume that, over time, the traffic may substantially vary in its data volume and in its potential compression rate. For instance, a movie streaming service may have more or less active customers and therefore,

different traffic requirements. In particular, during the peak hours, many movies are requested, while during night time less demands occur. Similarly as the traffic volume changes over the day, the compression ratio also varies as relatively more or less unique content is requested by the customers.

In this regard, we assume that some parameters of the NDPC problem, e.g., the demand values d^q and the compression ratios γ^q, are not known *a priori*. Consequently, any solution to the NDPC problem has to be found without exact knowledge of these parameters. As indicated in Section 1.4, classical approaches to circumvent data uncertainty typically consider a so-called *worst case* setting, so to guarantee that the network will remain operational even for peak requirements. Although guaranteeing feasibility, this practice comes at an often unnecessary cost as, in many cases, it is very unlikely for every parameter to simultaneously be at its peak. Indeed, in a number of practical cases, it is reasonable to assume that the probability that *all* parameters simultaneously reach their peak values is fairly small. This is reasonable, in our example, when assuming that the streaming service operates world wide, i.e., not all peak hours take place for *all* customers at the same time.

Consequently, the idea is to look for a solution where the network is provisioned for parameter values which are smaller than their peak values, thus guaranteeing that the substrate network has sufficient capacity for *almost all* the traffic configurations, only neglecting a few unlikely cases. This way, we are likely to obtain more profitable solutions where less additional capacities are installed, thus avoiding costly issues of over-provisioning the network.

For this purpose, we apply the two concepts for data uncertainty, i.e., Γ-robustness and two-source robustness (as introduced in Section 1.4) to the NDPC problem. In these settings, the parameter Γ corresponds to the amount of traffic, respectively the degree of compression, the network is provisioned for. In both models, the uncertain coefficients are described by random events. Based on this, they impose that, given the parameter Γ, any constraint of the problem holds if at most Γ many events deviate from their expected outcome.

Throughout this section, we focus on the NDPC problem as presented in Definition 2.3. We assume that the capacity function $k_{uv} = 1$ is constant for all edges uv, that is, exactly one unit of capacity is employed per installed edge module. In particular, we point out that γ^q is not constant, but depends on the corresponding commodity $q \in Q$.

2.5.1 Applying bijective uncertainty

We assume that a parameter $\Gamma \in \mathbb{Z}_+$ is given. The bijective uncertainty (Γ-robustness) can be applied to the NDPC problem as described in Definition 2.4 to obtain a robust MILP formulation. Here, we focus on uncertainties within the Capacity Constraint (2.2c). For the sake of notation, we assume that the compression ratios, given as

$\frac{1}{\gamma^q}$ in Section 2.2, are now written as $\lambda^q := \frac{1}{\gamma^q}$. Consider Constraint (2.2c), rewritten as

$$\sum_{q\in Q} \left(d^q \left(f^q_{uv} + f^q_{vu}\right) + d^q\lambda^q \left(g^q_{uv} + g^q_{vu}\right)\right) \leq x_{uv} \qquad \forall\, uv \in E. \tag{2.117}$$

As one can see, there are two different types of coefficients: d^q and $d^q\lambda^q$ (we assume that the coefficient in front of x_{uv} is certain). According to the Γ-robust setting, each coefficient is given by a unique event which realizes in an interval. Such interval is defined by the expected value of the outcome of the event $\bar{d}^q$, respectively $\overline{d^q\lambda^q}$, and the maximal deviation from this expected value $\hat{d}^q$, respectively $\widehat{d^q\lambda^q}$. Identifying each coefficient with its underlying event, we write

$$d^q \in \left[\bar{d}^q - \hat{d}^q, \bar{d}^q + \hat{d}^q\right], \quad \text{and} \tag{2.118a}$$

$$d^q\lambda^q \in [\overline{d^q\lambda^q} - \widehat{d^q\lambda^q}, \overline{d^q\lambda^q} + \widehat{d^q\lambda^q}]. \tag{2.118b}$$

Note that in this case, the coefficients/events are not independent. That is, if the coefficient of f^q_{uv} deviates, the coefficient of f^q_{vu} deviates as well. Hence, $f^q_{uv} + f^q_{vu}$, respectively $g^q_{uv} + g^q_{vu}$, is interpreted as a single variable. We apply Theorem 1.5 to obtain a Γ-robust MILP formulation of NDPC:

Corollary 2.16. *Let $\Gamma \in \mathbb{Z}_+$ be given. Let $\pi_{uv} \geq 0$ for all $uv \in E$, $\rho^1_{uv,q} \geq 0$ for all $uv \in E, q \in Q$ (corresponding to the f variables), and $\rho^2_{uv,q} \geq 0$ for all $uv \in E, q \in Q$ (corresponding to the g variables). By Theorem 1.5, the* Γ*-robust* NDPC *problem can be obtained by replacing Constraint* (2.2c) *respectively* (2.117) *by*

$$\sum_{q\in Q} \left(\bar{d}^q \left(f^q_{uv} + f^q_{vu}\right) + \overline{d^q\lambda^q} \left(g^q_{uv} + g^q_{vu}\right)\right) + \Gamma\pi_{uv} + \sum_{q\in Q} \rho^1_{uv,q} + \sum_{q\in Q} \rho^2_{uv,q} \leq x_{uv} \qquad \forall\, uv \in E \tag{2.119a}$$

$$\pi_{uv} + \rho^1_{uv,q} \geq \hat{d}^q \left(f^q_{uv} + f^q_{vu}\right) \qquad \forall uv \in E, q \in Q \tag{2.119b}$$

$$\pi_{uv} + \rho^2_{uv,q} \geq \widehat{d^q\lambda^q} \left(g^q_{uv} + g^q_{vu}\right) \qquad \forall uv \in E, q \in Q. \tag{2.119c}$$

Proof. Consider (1.25a)–(1.25d). □

With respect to the following subsection, we remark that the above problem is the corresponding (see Definition 1.12) Γ-robust problem to the two-source robust problem as given in Corollary 2.19 (below).

Now, one could argue that, in fact, d^q and λ^q should be treated as two different sources of uncertainty. Therefore, we assume that

$$d^q \in \left[\bar{d}^q - \hat{d}^q, \bar{d}^q + \hat{d}^q\right] \quad \text{and} \quad \lambda^q \in \left[\bar{\lambda}^q - \hat{\lambda}^q, \bar{\lambda}^q + \hat{\lambda}^q\right] \qquad \forall\, q \in Q. \tag{2.120}$$

This can be incorporated into the above problem by defining

$$\widehat{d^q\lambda^q} := \bar{d}^q\hat{\lambda}^q + \hat{d}^q\bar{\lambda}^q + \hat{d}^q\hat{\lambda}^q, \tag{2.121}$$

which assures that the possible deviations of the composed parameter $d^q\lambda^q$ are contained in the interval which describes the possible realization of the coefficient. However, in this setting, if a coefficient $d^q\lambda^q$ is assumed to deviate maximally, it implies that both d^q and λ^q deviate maximally at the same time. In this context, a more fine grained control over the single random influences, d^q and λ^q, could be desired. While we refer to the following subsection for such a model, the cases where only a single influence, i.e., d^q or λ^q deviates, can still be modeled by Γ-Robustness.

At first, consider the case that (only) d^q is deviating, i.e.,

$$d^q \in \left[\bar{d}^q - \hat{d}^q, \bar{d}^q + \hat{d}^q\right] \quad \text{and} \quad \lambda^q = \bar{\lambda}^q \qquad \forall\, q \in Q. \tag{2.122}$$

As above, we have

Corollary 2.17. *Let $\Gamma \in \mathbb{Z}_+$ be given. Let $\pi_{uv} \geq 0$ for all $uv \in E$, $\rho^1_{uv,q} \geq 0$ for all $uv \in E, q \in Q$ (corresponding to the f variables), and $\rho^2_{uv,q} \geq 0$ for all $uv \in E, q \in Q$ (resembling to the g variables). By Theorem 1.5, the Γ-robust* NDPC *problem can be obtained by replacing Constraint* (2.2c) *respectively* (2.117) *by*

$$\begin{aligned}
&\sum_{q\in Q}\left(\bar{d}^q\left(f^q_{uv} + f^q_{vu}\right) + \bar{d}^q\bar{\lambda}^q\left(g^q_{uv} + g^q_{vu}\right)\right) \\
&\qquad + \Gamma\pi_{uv} + \sum_{q\in Q}\rho^1_{uv,q} + \sum_{q\in Q}\rho^2_{uv,q} \leq x_{uv} && \forall\, uv \in E && \text{(2.123a)}\\
&\pi_{uv} + \rho^1_{uv,q} \geq \hat{d}^q\left(f^q_{uv} + f^q_{vu}\right) && \forall uv \in E, q \in Q && \text{(2.123b)}\\
&\pi_{uv} + \rho^2_{uv,q} \geq \hat{d}^q\bar{\lambda}^q\left(g^q_{uv} + g^q_{vu}\right) && \forall uv \in E, q \in Q. && \text{(2.123c)}
\end{aligned}$$

The case that only λ^q is deviating, where

$$d^q = \bar{d}^q \quad \text{and} \quad \lambda^q \in \left[\bar{\lambda}^q - \hat{\lambda}^q, \bar{\lambda}^q + \hat{\lambda}^q\right] \qquad \forall\, q \in Q, \tag{2.124}$$

is modeled analogously.

Corollary 2.18. *Let $\Gamma \in \mathbb{Z}_+$ be given. Let $\pi_{uv} \geq 0$ for all $uv \in E$ and let $\rho^2_{uv,q} \geq 0$ for all $uv \in E, q \in Q$ (corresponding to the g variables). By Theorem 1.5, the Γ-robust* NDPC *problem can be obtained by replacing Constraint* (2.2c) *respectively* (2.117) *by*

$$\begin{aligned}
&\sum_{q\in Q}\left(\bar{d}^q\left(f^q_{uv} + f^q_{vu}\right) + \bar{d}^q\bar{\lambda}^q\left(g^q_{uv} + g^q_{vu}\right)\right) \\
&\qquad + \Gamma\pi_{uv} + \sum_{q\in Q}\rho^2_{uv,q} \leq x_{uv} && \forall\, uv \in E && \text{(2.125a)}\\
&\pi_{uv} + \rho^2_{uv,q} \geq \bar{d}^q\hat{\lambda}^q\left(g^q_{uv} + g^q_{vu}\right) && \forall uv \in E, q \in Q. && \text{(2.125b)}
\end{aligned}$$

We point out that the latter case has been extensively studied by Coudert et al. [43]. For the case that both, d^q and λ^q are uncertain, we refer to the following section.

We conclude by rephrasing the above results from an application perspective: In summary, there are two possibilities to apply Γ-robustness to the NDPC problem. The first one is to aggregate the two uncertain influences into a single one and then applying Γ-robustness to the "composed coefficient". In this case, we obtain the problem stated in Corollary 2.16. The other possibility is to apply Γ-robustness to a single influence, i.e., applying Γ-robustness to the single source of uncertainty and replacing the other source of uncertainty with its expected, respectively its nominal, outcome. This way, one source of uncertainty is replaced by a deterministic input. The resulting problems are stated in Corollary 2.17 and in Corollary 2.18. Of course, instead of considering the nominal values $\bar{d}^q$, receptively $\bar{\lambda}^q$, in these problems, one can also consider more conservative variants where, e.g., $\bar{d}^q + \hat{d}^q$, respectively $\bar{\lambda}^q + \hat{\lambda}^q$, replace one of the sources of uncertainty. In practice, the decision which of these models is to be preferred is not trivial and depends on the specific situation in which the NDPC problem arises. An exemplary discussion of the different models with respect to a specific dataset is given in the next section, in particular in Subsection 2.6.4.

2.5.2 Applying two-source uncertainty

As mentioned in the previous subsection, the more "natural" setting for the uncertain behavior of the coefficients within the Capacity Constraint (2.2c) respectively (2.117) is that, each coefficient is commonly formed by two events. However, when identifying the parameters d^q and λ^q, with these events, i.e., assuming that

$$d^q \in \left[\bar{d}^q - \hat{d}^q, \bar{d}^q + \hat{d}^q\right] \quad \text{and} \quad \lambda^q \in \left[\bar{\lambda}^q - \hat{\lambda}^q, \bar{\lambda}^q + \hat{\lambda}^q\right] \qquad \forall\, q \in Q, \tag{2.126}$$

it appears that the setting of the two-source robust problem is not precisely met. Indeed, a single event (parameter) d^q, respectively γ^q, influences four (two) of the variables (coefficients) per row. We refer to Figure 2.18 for a sketch of this behavior. As a consequence, the coefficients (events) are not independent.

Nevertheless, the random influences can be categorized into two groups, one group determining d^q and one determining λ^q. Again, we impose that, for the robust NDPC problem, a solution has to be feasible, if at most $\Gamma_1 \in \mathbb{Z}_+$ and $\Gamma_2 \in \mathbb{Z}_+$ many of these events deviate from their expectation per group. Since, in Definition 1.11, the robust constraints do not require the underlying events to be independent or even unique, we can express this requirement in the same way as stated there and obtain the same results (i.e., a compact reformulation). With a slight abuse of notation, we refer to the resulting problem as the two-source robust NDPC problem.

Applying this construction to the NDPC problem, we obtain the following corollary.

Corollary 2.19. *Let $\pi_{uv,1}, \pi_{uv,2} \geq 0$ for all $uv \in E$ and let $\rho_{uv,q}, \tau_{uv,q}, \sigma_{uv,q}, \nu_{uv,q} \geq 0$ for all $uv \in E, q \in Q$. By Theorem 1.5, the two-source robust* NDPC *problem can be*

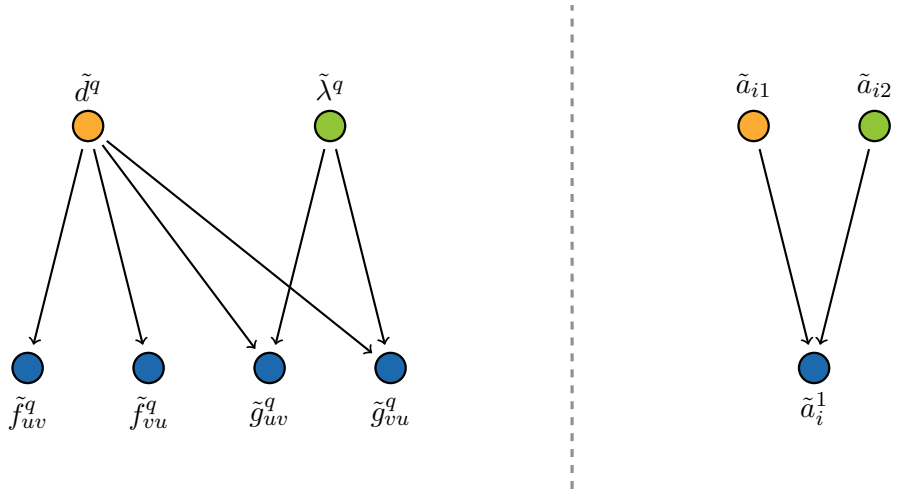


Figure 2.18: Uncertain influences on the coefficients as for the NDPC problem (left) in comparison to the two-source robustness (right). The coefficients are depicted in blue, random influences are indicated in orange for the demand volumes and in green for the compression ratios.

obtained by replacing the Capacity Constraint (2.2c) *by*

$$\sum_{q\in Q}\left(\bar{d}^q\left(f^q_{uv}+f^q_{vu}\right)+\bar{d}^q\bar{\gamma}^q\left(g^q_{uv}+g^q_{vu}\right)\right) \tag{2.127a}$$

$$+\Gamma_1\pi_{uv,1}+\Gamma_2\pi_{uv,2}+\sum_{q\in Q}\tau_{uv,q}+\sum_{q\in Q}\nu_{uv,q}\leq x_{uv} \qquad \forall\, uv\in E \tag{2.127b}$$

$$\pi_{uv,1}-\rho_{uv}+\tau_{uv}\geq \hat{d}^q\bar{\lambda}^q\left(f^q_{uv}+f^q_{vu}+g^q_{uv}+g^q_{vu}\right) \qquad \forall\, uv\in E, q\in Q \tag{2.127c}$$

$$\pi_{uv,2}-\sigma_{uv}+\nu_{uv}\geq \bar{d}^q\hat{\lambda}^q\left(g^q_{uv}+g^q_{vu}\right) \qquad \forall\, uv\in E, q\in Q \tag{2.127d}$$

$$\rho_{uv}+\sigma_{uv}\geq \hat{d}^q\hat{\lambda}^q\left(g^q_{uv}+g^q_{vu}\right) \qquad \forall\, uv\in E, q\in Q. \tag{2.127e}$$

Proof. Compare (1.33b)– (1.33e). □

2.6 Computational studies

In this section, we present computational experiments on the results shown in the preceding sections. Therefore, we split this section into four parts. At first, in Subsection 2.6.1, we briefly introduce the dataset. In the following Subsection 2.6.2, we give a comparison between the NDP problem and the NDPC problem on this dataset. In Subsection 2.6.3, we consider the NDPC problem formulation with the additional cutting planes as presented and discussed in Section 2.3. Finally, in Subsection 2.6.4, we evaluate the NDPC problem under data uncertainty as described in Section 2.5.

Table 2.4: Characteristics of the six test instances as used in Koster et al. [85].

Name	Network	#Nodes	#Edges	#Commodities	Aggregation
ABILENE8	abilene	12	15	66	week 8
ABILENE16	abilene	12	15	66	week 16
GEANT4	geant	22	36	231	week 4
GEANT13	geant	22	36	231	week 13
GERMANY17	germany17	17	26	136	day 1
GERMANY50	germany50	50	89	1044	day 1

2.6.1 Introducing the dataset

Throughout this section, we adopt the same dataset as presented in the works of Koster et al. [85], however, suitablly extended for the case of the NDPC problem. That is, in the cited paper, real-life traffic traces for networks as specified in the SNDLIB [100] have been used to computationally evaluate the (Γ-robust) NDP problem. In detail, the four networks ABILENE, GEANT, GERMANY17, and GERMANY50 have been considered. For each of these networks, traffic traces are specified over a time frame of six months and with a granularity of five to 15 minutes. From this data, single weeks, respectively days, have been aggregated as described by Koster et al. [85] to obtain six instances for the NDP problem. We refer to Table 2.4 for some statistics on the resulting instances. In the data files, for each commodity $q \in Q$, three different traffic volumes d^q are specified (*min*, *average*, *peak*) from which we employ the *average* and the *peak* values. We will point out which of these we consider when necessary. For all instances, the edge capacities and their costs are constant with $k_{uv} = 40,000$ and $c_{uv} = 38.84$ for all $uv \in E$.

As specified in the SNDLIB, these instances do not contain the data on the compression aspect for the NDPC problem. Therefore, we extend the data artificially as follows: As for the edge-capacities, we assume that the compressor costs are constant, i.e., we assume that $c_u = 16.67$ for all compressors $u \in V$. This price is given in the "NODES2CAPS" entry in the original data-field, where this cost is specified as the cost for the smallest batch of node capacities, a feature which is not used for the NDPC, respectively the NDP problem. In Subsection 2.6.2 and Subsection 2.6.3, we assume that $\gamma^q = 2$ for all commodities $q \in Q$, if not stated otherwise. When considering robustness for the NDPC problem as in Subsection 2.6.4, we determine $\lambda^q = \frac{1}{\gamma^q}$ at random. We describe this process in detail in the corresponding subsection.

We point out that all data used in this section, for the deterministic as well as for the robust case, is available for download and is briefly described on the website [39].

All computations are carried out on an Intel(R) Core(TM) i7-3770 CPU @ 3.40 GHz with 32 GB RAM. We employ the state-of-the-art MILP solver CPLEX 12.6 [45], relying on AMPL [6] as modeling language.

Table 2.5: Comparison between NDPC solutions and the corresponding NDP solutions with a time limit of 3,600 seconds. A dash indicates that the timelimit was reached. The column Relative Cost (Rel.) indicates the improvement in the objective cost of the NDPC solution in comparison to the NDP solution.

	Instance	LP Bound		Rel.	Time		Gap	
		NDPC	NDP	Cost	NDPC	NDP	NDPC	NDP
peak traffic	ABILENE16	531.12	722.26	0.21	7.61	0.13	0.00	0.00
	ABILENE8	752.32	1125.56	0.32	13.80	0.19	0.00	0.00
	GEANT13	738.91	969.70	0.18	-	752.46	0.12	0.00
	GEANT4	713.23	900.45	0.13	-	977.02	0.14	0.00
	GERMANY17	799.93	1156.90	0.26	1050.02	3.59	0.00	0.00
	GERMANY50	1093.55	1502.92	0.10	-	-	0.48	0.26
average traffic	ABILENE16	357.04	421.67	0.14	13.20	0.17	0.00	0.00
	ABILENE8	352.43	424.83	0.17	5.75	0.24	0.00	0.00
	GEANT13	450.56	492.93	0.05	-	-	0.24	0.09
	GEANT4	466.06	521.12	0.06	-	1140.77	0.21	0.00
	GERMANY17	462.70	557.34	0.12	-	5.52	0.06	0.00
	GERMANY50	611.58	722.21	-0.47	-	-	0.76	0.43

2.6.2 Comparing network design and network design with compression

Solution times and objective values In this subsection, we evaluate the difference between the NDPC problem and the NDP problem. Therefore, for a given NDPC instance as described above, we consider the corresponding NDP_0 instance as described by Remark 2.3 and as originally given by Koster et al. [85]. That is, we consider problem where all compressors are deactivated. We refer to Table 2.5 for a straight forward comparison between both models. The column "Rel. Cost" (relative cost) describes the improvement the objective cost of the best integer solutions found for the NDPC problem and for the NDP problem, i.e., the NDPC problem requires a fraction of the price induced by the NDP problem. We further show the total time required to solve an instance ("Time"). Since a time limit of 3600 seconds was imposed, the column "Gap" denotes the relative optimality gap if an instance was not solved to optimality within the time limit. For later reference, we also present the solution value ("LP Bound") of the linear relaxation of each instance.

We point out some observations. As we can see in the "Rel. Cost" column, the NDPC problem offers much cheaper solutions than the corresponding NDP problem. This is expected since any solution to the NDP problem is also a feasible solution for the NDPC problem, i.e., the NDPC problem offers more flexibility and therefore, it has a larger solution space. Excluding the very last row, the improvement in the objective costs ranges between 5%-32%, indicating the (economical) importance of NDPC.

For the GERMANY50 instance, not even the root node can be solved for NDPC (within the time limit). Thus, no "proper" primal/dual bounds can be obtained and hence, the outlier of an -47% improvement can be explained by a too low solution quality.

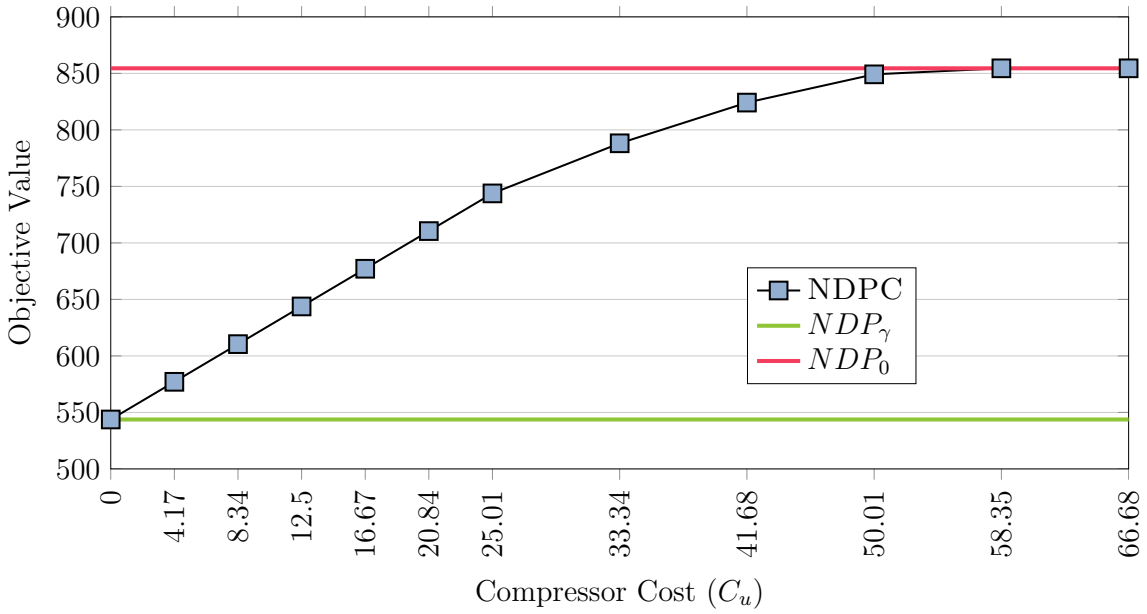
Indeed, both problems are difficult in the sense, that even "small" instances, e.g., GEANT13, cannot be solved to optimality in all cases and that the "bigger" instances (GERMANY50) are completely out of scope. Nevertheless, our results indicate that the NDPC problem is significantly more difficult than the NDP problem, as the former hits the time limit in seven out of twelve cases whereas the latter does so in only three out of the twelve cases. The optimality gaps imply the same message. For more insights into this, we refer to Section 2.4. Furthermore, it seems that the instances with *average* traffic load take (on average) more time to solve than the *peak* traffic instances.

In the following, we further analyze the impact of the compression aspect of NDPC.

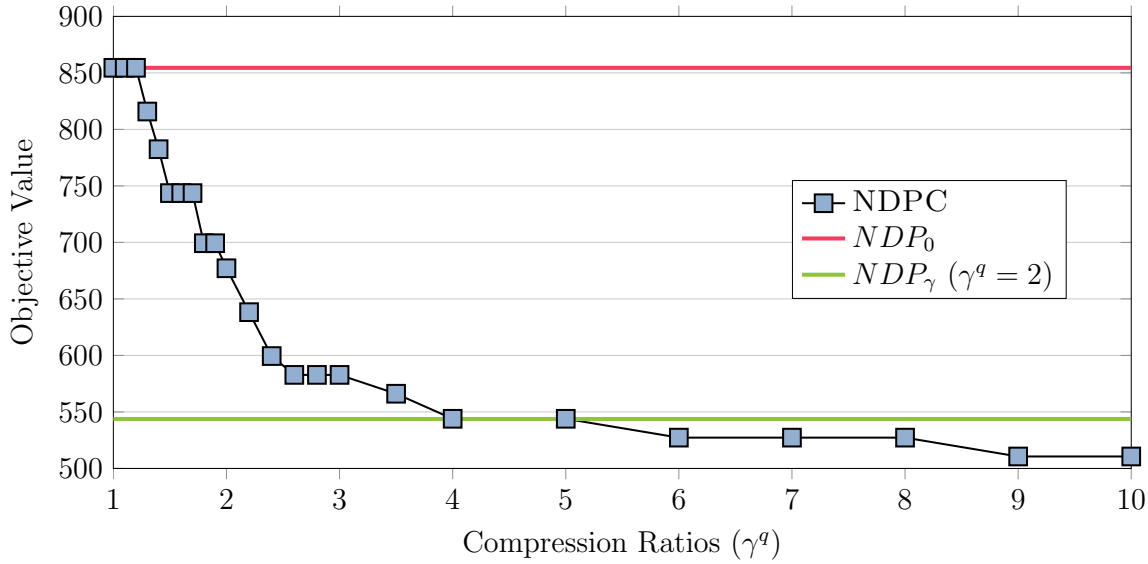
Compression cost and compression factors As we have observed in the previous paragraph, the NDPC problem can generally offer a cheaper solution than the corresponding NDP_0 problem. We visualize the "convergence" from NDPC to NDP if either $c_u \to \infty$ for all $u \in V$ or, if $\gamma^q \to \infty$ for all commodities $q \in Q$, in the following figures. Throughout this paragraph, we focus on the ABILENE16 instance with *peak* traffic.

Consider Figure 2.19 (a), where we visualize the impact of the compressor cost on the solution value for the NDPC problem. We start with $c_u = 0$ for all compressors $u \in V$ and subsequently increase this cost to 66.68. We indicate the cost of the optimal solution (y-Axis) in relation to the compressor cost (x-Axis) in blue. Recall that $\gamma^q = 2$ for all $q \in Q$. By construction, the resulting graph is monotone increasing. For $c_u = 0$, the optimal solution value is equal to the optimal solution value of the corresponding NDP problem where all compressors are turned on (for free), compare Remark 2.3. The solution value of the corresponding NDP_γ problem is shown in green. For $c_u \geq 58.345$, the optimal solution value of NDPC is equivalent to the solution value of NDP_0 (indicated in red), i.e., the compressors are too expensive to be used at all.

Now, let us consider the same instance ($c_u = 16.67$ for all compressors $u \in V$) but let us vary the compression ratios γ^q for all commodities $q \in Q$. That is, we start with $\gamma^q = 1$ and subsequently increase these ratios to (the artificially high value of) $\gamma^q = 10$. This way, the resulting graph is monotone decreasing, compare Figure 2.19 (b). By construction, for $\gamma^q = 1$, the optimal solution of the NDPC problem is equivalent to the optimal solution of the NDP_0 problem. In this particular instance, this even holds as soon as $\gamma^q \leq 1.2$ indicating that with such low compression rates no sufficient gain in the objective can be obtained by compressing the data streams. Note that with $\gamma^q \geq 9$, each edge-module is installed at most once, such that additional compression cannot improve the objective value any further. In this case, the solution is even cheaper than the one of NDP_γ problem (for $\gamma^q = 2$) as less edge-modules respectively compressors are required.



(a) Optimal solution values for different compressor costs (blue). The corresponding solutions for NDP_γ (NDP_0) are depicted in green (red). It is $\gamma^q = 2$ for all $q \in Q$.



(b) Optimal solution values for different compression ratios (blue). The solution for NDP_0 is colored red. For comparison, the solution of NDPC$_\gamma$ ($\gamma^q = 2$) is indicated in green, compare Sub-figure a).

Figure 2.19: Optimal solution values for the ABILENE16 instance. In a) for different compressor costs and in b) for different compression ratios.

Conclusion From a practical point of view, NDPC offers great potential for efficient (with respect to objective cost) network design if the data is eligible, i.e., if the compression ratios are high and the compressor costs are low enough. Since the NDPC problem encompasses the possibility not to employ any compression at all, it is advisable to employ this more general model when concerned with such optimization problem, even if the above described situation is not met.

The drawback encountered when dealing with the NDPC problem in comparison to the NDP problem is the increased computational effort required. The NDPC problem takes substantially more time than its NDP counterpart as was also indicated by our study on the computational complexity in Section 2.4. However, we believe that the economical benefits outweighs this drawback and we advise to prefer the NDPC problem over its predecessor, when applicable.

2.6.3 Cutset and extended cutset Inequalities

In this subsection, we analyze the impact of the cutting planes as presented in Section 2.3 when tackling the NDPC problem with mixed integer linear programming. That is, we consider the Cutset and the Extended Cutset Inequalities for the NDPC problem.

We split the subsection into multiple paragraphs. In the first paragraph, we focus on the improvement of the LP bound, achieved when separating these inequalities at the root node of the corresponding MILP formulation for NDPC. In the second paragraph, we analyze the impact of the Cutset and Extended Cutset Inequalities, depending on the size of the considered cut. In the third paragraph, we describe a heuristic way to exploit such inequalities, based on the "shrinking" Theorem 2.5, yielding an algorithm tailored for bigger instances as, for example, the GERMANY50 instance. We conclude by summarizing our results in the last paragraph.

Direct separation via MILP At first, we present a straight forward separation approach indicating the improvement in the tightness of the formulation when Cutset and Extended Cutset Inequalities are added to the LP relaxation. Therefore, based on the LP solution, we separate violated inequalities in the root node via the separation MILP as presented in Subsection 2.3.6. Note that the separation is not exact since we employ a time limit of 300 seconds for the separation problem at each iteration. When no further violated cut can be found (within the time limit), we solve the resulting formulation for the NDPC problem (to integer optimality), respecting a time limit of $3,600$ seconds for this final solution. The complete results can be found in Table 2.6 for the *peak* traffic case, and in Table 2.7 for the *average* traffic case.

In the tables, "TP" denotes the type of the inequalities separated. In this column, we denote "std" for Cutset Inequalities (only), "ext" for extended Cutset Inequalities (only) and "both" if both families are separated, i.e., at first it is checked whether a violated Cutset Inequality is found and if none can be found, the separation for the Extended

Table 2.6: Bounds and solution values when separating Cutset Inequalities (std), Extended Cutset Inequalities (ext) or both families of inequalities (both) for *peak* traffic volumes. The column "TP" denotes which inequalities are separated. The column "LP" denotes the LP bound after separating all violated inequalities, "#Cuts" indicates the number of iterations, and "T. Sep." denotes the total time spent for solving the separation problem. The column "Best Sol." gives the best solution found within the time limit, "T. Opt." states the time needed to solve an instance to optimality, and "RMG" refers to the final relative MILP gap when the time limit was reached.

Instance	**TP**	**LP**	**#Cuts**	**T. Sep.**	**T. Opt.**	**Best Sol.**	**RMG.**
ABILENE16	no cuts	531.12	-	-	7.61	677.12	1.00
	std	608.82	12	1.69	7.36	677.12	1.00
	ext	587.31	8	2.05	8.25	677.12	1.00
	both	611.23	16	3.23	8.50	677.12	1.00
ABILENE8	no cuts	725.32	-	-	13.80	876.82	1.00
	std	815.97	13	1.38	10.20	876.82	1.00
	ext	773.28	6	1.22	12.48	876.82	1.00
	both	816.31	14	1.86	8.80	876.82	1.00
GEANT13	no cuts	738.91	-	-	3,600	1121.03	0.12
	std	889.70	27	35.74	3,600	1137.70	0.11
	ext	884.14	24	131.97	3,600	1121.03	0.08
	both	900.76	51	141.77	3,600	1121.03	0.08
GEANT4	no cuts	713.23	-	-	3,600	1076.69	0.15
	std	842.79	25	31.31	3,600	1082.19	0.13
	ext	818.84	26	138.34	3,600	1076.69	0.13
	both	855.80	40	133.15	3,600	1082.19	0.12
GERMANY17	no cuts	799.93	-	-	1050.02	1060.02	1.00
	std	934.47	25	9.29	476.20	1060.02	1.00
	ext	899.20	18	27.46	766.95	1060.02	1.00
	both	952.84	26	24.92	322.26	1060.02	1.00
GERMANY50	no cuts	1093.55	-	-	3,600	2363.91	0.47
	std	1472.38	55	11943.00	3,600	3118.54	0.50
	ext	1093.55	1	300.00	3,600	2325.07	0.47
	both	1472.44	61	14125.00	3,600	3189.72	0.51

Cutset Inequalities is run. In order to compare to the standard case, we denote with "no cuts" the case that no inequalities are separated at all. In the next column, we denote with "LP" the LP bound after separating the corresponding inequalities. In this context, the column "#Cuts" refers to the number of separation routines which where run. This number indicates that "#Cuts-1" cuts where added to the problem, since in the very last iteration of the separation process, no violated inequality was found any more. The column "T. Sep." states the total time spent in the separation MILP. Concluding the description of the tables, the column "T. Opt." refers to the time needed to solve a problem instance, including the time of the separation. "Best Sol." gives the best solution found so far and "RMG" refers to the relative MILP gap if no optimality could be proven for some solution.

At first, we comment on the *peak* traffic case, see Table 2.6. For any of the six instances, the separation of any type of inequalities did not have any effect on its solvability within the time limit. With or without cuts, the same instances could be solved to optimality. In the following, we will exclude the GERMANY50 instance from the analysis and postpone the discussion of this case to the end of this paragraph.

As we can observe in the "LP" column, the additional inequalities have a significant influence on the derived LP bounds. In detail, when separating both families of inequalities, we obtain an average bound improvement of 18% (a minimal improvement of 13% and a maximal improvement of 22%) over the "plain" LP bound without any solver generated cuts. We can see that an improvement of similar magnitude can already be obtained by only separating the Cutset Inequalities (std). With these inequalities (std), we obtain an average improvement of 17%. Separating the Extended Cutset Inequalities alone gives an improvement of only 13%. This trend is visualized in Figure 2.20. It shows the "gap closed" (GC, see below) in comparison to the plain LP solution, when separating each family of inequalities. The gap closed is calculated as follows: let DB denote the (plain) LP relaxation of an instance, DB_s the dual bound obtained after separation and PB the best primal bound available. Then, we define the *gap closed* as

$$GC := \frac{DB_s - DB}{PB - DB}. \tag{2.128}$$

As we can observe in Table 2.6, most of the gap closed is due to the standard Cutset Inequalities. However, with respect to GC, in all cases, the impact of the separation process is quite significant.

The improvement in the LP bound (respectively, in "gap closed") is obtained by separating between 13 and 50 violated inequalities (when separating both families of inequalities). We point out that, this number seems to correlate to the "difficulty" of an instance, i.e., the more time it takes to solve an instance to optimality, the more inequalities are separated. In general, the time investment is significant. Nevertheless, in all cases, "T. Sep." is lower than 300 seconds, indicating that the time limit for the separation problem was never reached, i.e., that the separation was indeed exact. Surprisingly, this improvement leads to a faster overall solution process, even though we are separating the inequalities exactly via a MILP. Consider, for example, the GEANT13

Table 2.7: Bounds and solution values when separating Cutset Inequalities (std), Extended Cutset Inequalities (ext) or both families of inequalities (both) for *average* traffic volumes. The column "TP" denotes which inequalities are separated. The column "LP" denotes the LP bound after separating all violated inequalities, "#Cuts" indicates the number of iterations, and "T. Sep." denotes the total time spent for solving the separation problem. The column "Best Sol." gives the best solution found within the time limit, "T. Opt." states the time needed to solve an instance to optimality, and "RMG" refers to the final relative MILP gap when the time limit was reached.

Instance	**TP**	**LP**	**#Cuts**	**T. Sep.**	**T. Opt.**	**Best Sol.**	**RMG.**
ABILENE16	no cuts	357.04	-	-	13.20	532.76	1.00
	std	460.21	11	1.29	9.70	532.76	1.00
	ext	413.47	9	2.08	8.84	532.76	1.00
	both	474.29	26	5.36	13.10	532.76	1.00
ABILENE8	no cuts	352.43	-	-	5.75	516.09	1.00
	std	456.01	12	1.49	8.98	516.09	1.00
	ext	413.18	9	1.73	8.35	516.09	1.00
	both	463.77	22	4.25	9.33	516.09	1.00
GEANT13	no cuts	450.56	-	-	3,600	954.50	0.24
	std	644.93	27	17.01	3,600	954.50	0.22
	ext	660.51	24	170.74	3,600	954.50	0.22
	both	661.31	51	169.06	3,600	937.83	0.18
GEANT4	no cuts	466.06	-	-	3,600	915.66	0.21
	std	644.71	26	29.22	3,600	915.66	0.17
	ext	639.01	35	180.86	3,600	932.33	0.18
	both	656.19	44	135.30	3,600	915.66	0.17
GERMANY17	no cuts	462.70	-	-	3,600	782.47	0.06
	std	604.49	27	9.38	3,600	793.46	0.05
	ext	580.66	19	19.46	3,600	782.47	0.02
	both	608.57	32	16.23	252.55	782.47	-
GERMANY50	no cuts	611.58	-	-	3,600	3085.20	0.75
	std	1016.22	42	6723	3,600	2891.17	0.61
	ext	611.58	1	300	3,600	3190.72	0.77
	both	1016.22	29	3630	3,600	5149.73	0.79

Figure 2.20: Gap closed (GC) in comparison to the linear relaxation of all instances when separating Cutset Inequalities, Extended Cutset Inequalities or both families of inequalities. Restricted to the instances with *peak* traffic load.

instance, which can be solved in less than 33% of the time or, in the case that the instances could not be solved to optimality, the relative MILP gap when the time limit was reached is (always) improved. The only exception to this is the ABILENE16 instance, where the solution time increases from 7.61 seconds to 8.50 seconds. It seems that this particular instance is too "easy", i.e., the time investment in the LP bound does not pay back in time.

We briefly comment on the *average* traffic load case, see Table 2.7. In general, the same trend as for the *peak* traffic is observed, even though less instances can be solved to optimality. However, a substantial LP bound improvement allows to decrease the relative final MILP gaps. In particular, this means that the instance GERMANY17, which cannot be solved to optimality without any cuts (or with separating only one of the two families of inequalities) can be solved to optimality in only 252.55 seconds, when separating both families of inequalities. We point out that, opposed to the *peak* traffic case, the Extended Cutset Inequalities seem to be stronger in relation to the standard Cutset Inequalities. Still, in many cases, the bounds achieved by separating either of the two families are of comparable magnitude.

In general, we believe that the LP bound improvement as well as the improvement in the run time, even with respect to an exact separation approach, indicates the importance of the (Extended) Cutset Inequalities for NDPC.

Let us conclude this paragraph with some insights into the GERMANY50 instance. As in-

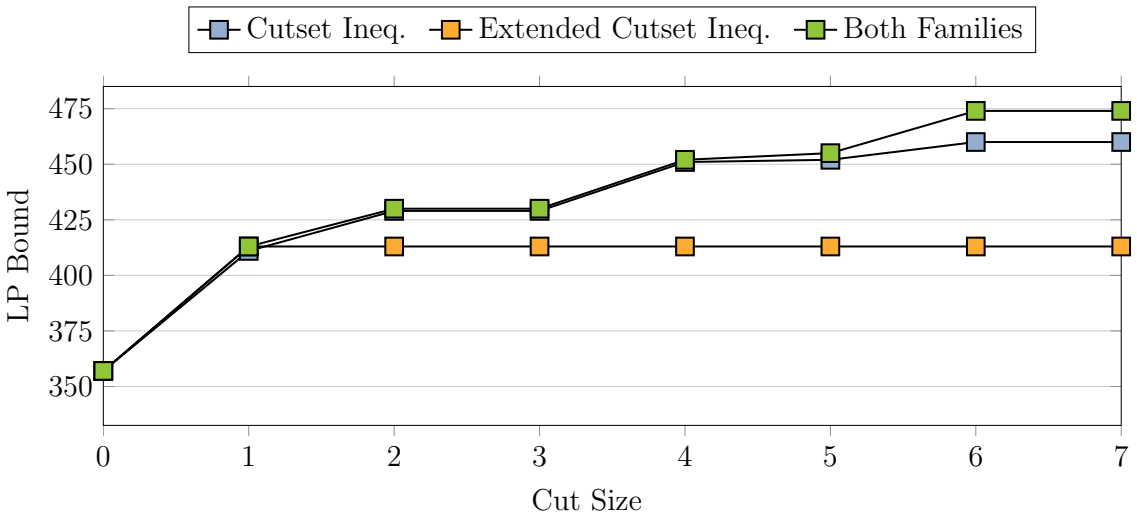


Figure 2.21: The ABILENE16 instance with *average* traffic load. The figure shows the LP bound (y-axis) when introducing all standard Cutset Inequalities (blue), Extended Cutset Inequalities (red), or both families of inequalities (green), corresponding to cuts of a certain node size (x-axis). $x = 0$ refers to no cuts at all, i.e., to the LP bound.

dicated in the preceding paragraph, the GERMANY50 instance is practically out of scope for a MILP approach as not even the root node can be completely solved, even without separating any inequalities. In particular, the LP relaxation is computed completely, but the additional procedures for the root node invoked by the solver (e.g., heuristics, etc.) cannot be finished. However, the solver is able to produce feasible solutions, such that a "good" LP bound is of interest to judge the quality of the obtained solutions. As we can observe in both tables, this bound is dramatically improved by separating Cutset Inequalities (std), albeit at a tremendous time investment of several hours. This is due to the fact, that the resulting separation problem is very difficult, often running into its time limit of 300 seconds. Thus, the separation is not exact for this instance. This trend is even increased for the Extended Cutset Inequalities. In both cases, *peak* and *average* load, only a single iteration of the separation procedure is performed. Thereby, the solver is not able to find a sufficient good solution to the separation problem, such that no violated inequality can be found, and hence, the separation procedure stops.

Even though the improvement obtained by the cutset inequalities approach is promising, the exact separation via MILP seems to be inappropriate. We present a heuristic way of separating the inequalities, especially tailored for bigger instances in the Paragraph "Iterative Cut Generation Based on Graph Shrinking".

Evaluating the cut closure As we have seen in the preceding paragraph, the (Extended) Cutset Inequalities play an important role for the NDPC problem. In the current paragraph, we investigate the effectiveness of the inequalities in conjunction with

the node sizes of their induced cuts. Therefore, we focus on the ABILENE16 instance with *average* traffic and add all (Extended) Cutset Inequalities where the corresponding cut has *at most* size $k \in \mathbb{Z}_+$, i.e., we enumerate all inequalities corresponding to the cuts of a certain size. The results are visualized in Figure 2.21, where we plot the LP bound (y-axis) when adding all inequalities corresponding to a cut of at most size k (x-axis). The blue line refers to the case where only standard Cutset Inequalities are considered, the orange line indicates that only the Extended Cutset Inequalities are separated, and the green line refers to the bound obtained by adding both families of inequalities.

By construction, the functions are monotone increasing as with rising k, additional inequalities are added to the ones already considered for a lower value of k. Interestingly, we observe that for all families of cuts, the largest impact is achieved by adding the inequalities corresponding to single node cuts. When considering bigger sized cuts, the resulting improvement diminishes, even though relatively more inequalities are added (there are $|V|$ inequalities when considering a cut of size 1 but $|V|^2 - |V|$ when considering a cut of size 2, etc.). As in the previous paragraph, we observe a remarkable difference between the Cutset and the Extended Cutset inequalities. While the former offer a steady improvement when considering increasing values of k, the latter show no further benefit when considering $k > 1$. In general, we observe that separating both families of inequalities yields the best bound improvement, followed by separating the standard Cutset Inequalities alone and then by separating the Extended Cutset inequalities alone, independently of the value of k. Nevertheless, for the smaller values of k, it seems that the improvement achieved by separating both families of inequalities yields a similar improvement as gained by separating the standard Cutset Inequalities only. On the contrary, when considering bigger values of k, separating both families yields a superior bound, even though the additionally incorporated Extended Cutset Inequalities show no increasing (improving) trend them-self when considered separately.

Such procedure is also attractive in cases where an (exact) separation approach is too time consuming, i.e., for the GERMANY50 instance. For this instance, a significant bound improvement that is, from 612 units without any cuts to $1,105$ with any family of cuts, can directly be obtained by adding all (precomputed) single node cuts. The expense of about 30 seconds of additional computing time is very minor compared to the more than $3,600$ seconds required in the previously described separation approach (compare Table 2.7). However, for this instance, the improvement stalls for $k = 1$. That is, no bigger values of k need to be considered. In particular for $k = 3$, $20,875$ cuts are added and the time required to solve the LP increases to one minute while no further bound improvement can be achieved. For $k = 4$, the approach becomes intractable as the LP cannot be solved any longer. For more information, we refer to Table 2.8.

As a conclusion of this paragraph, we remark that the diminishing return when considering bigger size cuts can be circumvented by restricting to smaller size cuts. E.g., adding all inequalities corresponding to cuts of size one or two is computationally tractable but gives a significant bound improvement, without requiring to run a complete separation procedure. This is especially beneficial in cases where separation routines are very time

Table 2.8: The GERMANY50 instance with *average* traffic load. The table shows the LP and the solution time (in seconds) bound when introducing all standard Cutset Inequalities (std), Extended Cutset Inequalities (ext), or both families of inequalities (both), corresponding to cuts of a certain node size (k).

TP	**LP Bound** Cut Size				**Time (s)** Cut Size			
	0	1	2	3	0	1	2	3
std	612	1,089	1,105	1,106	4	38	42	69
ext	612	1,089	1,105	1,105	3	36	42	59
both	612	1,089	1,105	1,106	3	31	47	64

consuming.

Iterative cut generation based on graph shrinking In this paragraph, we present a heuristic separation procedure, especially tailored for bigger sized instances. Therefore, we recall that while the (Extended) Cutset Inequalities allow for significant bound improvements, an exact separation procedure fails e.g., for the GERMANY50 instance (compare Table 2.6 and Table 2.7). This heuristic separation is based on Lemma 2.5 and, for a given parameter $k \in \mathbb{Z}_+$ with $k \leq |V|$, relies on iteratively shrinking the graph until $|V| = k$. A detailed description of the algorithm is given in the following. Note that for the decision which edge is to be contracted, we follow the ideas of Raack et al. [106], respectively of Raghavan et al. [107], where similar ideas are evaluated for the Cutset Inequalities in the NDP problem.

As for the exact separation approach, we invoke the procedure after the linear relaxation of a given NDPC instance has been solved. For every edge $uv \in E$, we introduce the edge weight w_{uv} defined as the absolute value of difference between the slack of the corresponding Capacity Constraint (2.2c) and the constraint's dual solution value. We order the edges accordingly (decreasing). Then, we iteratively shrink the edge with the highest weight as described in Definition 2.7 until the shrunken graph has only k nodes left. Note that after an iteration in the shrinking process, the edge weights w'_{uv} in the shrunken graph are adapted as follows: If the edge $\bar{u}\bar{v}$ is contracted into node u', it is

$$w'_{uv} = w_{uv} \qquad \forall \{u, v\} \cap \{\bar{u}, \bar{v}\} = \emptyset, \tag{2.129a}$$

$$w'_{u'v} = \begin{cases} w_{\bar{u}v} & \forall \bar{u}v \in E, \bar{v}v \notin E, \\ w_{\bar{v}v} & \forall u\bar{v} \in E, \bar{u}v \notin E, \\ w_{\bar{u}v} + w_{\bar{v}v} & \forall \bar{u}v \in E, \bar{v}v \in E. \end{cases} \tag{2.129b}$$

Finally, for the resulting k node graph, we enumerate all possible cuts and add the corresponding de-aggregated (Extended) Cutset Inequalities to the original problem.

Algorithm 2.1 Iterating graph shrinking and enumeration of cuts: Enumerate all cuts in a $k \in \mathbb{Z}_+$ node contracted graph, de-aggregate them and add them to the LP relaxation. Resolve the LP relaxation.

Require: NDPC on $G = (V, E)$, Linear relaxation of NDPC, $k \in \mathbb{Z}_+$

while true **do**
  Solve linear relaxation
  if LP bound does **not** improve **then**
    break
  end if
  $G'(V', E') \leftarrow G(V, E)$
  Initialize w_{uv}
  while $|V|' > k$ **do**
    $\overline{uv} = \arg\max_{uv \in E'} \{w_{uv}\}$
    $G'(V', E') \leftarrow$ Contract uv in $G'(V', E')$
    Adapt w_{uv}
  end while
  Enumerate all cuts in $G'(V', E')$
  De-aggregate all cuts and add them to the linear relaxation
end while

Then, we re-optimize the LP relaxation. In case that the LP relaxation improved, we iterate this procedure. A sketch of this algorithm can be found in Algorithm 2.1.

Apparently, the algorithm is heavily dependent on the choice of k. For $k = 2$, at most three inequalities are added as the shrunken graph consists of two nodes and a single edge, such that one Cutset and two Extended Cutset Inequalities can be found. For $k = 7$, the shrunken graph admits $2^7 - 1$ cuts, such that 381 inequalities are added. This way it is clear that only smaller sizes of k can be considered as computationally tractable. We remark that opposed to the exact separation, there is no guarantee that any violated cut is found (if such exists). We present computational results for the GERMANY50 instance for $k = 2, \ldots, 7$ in Table 2.9. In this table, the columns associated to a) refer to *average* traffic and the columns associated to b) refer to *peak* traffic. The column "k" describes the number of nodes the graph is shrunken to. For the shrunken graph, all cuts of the type "TP" are enumerated and then added to the original problem. The column "Bd. Ipr." describes the bound improvement relative to the linear relaxation which could be achieved after "#It." iterations with a total time investment of "Time".

We point out some observations. For both cases, *peak* and *average* traffic, the best improvement (68%, respectively 35%) is achieved for $k = 7$ when adding both types of inequalities. On the opposite, for $k = 2$, respectively $k = 3$ only a small improvement can be obtained when separating a single family of inequalities. In general, it seems that if the algorithm is able to perform more than two iterations, a significant progress in the bound can be achieved, which, in general, favors higher values of k. In this context,

the algorithm is more successful in the sense that more iterations are performed in the average traffic case, which, in the end, leads to a higher bound improvement compared to the peak traffic case.

In general, the algorithm is more successful when considering a higher number of k since, for a higher value of k significantly more cuts are considered. Note that the time consumption also increases with rising k. However, each iteration (even for $k = 7$) takes less than 25 seconds. This time is due to re-optimizing the LP solution of the original problem. All in all, the time consumption is significantly lower than for the exact approach, for example for $k = 7$, 117 instead of $14,000$ seconds of computing time are required, so that the overall time frame stays much more "manageable". We remark that, especially for the *peak* traffic case, the Extended Cutset Inequalities have a much stronger impact than the Cutset Inequalities, a trend which is different to the one observed during the exact separation.

To conclude, it seems that this way of improving the LP bound is computationally very attractive for bigger sized instances. A substantial part of the potential of the (Extended) Cutset Inequalities can be exploited at a comparatively low time investment when considering adequate values for k, e.g., $k = 5$ or $k = 7$. Clearly, this approach is scalable in the sense that for even bigger instances, either k can be lowered, say to $k = 3$ or $k = 4$ or, at the expense of additional computing time, higher values of k, e.g., $k = 10$, can be considered as these still seem to be computationally tractable for the GERMANY50 instance.

Conclusion We present an overall conclusion on the (Extended) Cutset Inequalities. We have seen that these inequalities can substantially improve the dual bound of the NDPC problem. However, it seems that an exact separation is only possible for small to medium sized instances. This drawback can partially be circumvented by enumerating (and adding) all inequalities, corresponding to specific cuts in the network, e.g., to all one-node cuts since these cuts can already lead to a major bound improvement.

Finally, we have presented a parameter dependent heuristic which, iteratively, shrinks the graph of a given instance to a node size according to that parameter. On the shrunken graph, the heuristic enumerates all cuts and adds the corresponding de-aggregated inequalities to the original problem. The heuristics appears to be successful for bigger sized instances where the other approaches struggle.

All in all, the results indicate that the (Extended) Cutset inequalities play a significant role for the NDPC problem and should be incorporated into the solution process when employing a MILP approach for the NDPC problem.

We conclude this subsection by pointing out that in this work, we did not consider any separation scheme especially tailored for a fast execution in practice. This could, for instance be a heuristic algorithm where the underlying cut is chosen greedily or, the separation MILP could be stopped as soon as the first violated cut is found. However, such schemes are surely important for the application and we devise such analysis as one of the directions for future research, see Section 2.7 for additional comments.

Table 2.9: The GERMANY50 instance with a) *average* traffic and b) *peak* traffic. The column "k" describes the number of nodes the graph is shrunken to. For the shrunken graph, all cuts of the type "TP" are enumerated and then added to the original problem. The column "Bound" describes the improved bound which could be achieved after "#It." iterations with a total time investment of "Time". For comparison, the column "LP" shows the value of the linear relaxation without any additional cuts.

		a) average traffic				b) peak traffic			
TP	**k**	**#It.**	**Bound**	**LP**	**Time**	**#It.**	**Bound**	**LP**	**Time**
std		3	663.07	611.58	16.36	3	1,125.17	1,093.55	15.89
ext	2	2	722.15	611.58	10.75	2	1,335.20	1,093.55	13.57
both		2	735.27	611.58	17.24	2	1,335.20	1,093.55	15.07
std		2	663.07	611.58	13.89	2	1,125.17	1,093.55	17.21
ext	3	2	722.15	611.58	17.93	3	1,456.56	1,093.55	77.43
both		3	760.16	611.58	26.68	2	1,446.87	1,093.55	37.84
std		3	690.65	611.58	28.16	2	1,125.17	1,093.55	18.51
ext	4	2	722.15	611.58	21.37	2	1,456.56	1,093.55	37.29
both		5	859.28	611.58	88.62	2	1,456.56	1,093.55	40.26
std		2	690.65	611.58	28.89	2	1,134.04	1,093.55	33.94
ext	5	2	722.15	611.58	21.29	2	1,476.94	1,093.55	55.07
both		4	830.14	611.58	69.86	2	1,476.94	1,093.55	27.69
std		2	690.65	611.58	25.84	2	1,134.04	1,093.55	30.14
ext	6	2	722.15	611.58	43.01	2	1,476.94	1,093.55	46.61
both		5	886.68	611.58	103.42	2	1,476.94	1,093.55	49.70
std		7	904.33	611.58	220.51	2	1,134.19	1093.55	45.14
ext	7	2	722.20	611.58	43.08	2	1,476.94	1,093.55	52.41
both		5	1,031.25	611.58	116.20	2	1,476.94	1,093.55	52.27

2.6.4 Robust network design with compression

In practice, optimization problems are often subject to data uncertainty. For the NDPC problem, we discuss the case where the uncertainty concerns the compression ratios and the demand volumes. This uncertainty can be tackled by the means of robust optimization as described in Section 2.5 and, in particular, by the Γ-robustness and by the two-source robustness. We give a conclusion in the last paragraph.

Uncertain data and protection measurement In this subsection, we assume that for every commodity $q \in Q$, the demand volume d^q and the compression ratio γ^q are uncertain. In detail, we assume that the specific values of these parameters are not known, but we assume that each of them realizes in a given interval. For the demand volume, this interval is given by the nominal traffic volume $\bar{d}^q \in \mathbb{R}_+$ and the deviation $\hat{d}^q \in \mathbb{R}_+$, i.e., we assume that

$$d^q \in \left[\bar{d}^q - \hat{d}^q, \bar{d}^q + \hat{d}^q\right]. \tag{2.130a}$$

We derive the value as the *average* traffic volume as specified in the data above and define the maximum deviation possible as the difference between the *peak* and the *average* demand volume. Note that if the deviation is too big, we truncate the left border of the interval at zero to avoid realizations where the demand volume is negative.

Similar, the compression ratio $\lambda^q := \frac{1}{\gamma^q}$ is determined by the nominal value $\bar{\lambda}^q \in \mathbb{R}_+$ and the deviation $\hat{\lambda}^q \in \mathbb{R}_+$, i.e.,

$$\lambda^q \in \left[\bar{\lambda}^q - \hat{\lambda}^q, \bar{\lambda}^q + \hat{\lambda}^q\right]. \tag{2.131a}$$

For our computational experiments, we draw $\bar{\lambda}^q \in [0.4, 0.6]$ and $\hat{\lambda}^q \in [0.1, 0.35]$ uniformly at random. We remark that, with respect to the next two paragraphs, when applying Γ-robustness, we choose $\Gamma \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 16\}$. As we will see there, sufficient protection can already be obtained by $\Gamma = 16$, such that, independently of the specific instance, higher values of Γ need not to be considered.

For the NDPC problem with data uncertainty, we assume that the specific realizations of these parameters are not known. In the following, we evaluate different methods, respectively models, to produce robust solutions for this problem. The quality of these solutions will be discussed based on two different measures. The first measure is the *objective value* of a solution. In this regard, a cheaper solution is more desirable than a more expensive one. The second measure is more involved. For this measure, we introduce the notion of the *approximated violation probability* (*avp*) of a solution which approximates the probability that a solution violates at least one constraint, i.e., it is infeasible. That is, *avp* is obtained by testing a solution against 1000 realizations of the uncertain data. More precisely, 1000 test scenarios, where the data is sampled uniformly from the intervals as described above, are created for which the solution is tested. Then, *avp* is defined as the number of scenarios, for which the solution was *not* feasible divided

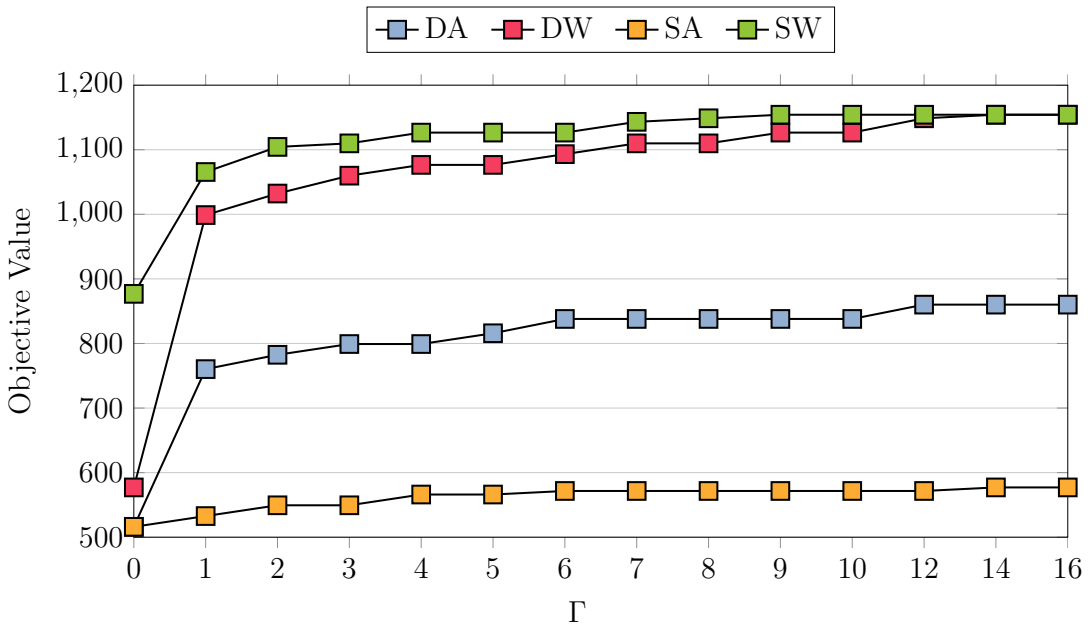
by 1.000. This way, a solution with a lower *avp* value (say: a lower level of *avp*) is preferable to a solution with a higher value (level of *avp*).

In general, there is a trade-off between both measures as favoring one will come at an expense in the other one. The increase in the objective cost when requiring a certain level of *avp* is usually referred to as *price of robustness*, i.e., it is interpreted as the price an optimizer has to pay, if he wants a solution to be more robust against deviating parameters. In general, we interpret a certain quantity in *avp* as a primary (necessary) criterion. That is, depending on the application, we assume that a solution with a too low value of *avp* would induce a complete service disruption in practice. Beyond that, if we can choose among multiple solutions which offer a given level of *avp*, we prefer the one with the lowest objective cost, interpreting the objective value as a secondary criterion.

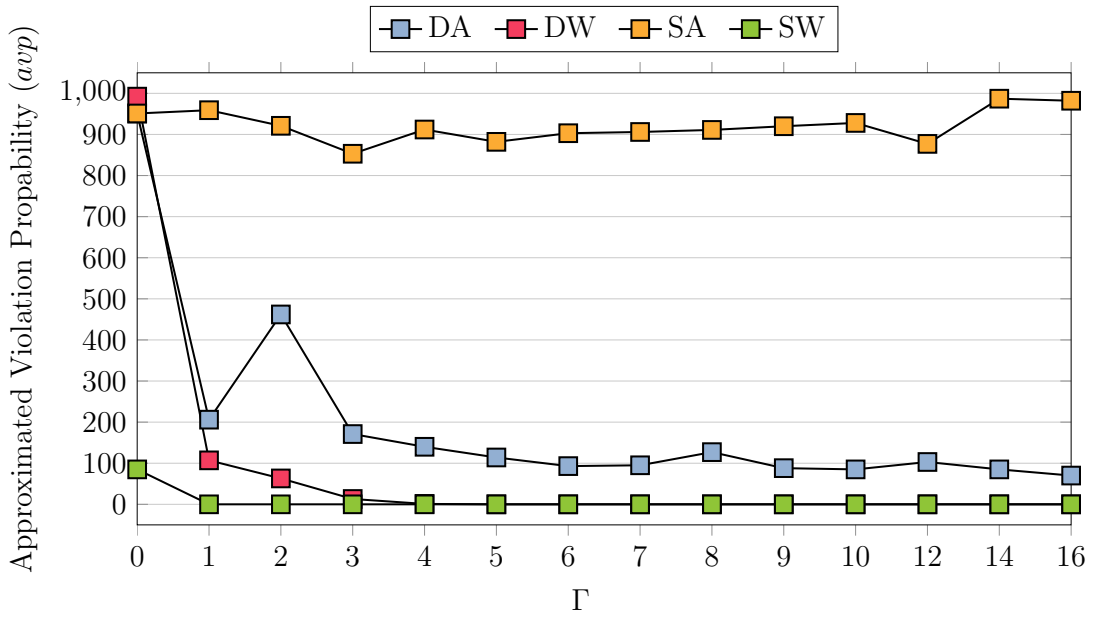
We conclude the paragraph with a short motivation of robust optimization. As discussed in Section 1.4, even with uncertain data, the NDPC problem can be solved as in the deterministic setting by replacing each uncertain value by its average value, i.e., planning with the expected outcome, or by its peak value, i.e., planning with the worst-case. As we can see in Table 2.5, the thus obtained solution values are 516.09 and 1154.20 for the ABILENE8 instance. The corresponding *avp* levels are 0.951 and 0. Hence, the deterministic approaches yield the two *extreme* cases of the above mentioned trade-off: full protection versus no protection at all or, economically speeking, describe massive over- versus massive under-provisioning of network capacities. At this point, robust optimization approaches can offer more and diverse solutions which allow to exploit different and potentially more attractive trade-offs between the two competing figures.

Γ-robustness i) We apply Γ-robustness to the NDPC problem with data uncertainty as described above. We focus on the Γ-robust models as introduced in Subsection 2.5.1. At first, we consider the case where Γ-robustness is applied to a single uncertain influence at a time and where the other uncertain influence is accounted for as a fixed value. In this case, we say that the parameter the Γ-robustness is applied to is *protected*, respectively it is made *robust*, against random deviations.

We consider four robust problems: DA, DW, SA and SW. In this context, A stands for average and W for worst-case. Similar, D stands for demand, i.e., it refers to the d^q values and S refers to the scaling, i.e., to the λ^q values. In the problems DA (DW), the parameter d^q is protected whereas λ^q is fixed to the nominal value $\bar{\lambda}^q$ (respectively, its peak value $\bar{\lambda}^q + \hat{\lambda}^q$). Vice versa, in the problems SA (SW), the parameter λ^q is protected whereas d^q is fixed to $\bar{d}^q$ ($\bar{d}^q + \hat{d}^q$, respectively). For a more formal description of these problems, we refer to Corollary 2.17 and to Corollary 2.18. We point out that in case $\Gamma = 0$, the DA and the SA problems boil down to the same problem, i.e., to the (deterministic) NDPC problem where we assume nominal data for all uncertain parameters. In the case $\Gamma = \infty$, the DW and the SW problem are equivalent to the (deterministic) NDPC problem where we assume worst case data for all parameters.



(a) Objective values.



(b) Approximated violation probability (*avp*).

Figure 2.22: In (a) optimal solution values, and in (b) the level of protection (*avp*) for the ABILENE8 instance obtained by the Γ-robust models *DA*, *DW*, *SA*, and *SW* depending on the value of Γ.

By construction, the choice of Γ heavily influences the quality of the obtained solutions. In general, it holds that, on the one hand, the higher Γ, the more link capacity is required and the more expensive an optimal solution is. On the other hand, a solution which employs more link capacity has got a higher chance of being feasible for random realizations of the uncertain parameters, i.e., these solutions induce a lower *avp* value.

We visualize the impact of Γ on the solution values of the different problems in Figure 2.22 (a). In this figure, we focus on the ABILENE8 instance and plot the objective value (y-axis) of the optimal solution of the different problems as a function of the value of Γ (x-axis). It is clear that the problems where one parameter is fixed to its worst case realization yield more expensive solutions than the problems where the corresponding parameter is always at its nominal value. This implies, with respect to the objective value, $DW \geq DA$ and $SW \geq SA$. Additionally, we observe that, for fixed Γ, the solution values of the problems obey the ordering $SW \geq DW \geq DA \geq SA$. This indicates that, with respect to the objective value, the uncertainty in the volume of a commodity plays a more important role than one in the compression ratio.

With the exception of the SA problem, we observe that the solution values increase dramatically when Γ increases from zero to one. This increase diminishes for bigger values of Γ. The reason for this can be found in the data since there are usually only a few traffic entries which dominate the data set. As a consequence, as soon as a solution is protected against a few deviations, deviations within the other traffic volumes do also fit into the correspondingly reserved capacity. In any case, the solution values do not increase any further for $\Gamma \geq 14$. This indicates that complete protection can be reached for comparably low values of Γ. We point out that, over the whole dataset, the SA solution is very cheap and nearly constant in Γ, indicating that Γ-robustness has very little effect for this parameter setting.

As mentioned above, a cheap solution is very attractive from an economical point of view, such that the solutions of the SA problem seem to be preferable. However, this measure does not take into account, whether the solutions are expected to be feasible, given that the parameters are uncertain. For this, we consider the measure *avp*. Again, we visualize the impact of different Γ values on this measure at the ABILENE8 instance, see Figure 2.22 (b). In this figure, we observe the opposite behavior as for the objective values: with increasing Γ (x-axis), the solutions get more attractive with respect to *avp* (y-axis), i.e., they get more secure. For fixed Γ, the solutions of the different problems can be ordered according to their *avp* value, namely: $SA \geq DA \geq DW \geq SW$, i.e., in the reverse order as was implied by the objective values. This time, we see that for $\Gamma \geq 4$, the solutions become mostly stationary, such that no further improvement can be obtained with increasing Γ. When focusing on the DA model, it appears that the graph has an outlier at $\Gamma = 2$, respectively at $\Gamma = 1$. We assume that this is due to random effects in testing the 1000 data realizations against the obtained solution, respectively it seems that the routing chosen in that solution adapts very badly to changing traffic demands.

In general, it appears that the SA problem produces solutions which can be expected

Table 2.10: Given a certain *avp*, we report the best solution value (over $\Gamma \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 10, 12, 14, 16\}$) and the robust model it was obtained by. The GERMANY50 instance is omitted since, in most cases, not even a feasible solution can be found.

Instance	**Desired level of avp**							
	25%	20%	15%	10%	5%	2%	1%	0.0%
ABILENE16	582.77 *DA*	582.77 *DA*	582.77 *DA*	582.77 *DA*	599.44 *DA*	621.61 *DA*	660.45 *DA*	677.12 *DW*
ABILENE8	760.30 *DA*	799.14 *DA*	799.14 *DA*	837.98 *DA*	876.82 *SW*	876.82 *SW*	876.82 *SW*	876.82 *SW*
GEANT13	1043.35 *DA*	1082.19 *DA*	1082.19 *DA*	1082.19 *DA*	1098.86 *SW*	1098.86 *SW*	1098.86 *SW*	1098.86 *SW*
GEANT4	1082.19 *SW*	1082.19 *SW*	1082.19 *SW*	1082.19 *SW*	1082.19 *SW*	1082.19 *SW*	1082.19 *SW*	1082.19 *SW*
GERMANY17	987.84 *DA*	987.84 *DA*	1010.01 *DA*	1010.01 *DA*	1060.02 *DA*	1082.19 *SW*	1082.19 *SW*	1082.19 *SW*

to be violated in more than 85% of the test cases. As a consequence, the SA problem is not very promising with respect to the *avp* measure. On the contrary, the solutions of the SW problem offer very low *avp* levels for all values of Γ. However, with respect to the solution values, compare Figure 2.22 (a), the solutions are of low quality. This illustrates the price of having a high level of protection. In this regard, the other problems offer more interesting solutions. As we can see, if a *avp* of 10% is appropriate for the application, the solutions from the DA problem can already be sufficient (depending on the choice of Γ). For lower rates of *avp* one has to consider the DW or the SW problem. Clearly, if multiple problems yield solutions with sufficient low *avp*, the one which yields the lowest objective value is chosen, e.g., if DA yields sufficient *avp*, its objective cost makes it preferable over the other two problems.

We refer to Table 2.10 for an overview over the complete dataset. Note that the GERMANY50 instance is omitted since it cannot be solved within the time limit. In particular, in most cases not even a feasible solution can be found for this instance. See below for additional comments on the solvability of the instances within the time limit. In Table 2.10, for any instance, we show which robust problem variant provides an optimal solution with the lowest cost if a certain *avp*, i.e., $avp \in \{0.25, 0.20, 0.15, 0.10, 0.05, 0.02, 0.01, 0.001, 0.0\}$, is required. That is, each instance is evaluated with $\Gamma = 0, \ldots, 16$ for all four variants. Then, for each value of *avp*, the best solution among all values of Γ and the model which produced it is reported. As for the show-case ABILENE8, it seems that, in general, there are two preferable models: if a relatively low value of *avp* is sufficient, i.e, $avp \geq 0.15$, the DA problem tends to yield the best solutions. If more conservative solutions are

Table 2.11: Objective value, avp, and solution time of the Γ-robust *AC* problem for the ABILENE8 instance depending on Γ.

Γ	0	1	2	3	4	5	6
Obj. Val.	516.09	782.47	982.17	998.84	1015.51	1032.18	1048.85
avp	0.951	0.571	0.333	0.104	0.016	0.000	0.006
Sol. Time.	7.79	27.63	19.50	19.99	27.12	23.96	41.76
Γ	7	8	9	10	12	14	16
Obj. Val	1054.35	1076.52	1093.19	1093.19	1109.86	1109.86	1132.03
avp	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Sol. Time	26.83	53.51	76.85	48.65	78.12	41.73	96.13

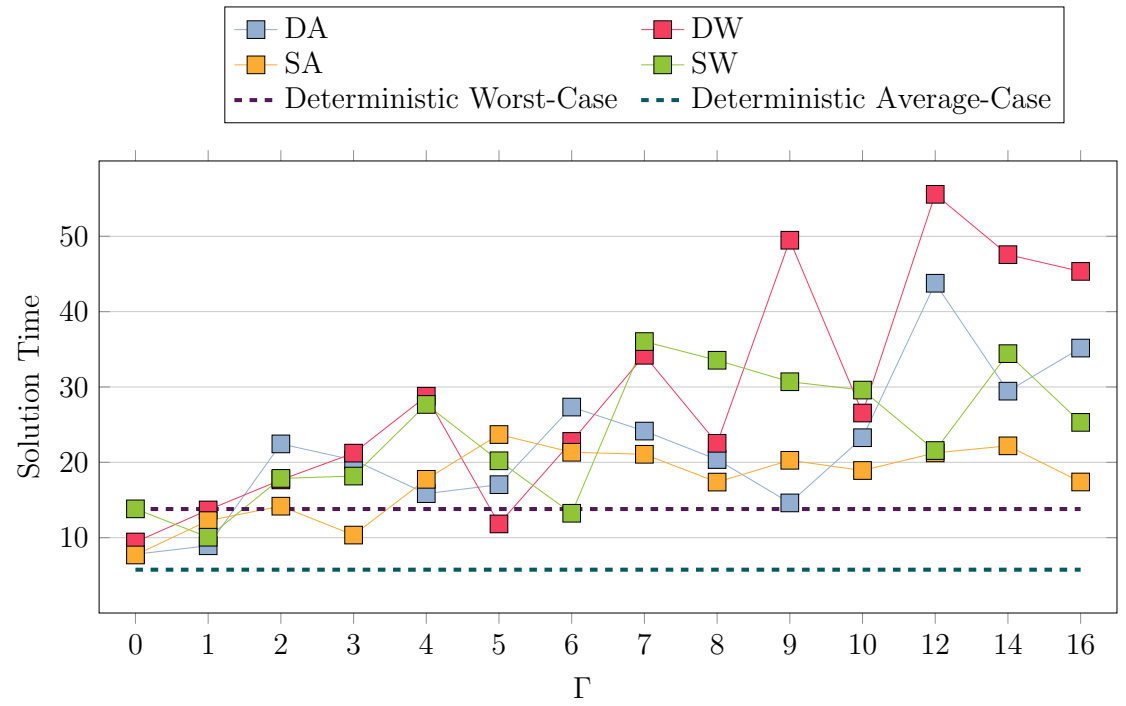
requested, the *SW* model is superior. The sole exception to this trend is the GEANT4 instance where in any case, the *SW* model yields the best solution. In this case, a single solution can be found which offers the best solution value and fulfills *all avp* requirements.

We conclude the paragraph with some comments on the required solution time. In general, it appears that the introduction of the Γ-robustness makes the problem computationally harder, compare Table 2.5 for the deterministic setting, but to a "manageable" extent. That is, the two ABILENE instances can still be solved, in all cases, but require some additional time, see Figure 2.23 (a) for the ABILENE8 instance. The robust GEANT instances can still not be solved to optimality, but the resulting optimality gap is comparable, albeit higher in some cases, see Figure 2.23 (b) for the GEANT4 instance. The GERMANY50 instance is of course out of scope, and the GERMANY17 instance can only be solved up to a small, i.e., less than 10%, optimality gap.

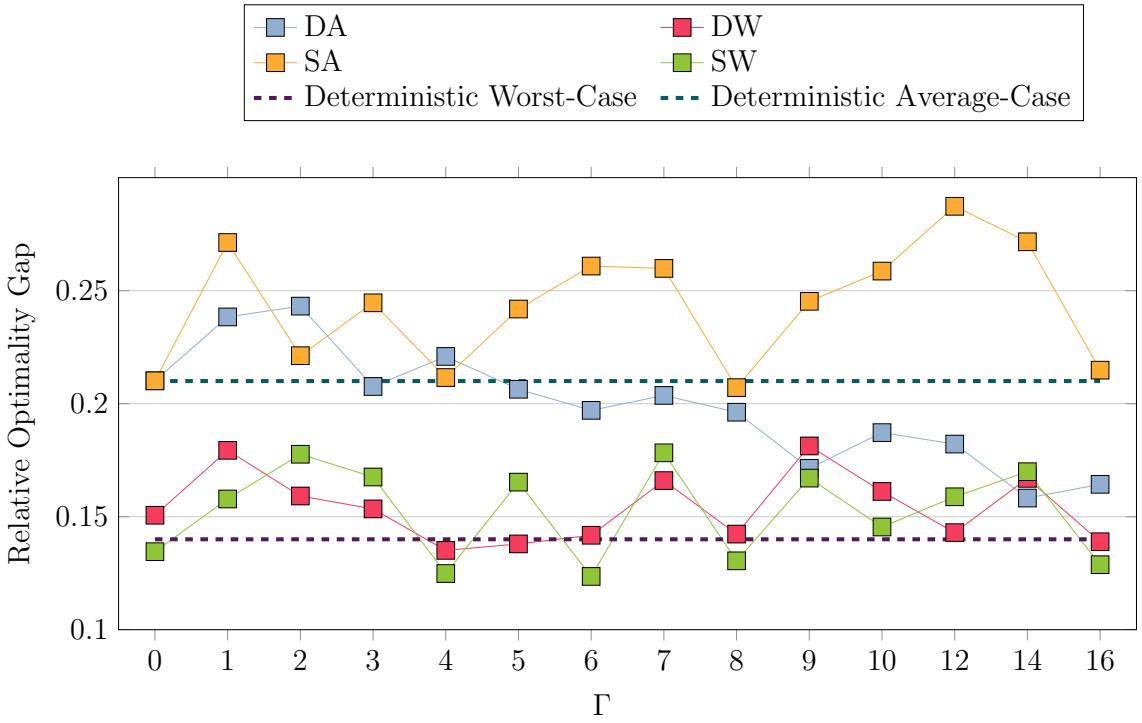
Γ-robustness ii) We apply the Γ-robustness to the NDPC problem with data uncertainty, focusing on the Γ-robust model as introduced in Subsection 2.5.1. Here, we focus on the case where the Γ-robustness is applied to the aggregated uncertain influences, i.e., to the setting where the parameters realize as described in (2.118a) and in (2.118b). The corresponding robust problem is formalized in Corollary 2.16, we abbreviate the problem as *AC* for "aggregated uncertainty".

Let us consider the ABILENE8 instance where we highlight the differences with respect to the other Γ-robust problems *DA*, *DW*, *SA*, and *SW* described in the previous paragraph. The solution characteristics, that is the optimal objective value, the induced level of *avp* and the employed computing time for this instance can be found in Table 2.11.

In general, we observe a similar trade-off between the solution value and the level of *avp* as for the Γ-robust models in the previous paragraph. In detail, low values of Γ lead to solutions with low objective value but low level of *avp*, whereas high values of Γ induce the opposite behavior. However, comparing those results to Table 2.10 where,



(a) Solution times of the ABILENE8 instance.



(b) Relative optimality gaps of the GEANT4 instance.

Figure 2.23: In a) solution times for the ABILENE8 instances, and in (b) relative optimality gaps for the GEANT4 instances of the Γ-robust problems *DA* (blue), *DW* (red), *SA* (yellow), and *SW* (green).

given a certain level of *avp*, we are looking for the best solution with respect to their objective value, it appears that the *AC* model cannot improve over the already obtained solutions. It seems that the *DA* model yields a superior trade-off for all but the most conservative levels of *avp*.

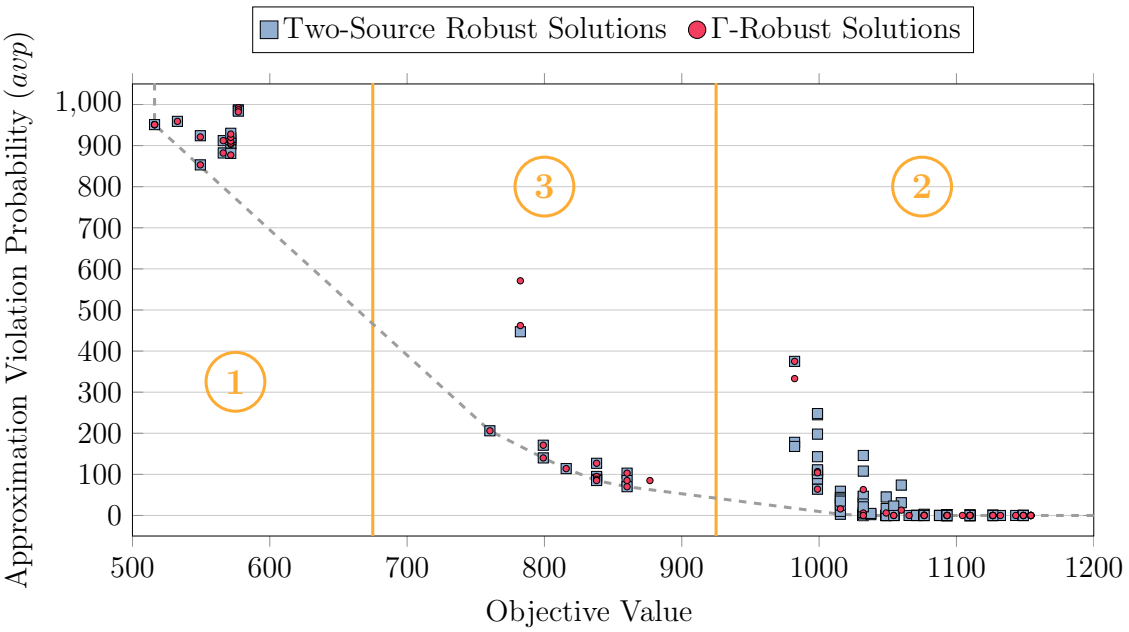
What is more to say is that a similar relation between this and the previous models can be observed in the required solution time. When comparing to Figure 2.23 (a), it appears that the maximal solution time of *AC* (96.13 seconds) is higher than the maximum solution time for any of the other models (55.57 seconds for the *DW* problem). The same also holds for the average solution time, where 42.11 seconds have been required by *AC* in comparison to 29.04 seconds required by the *DW* model (yielding the highest average among all other Γ-robust problems).

Since the relations between the robust problems are similar for the other instances, we conclude this paragraph by pointing out that the *AC* model is, from an application point of view, i.e., with respect to solution value, level of *avp* and time consumption, inferior to the other robust problems.

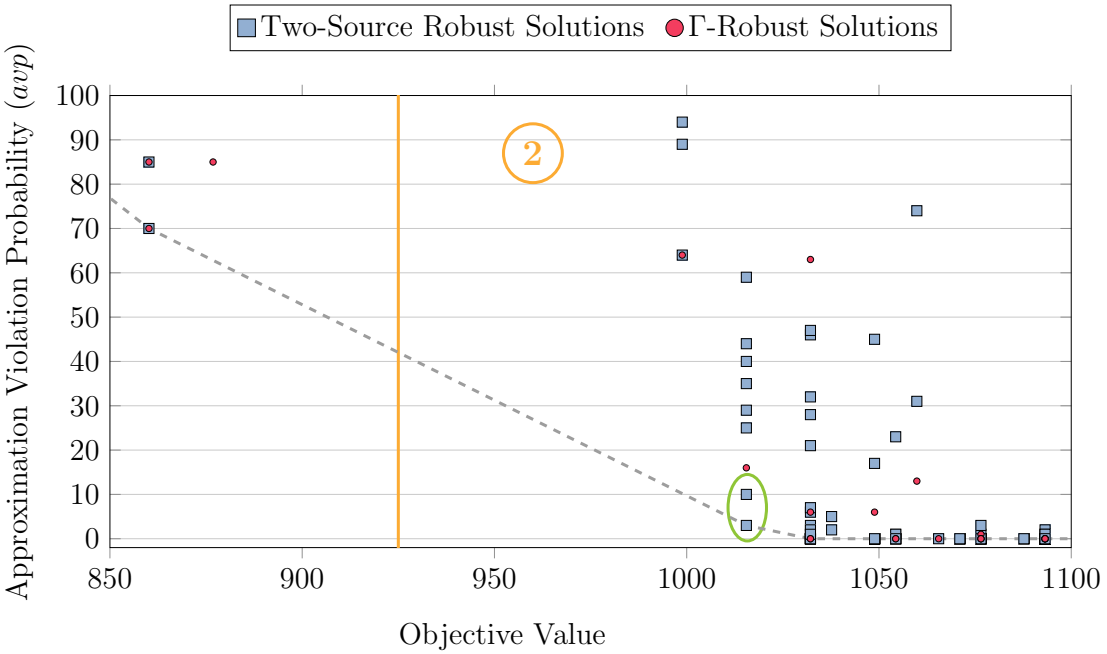
Two-source robustness The Γ-robustness as discussed in the previous paragraphs allows to tackle the NDPC problem under data uncertainty. While Γ-robustness can only tackle a single source of uncertainty, a more appropriate fit for the NDPC problem in this regard is the two-source robustness as presented in Subsection 1.4.4. Similar to the previous paragraphs, let $\Gamma_1, \Gamma_2 \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 16\}$ and consider the robust problem as defined in Corollary 2.19. This way, there are 196 potential parameter settings for each instance. To keep the presentation clear, we restrict ourselves to the show-case of the ABILENE8 instance. Note that in this context, Γ_1 limits the deviation in the demand values, whereas Γ_2 concerns the compression ratios.

At first, we point out that in principle, any robust problem stated in the Paragraph "Γ-Robustness i)" can be identified as a two-source problem if the parameters Γ_1 and Γ_2 are chosen accordingly. I.e., to obtain the Γ-robust *DA*, respectively the *DW* problem, one has to consider the two-source robust problem where $\Gamma_1 = \Gamma$ and $\Gamma_2 = 0$, respectively $\Gamma_1 = \Gamma$ and $\Gamma_2 = \infty$. The setting for the *SA* and the *SW* problems is analogous. Hence, it is clear that the two-source robustness can reproduce all solutions from the Γ-robust problems *DA*, *DW*, *SA*, and *SW*. Again, we observe the same trade-off between the solution value and the level of *avp* as for the Γ-robust models. However, the higher degree of freedom provided by the two-source uncertainty can be used to derive additional solutions, potentially yielding a better trade-off or more desirable characteristics.

For an illustration of the results for the ABILENE8 instances, we refer to Figure 2.24(a). In this figure, we plot the objective value and the *avp* of all robust solutions encountered so far. We mark the solutions obtained by the two-source robust problem in blue and the ones obtained from any other, i.e., Γ-robust, problem in red. At first, we observe that we do not have the strict correspondence between the red and the blue marks as discussed above. This is due to the fact that $\Gamma_i \leq 16$, $i = 1, 2$. Hence, the requirement $\Gamma_2 = \infty$, respectively $\Gamma_1 = \infty$, cannot be met in the experiments. However, if sufficiently large



(a) All robust solutions.



(b) Zoomed to robust solutions with objective values larger than 850 and $avp \leq 0.1$. Relevant solution value – avp combinations not obtainable by the Γ-robust models indicated in green.

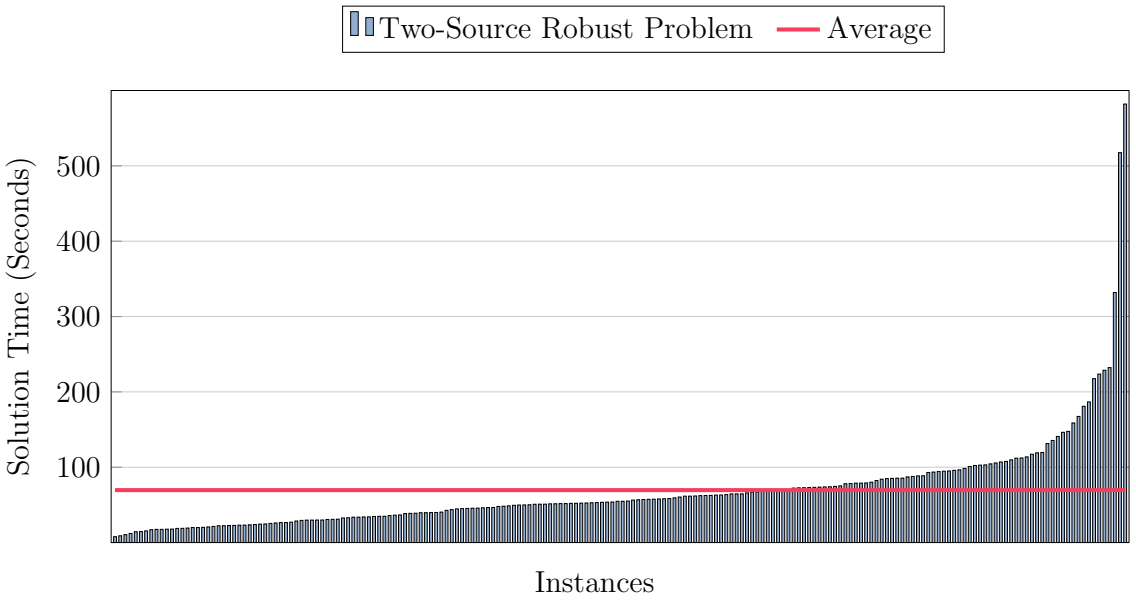
Figure 2.24: Objective values of the ABILENE8 instance obtained for the Γ-robust models (red) and obtained by the Two-Source Robust model (blue) and the corresponding *avp* values. Solutions can be categorized into three groups (orange). In (a), all solutions, in (b), zoomed in on objective values larger than 850 and *avp* lesser than 0.1. In both cases, the dashed lines mark pareto-optimal combinations.

Γ_i-values are considered, e.g., for this instance $\Gamma_i = 20$ is sufficient, indeed all results can be reproduced. Note that the same does not hold for the Γ-robust solutions as obtained in Paragraph "Γ-Robustness ii)".

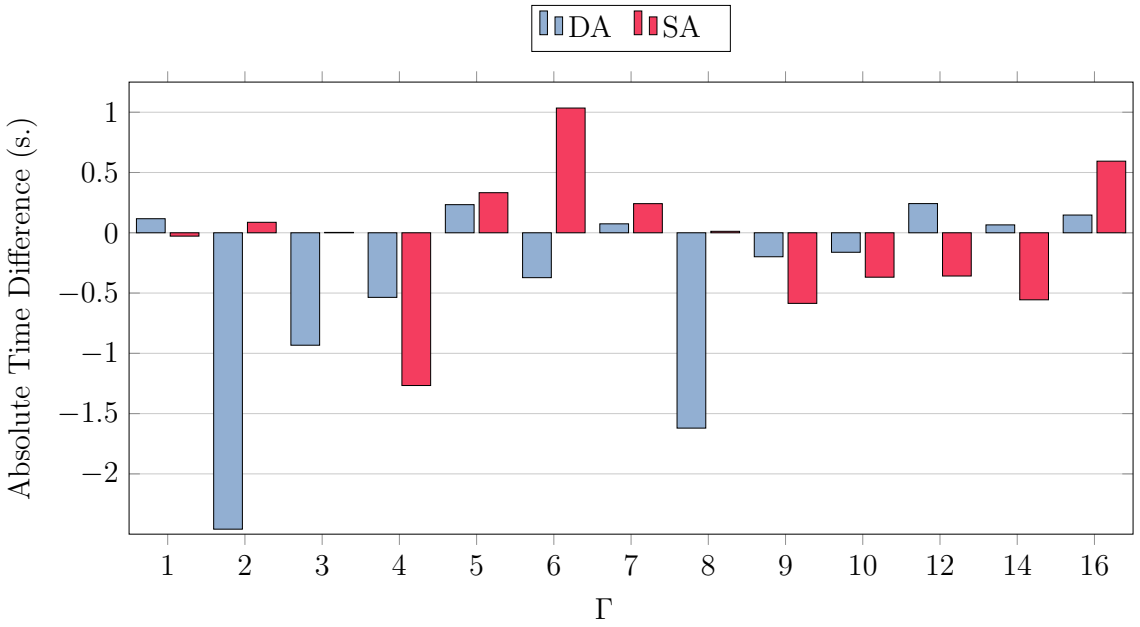
Interestingly, all solutions can be categorized into three groups as indicated in orange: The first one, situated at the top-left consists of solutions with a very low objective value (≤ 600) and a very high level of *avp* (≥ 0.95). Conversely, the second group at the bottom-right is given by solutions with opposite criteria: high objective value (≥ 950) and (mostly) low *avp* levels. The third group of solutions is intermediate with objective value between 700 and 900 and *avp* between 0.2 and 0.95. The relevance for applications is immediately clear. Depending on the desired level of *avp*, one of the categories is favored and among all solutions within such category, the best one, e.g., with respect to the objective value, can then be realized.

For a visualization of the benefit of the two-source robustness, we refer to Figure 2.24(b). In this figure, we provide the same plot as in Figure 2.24(a), zoomed in onto the third category of results: high objective function value (cost) but very low *avp* levels. What we can observe there is that the additional freedom of the two source-robustness yields more and different solutions than the ones obtained by the other robustness concepts. In particular, with respect to solutions with extremely low level of *avp*, there are additional solutions with better objective value than the ones previously known. Consider for example that *avp* should be lower than 0.015. The best Γ-robust solution, with respect to Paragraph "Γ-robustness i)", with that property has an objective value of $1,059.85$ and was obtained by the problem DW. The best Γ-robust solution, with respect to Paragraph "Γ-robustness ii)", with that property has an objective value of $1,032.18$. The best two-source robust solution with that value can improve on that and has an objective value of only $1,015.51$ and is obtained by setting $\Gamma_1 = 1$ and $\Gamma_2 = 4$ ((*avp*) of 0.003) respectively by $\Gamma_1 = 2$ and $\Gamma_2 = 3$ ((*avp*) of 0.01).

We conclude this paragraph by a short evaluation of the computing time required by the two-source robust problems. As it is the case for the Γ-robust problems, only the ABILENE instances can be solved to optimality within the time limit. The GERMANY50 instance remains out of scope and the GEANT instances, respectively the GERMANY17 instance, can only be solved up to a substantial gap. However, on the whole, the required computation time increases, nearly by a factor of two, in comparison to the Γ-robust model. We refer to Figure 2.25 (a) for an illustration of the solution times of the ABILENE8 instance. As we can see, the solution time can go up to 582.04 seconds for single instances with an average of 69.60 seconds per instance. For a comparison, recall that for the same instance, the highest solution time over all Γ-robust problems was 331.89 seconds with an average of 36.31 seconds and that the deterministic problems could both be solved in less than 15 seconds. Finally, see Figure 2.25 (b) for a comparison of the solution times when the two-source robustness mirrors the setting of the Γ-robust models DA and SA, see Paragraph "Γ-robustness i)". In this figure, we plot the difference of the solution times of these models. Note that if this difference is negative, the corresponding two-source robust problem was solved faster. Interestingly, we can observe that, over all instances, the two-source problem seems to be solved faster.



(a) Solution time of the two-source robust problem (sorted in increasing order).



(b) Difference in solution time of the Γ-robust problems DA and SA and the corresponding two-source robust problem. Negative values (downward bars) imply that the two source robust problem was solved faster.

Figure 2.25: In a) solution times for the ABILENE8 instance for the two-source robust problem (196 instances), and in (b) a comparison to the solution times of the Γ-robust problems DA and SA.

Conclusion We present an overall conclusion on data uncertainty and robust optimization as considered here for the NDPC problem. In this subsection, we have evaluated five different alternatives on how the Γ-robustness can be applied to the NDPC problem, and we have also shown how the two-source robustness can be employed. To different degrees, all robust formulations were successful in the sense that all models offer the possibility to obtain a range of solutions, establishing a trade-off between the solution cost and the degree of protection against the uncertainty. In particular, we have seen that the robust problems DA and SW are the most promising ones from this point of view, while the Γ-robust AC problem was observed to yield inferior solutions. By construction, the two-source robust problem can reproduce the solutions of the other robust models DA, DW, SA, and SW, additionally offering a more diverse parameter setting to obtain even more, diverse solutions. All solutions and the there-of induced characteristics heavily depend on the parameter Γ. Thus, from an application point of view, there is a broad range of possibilities available to generate solutions and among all solutions, select the ones with the most desirable characteristics.

With respect to computation time, the price of applying robust optimization to the NDPC problem is that all robust models are more time consuming to solve than their deterministic counterparts. Nevertheless, it appears that all instances which could be solved to optimality within the time-limit can still be solved whereas the other problems terminate at a higher, but still comparable optimality gap. It is worth mentioning that the two-source robust problem is, on the whole, the most time consuming to solve, however, when mirroring the settings of the other robust problems, it is, on average, faster than the Γ-robust models.

All in all, especially the two-source robust problem is a tool which allows to tackle the NDPC problem with data uncertainty. It allows for a fine-grained control over the desired level of protection, respectively over the desired level of conservatism of a solution. We advise to employ this model when data uncertainty occurs in an application.

2.7 Conclusion and outlook

We conclude this chapter by a brief recapitulation of our results and a discussion of open questions and further research directions.

In this chapter, we have extensively studied the NDPC problem. Introduced as an extension of the NDP problem, we started our work by a discussion of the importance of NDPC to real world applications and have then focused on an MILP formulation of the problem. In the following, we have investigated the properties of that formulation, in particular with respect to its underlying polyhedron and have shown that many of the results of the NDP problem can be extended and adapted to the NDPC problem. We have then continued by analyzing the computational complexity of NDPC, highlighting differences and similarities to the NDP problem. Finally, we have considered the case of data uncertainty and have shown how robust optimization concepts can be applied

to NDPC. In the last section, we have provided computational experiments to evaluate our results: we have shown how the NDPC problem practically relates to the NDP problem, how cutting planes can be used to improve the solution process, and finally how data uncertainty can be incorporated into "robust" problem formulations to obtain solutions resilient to parameter deviations.

Regarding future research, we are convinced that the NDPC problem will remain of interest, both from a practical and from a theoretical, research driven perspective. Based on the here presented MILP formulation for the NDPC problem, we have seen that polyhedral results can be successfully employed in the solution process. Therefore, a future research direction can be to extend on these results. Similar to the findings of Agarwal [1, 2], deriving a complete description of the convex hull for three and four node problems can be a further step in this direction. In this context, we especially mention the three-node path problems as a class of basic instances, for which results of NDP are significantly different to the NDPC case and for which a complete description of the underlying polyhedron could not be derived within this work. For further approaches in this direction, we refer to the master thesis of Hütten [76] which focuses on cut based inequalities for the Compressor Placement Problem, results from which could be transfered back to the NDPC problem.

Such research on the theoretical side of the NDPC problem should be joined with increasing efforts on the practical side as well. In this thesis, we did not consider any heuristic solution algorithms for the NDPC problem, even though these are surely important for the application, especially if a fast solution process is required. Clearly, fast, heuristic algorithms are also helpful when considering bigger problem instances, for which an exact MILP approach scales badly, e.g., as for the Germany50 instance. For this purpose, one could, once more, exploit the affinity of NDPC to the NDP problem and translate and extend results like heuristic- and/or approximation algorithms for the NDP problem to the more general case. For a first reference in this regard, we refer to the works of Goemans et al. [64] and of Gupta et al. [66]. Naturally, such results should also be extended to the robust problem.

To this end, results from both directions can be combined to enhance MILP approaches so to generate better scaling algorithms than the ones presented in this work. In particular, with respect to polyhedral results, we conjecture that, in practice, a MILP approach which relies on heuristic separation routines will work best for the NDPC problem. Based on our results, such routines should include the "static" addition of all (Extended-) Cutset Inequalities corresponding to the single node cuts in the network and should possibly also rely on a "shrinking and enumeration" based separation approach as discussed in the Paragraph "Iterative cut generation based on graph shrinking". Furthermore, the separation of cut based inequalities via (greedy) local search heuristics should be considered as well, as these heuristics have been observed to be very successful for the NDP problem, compare, for instance, the work of Mattia and Poss [95].

Finally, we want to give a short remark on data and test-instances for the NDPC problem. In this work, we relied on modified test instances from the SNDLIB [100]. A similar

source for data is the Internet Topology Zoo [83]. Depending on the application, networks from this source should also be considered in future experiments. In this context, we point out that no real-life data for the compression aspect (i.e., compression ratios and compressor costs) is available yet. Naturally, the availability of such data would greatly benefit the relevance of our computational investigations and boost the interest of both parties, researchers and practitioners into the topic. Therefore, we devise the creation or exhibition of such data, hopefully in close collaboration with the industry, as one of the primary future research goals. As a first step in this direction, the datasets used in this work are made available for the public. They can be download from the website [39]. We hope that this data can be extended and adapted where required and, in general, that is helpful in future research.

CHAPTER 3

Virtual network embedding

In this chapter, we focus on the Virtual Network Embedding Problem (VNE). Similar to the NDPC problem in the previous chapter, VNE can be considered as an extension of the NDP problem as presented in Chapter 1. That is, in contrast to the NDP problem, for the VNE problem, the underlying communication network is fixed but the amount of traffic which has to be routed is subject to a decision: which subset of a given set of virtual networks is to be embedded, and hence routed, on a given physical network. The chapter is structured into six sections.

At first, in Section 3.1, we give an introduction to VNE. We motivate the importance of VNE to the telecommunication industry, especially with respect to large scale telecommunication services such as the Internet. We conclude the introduction by describing the problem in detail and reviewing the literature on the topic.

In Section 3.2, we formalize the VNE problem, encapsulating it in a formal definition. This definition is then used to derive a formulation as mixed integer linear program. We discuss how this formulation can be expanded to include rent-at-bulks aspects and demonstrate how such formulation naturally allows for heuristic approaches to VNE.

In Section 3.3, we focus on the theoretical difficulty of VNE. We present results on the complexity and the approximability of the general problem, also discussing the complexity of related heuristic approaches. In the following, we consider special cases of the VNE problem where we fix input dimensions of the problem, observing that many of its subproblems are already $\mathcal{NP}$-hard. We conclude by investigating dynamic programming approaches and the special case where the virtual networks are isomorphic to stars.

In Section 3.4, we consider the VNE problem under data uncertainty. Starting from a chance-constrained formulation, we show how the problem with uncertainty can be tackled by applying Γ-robustness to its MILP formulation. Subsequently, we exploit the Γ-robust formulation to derive heuristic approaches to the problem with uncertainty.

Finally, in Section 3.5, we present a computational study of the VNE problem. We consider both, the deterministic problem and the one under data uncertainty and evaluate the here presented exact and heuristic solution approaches.

We conclude this chapter by a brief summary of our results and by an outlook into further research directions.

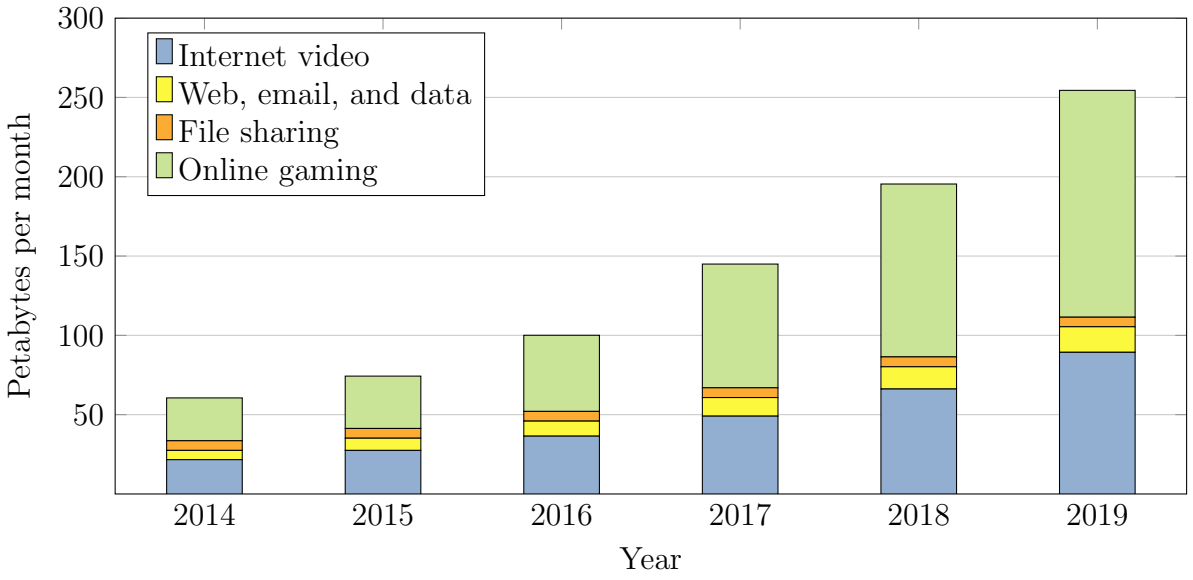


Figure 3.1: The forecasted global consumer Internet traffic volume between 2014 and 2019 by Cisco Systems Inc. [37], categorized into Internet video, Web, email and data, File sharing, and Online gaming.

Previous publications

Partial results presented in this chapter have been found in collobaration with different co-authors and have been published before. In particular, this concerns the work of Coniglio, Grimm, Koster, Tieves, and Werner [40] where an MILP formulation of the VNE problem is presented, together with preliminary computational experiments. This also includes some of the results on the complexity of VNE, as presented in Section 3.3, which have previously been published by Amaldi, Coniglio, Koster, and Tieves [5]. Furthermore, the results shown in Section 3.4 together with the corresponding computations in Section 3.5 have been published by Coniglio, Koster, and Tieves [41, 42].

3.1 Introduction

3.1.1 A motivation for virtual network embedding

Telecommunication services and especially services associated with the Internet are central elements of our daily life. Apparently, the importance and the sheer amount of such services has been rapidly growing over the last decades and is expected to do so in the future as well. Figures supporting this claim can be found in many works within the corresponding literature. Here, we point the reader to two of them. One is a report by Cisco Systems Inc. [37] in which it is stated that "Global IP traffic has increased more than fivefold in the past five years, and will increase nearly threefold over the next five years. Overall, IP traffic will grow at a compound annual growth rate of 23 percent

from 2014 to 2019" and "Global Internet traffic in 2019 will be equivalent to 64 times the volume of the entire global Internet in 2005. Globally, Internet traffic will reach 18 gigabytes (GB) per capita by 2019, up from 6 GB per capita in 2014". Another interesting source is Greentouch [65], where the authors estimate that, from 2010 to 2020, the global (wire line) Internet traffic will increase by a factor of 16 (to 250 exabytes per month). For a visualization of this growth, we refer to Figure 3.1. This figure, concerns the forecasted global consumer Internet traffic (not confined to a single service provider's network) which grows in a similar manner as the global Internet traffic. We visualize the traffic growth between 2014 and 2019, categorized into Internet video, Web, email, and data, File sharing, and Online gaming. Interestingly, the growth is mainly carried by the categories Internet video (e.g., YouTube, Netflix) and Online Gaming. These areas, in particular streaming services, are accounted to be most innovative, more and more popular, and require more demanding (with respect to traffic volumes) Internet services than ever before. In this context, see Hartley [70] for an analysis of YouTube's traffic.

On the one hand, this growths concerns the underlying infrastructure of these services, for example the backbone networks or the data centers, which we will refer to as the *physical network* or the *substrate network*. On the other hand, new services and technologies are rapidly emerging, changing and evolving as well, affecting, so to say, the software side of a telecommunication system. As a consequence, such communication systems are subject to a multitude of changes and adaptations, both on physical and on software level. The practical distinction between these two sides is not as clear since, at least from an economical point of view, changes on the software level can require changes on the physical level and vice versa.

To alleviate the interdependence between these two sides, the paradigm of *network virtualization* has garnered much attention and is being advocated as one of the key technologies for the future of networking, see fore instance Chowdhury and Boutaba [35] and Fischer et al. [53]. In its general form, the paradigm allows to decouple the physical (low level) management aspects of a networking environment from those (of higher level) involving service provisioning. For instance, it allows services to be executed on a simulated environment such that these services do not directly depend on the substrate network any more. This way, network virtualization benefits the two main actors of a networking environment: the owner of the *physical* or *substrate network*, the so-called *infrastructure provider* (IP)—who, this way, can solely concentrate on the management aspects of the substrate—and the *service provider* (SP)—who, this way, can only focus on the provisioning aspects of his services.

A prominent application scenario is that of the Internet. The Internet only allows for small and incremental updates to its structure as a consequence of its inherently plural nature. Therefore, it can largely benefit from virtualization techniques as a non-invasive way of upgrading itself, preventing so called *ossification* phenomena. For a more detailed discussion on ossification, we refer to the next subsection. Here, we give an example for network virtualization:

Example 3.1. *One example of network virtualization can be found in a Cloud Computing (CC) environment. In such environment, computing resources are aggregated into*

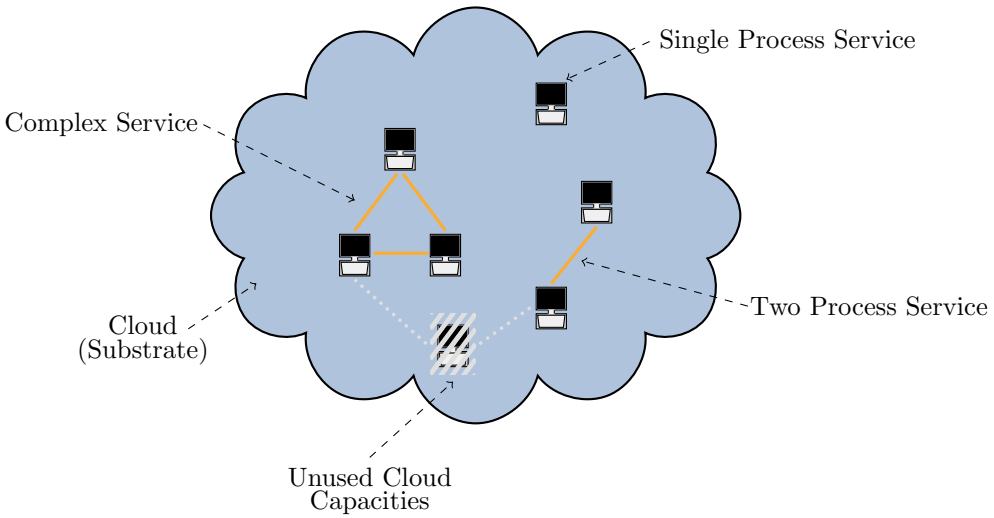


Figure 3.2: An Example for network virtualization: A computing cloud hosting three different services (tasks).

a network, the so called cloud, respectively the substrate. This cloud is managed by the so called Infrastructure Provider and can be accessed by users (Service Providers) so to host different services, respectively virtual networks.

Assume, that the cloud is some computing cluster and the virtual network requested by a user consists of two elements. The first is a server where the user can access and process his data, so to say a "front-end process". The second one is a secondary background, respectively "back-end", process. This way, the service hosted in the cloud requires computing capacities for the two processes, possibly interconnected to allow for a certain rate of data exchange. In this example, the benefit of such virtualization is that the SP can solely focus on his service, i.e., the two processes and their parallelization, while he is not concerned about the management of the underlying computers and their networking environment. Similar, the IP only needs to focus on the management of the cloud and not on the technical details within the different services in his network.

Note that such services, i.e., the virtual networks, can have any topology. We refer to Figure 3.2 for a visualization of this example.

In general, the *Virtual Network Embedding* problem (VNE) concerns the infrastructure provider and arises in the following kind of situation: Assume that the IP operates a substrate network, i.e., a given network topology which includes certain amounts of resources available at each node and link in the network. In the following, we refer to these resources as capacities, for example as the available bandwidth of a link, or as the available units of computing power at a node. The IP is approached by a set of SPs, each offering to rent a certain amount of these capacities at a certain price. Hereby, each *request* is, in turn, given as *virtual network* (VN) where the requested amount of capacities, corresponding to links and nodes, is specified as link and node demands. Then, VNE is, in the perspective of the IP, the problem of, in the first place,

deciding whether to *accept or reject* a subset of VN requests issued by the customers (the *admission control* aspect). Then, in the second place, it is the problem of *embedding* the accepted VNs onto the substrate network, subject to link and node capacity constraints. This decision is driven by the profit obtained by accepting a VN, i.e., the IP maximizes his total profit. Clearly, it involves a trade-off between the resource consumption and the profit of an embedded VN.

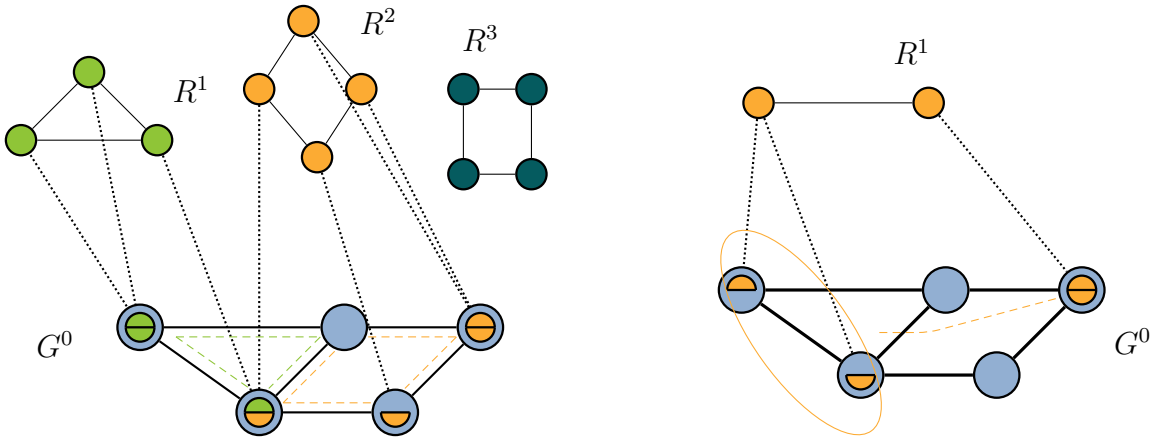
In particular, we assume that the substrate network is given either as a directed or as an undirected graph with specified node and link capacities. Without loss of generality, we assume that the substrate is "empty", i.e., that no virtual networks are already embedded, respectively that no node and link resources are already reserved. Further, we assume that each VN is composed of a set of *virtual nodes* and of a set of *traffic* or *link demands* between pairs of virtual nodes. We assume that each node demand is endowed with an estimate of the node resources it requires, which we refer to as *node demands*. An embedding of a VN thus consists of virtual-to-physical *node-to-node* and *link-to-path* mappings which, in combination with all other node-to-node and link-to-path mappings, do not exceed the node and link capacities of the substrate network. As a short-form, we will employ the abbreviations *node mapping* and *link mapping*.

Example 3.2. *We refer to Figure 3.3 (a) for an example of a VNE problem. In this figure, the substrate network is given in blue and there are three candidate VNs, depicted in green, orange and turquoise. The infrastructure provider can choose any combination of these to embed in his substrate. Here, the IP decided to accept two VNs and to reject the turquoise VN. The mappings of the nodes and links of the accepted VNs are indicated by the dashed lines.*

As customary for VNE, see Fischer et al. [53] and the references therein, we also consider *locality* constraints which restrict the set of physical nodes onto which a virtual node can be mapped. This allows for the encoding of technical specifications which are only met by certain physical nodes, as well as of geographical restrictions which prevent, for instance, the mapping of an important server which is too far away from its customers to reduce latency issues. We call the case that a virtual node is only allowed to be mapped on a single physical node *extreme locality*. Note that in the case that the VN with extreme locality is accepted, the corresponding node mapping is implicitly given.

It is important to mention that traffic flows of a virtual network can disappear if virtual nodes are *co-located*, i.e., if two virtual nodes of the same VN are embedded on the same physical node, their traffic demand does not need to be routed. If this is the case, the communication is assumed to take place "in house", e.g., within a single computer, respectively data center, such that infinite capacity, in comparison to the link capacity of the substrate, is available. We extend this to the following remark:

Remark 3.1. *We allow for a "many to one" mapping of virtual to physical nodes, i.e., the mapping of any number of virtual nodes belonging to the same VN request onto the same physical node, the so-called* co-location *(provided that the substrate node capacity*



(a) A VNE instance

(b) Enforced parallelism

Figure 3.3: Embedding virtual networks: In (a): two VN requests, R^1 (green) and R^2 (orange), are embedded into the substrate G^0 (blue), while R^3 (turquoise) is rejected. In (b): a (forbidden) embedding of a single VN (orange), a single virtual node is split on multiple physical nodes.

is sufficient). See Figure 3.3 (a) for an example where two virtual nodes of the green VN are co-located.

On the contrary, we forbid the mapping of a single virtual node to more than one single physical node, i.e., a "one to many" mapping of virtual to physical nodes. We assume that each task which can be parallelized on multiple physical nodes is described in a VN request with as many virtual nodes as the number of (parallel) nodes, respectively CPUs, it can use. This way, a task will be split only if, by splitting it, a more profitable embedding can be obtained. On the contrary, splitting a single virtual node would force us to split both the node and the traffic demands over the physical network in a not well defined way, see Figure 3.3 (b). In the above described situation, we do not allow the IP to enforce additional parallelism in the service specified by an SP.

In this work, we focus on the offline version of VNE in which the substrate network and the set of VN requests are known beforehand. This suits the case in which the VN requests are issued ahead of the time when their service will be activated, thus allowing for sufficient time for offline planning. Such requests are, typically, quite large in terms of resource requirements and, if accepted, sufficiently long lasting to assume that they will be embedded for an indefinite time span.

Differently to the NDP problem, in the situation of the VNE problem, the number of commodities is variable as it depends on the embedded VNs which is, in general, a subset of all VNs. Another difference is that the source and target of the requested traffic flows are not determined as long as the exact node to node mapping is not fixed.

However, as soon as those decisions are made, the problem boils down to a variant of the Multi-Commodity Flow Problem: Assuming a multiplicity of one unit of installed link capacity, can all commodities, that is, each of the communications of each accepted VN, be routed on the substrate?

While a formal definition of the VNE problem is given in Section 3.2, many extensions and variations of the VNE problem are possible. We list some of the main aspects usually considered for VNE, partially including some of the aspects mentioned above. Note that this list is by no means exhaustive, we refer to the survey of Fischer et al. [53] for a more thorough itemization.

- *Online* versus *offline* embedding: depending on the situation that we consider, VN requests can either arrive dynamically over time or be known *a priori*. If VNs arrive one at a time, the problem whether a single VN can be embedded on the substrate is usually called the *virtual private network* problem.
- *Resource requirements* holding for each VN request, usually of computing power (nodes) and of bandwidth (links). In this work, both kinds of resources are taken into account. However, cases where more types of resources are available can be considered as well.
- *(Extreme) locality requirements* imposing that a virtual node may only be mapped to a specified subset of the physical nodes, e.g., those belonging to a given region, or in the extreme case, that a virtual node may only be mapped on a specific physical node.
- *Admission control:* the possibility of accepting or rejecting a VN request, e.g., because it is not sufficiently profitable or it is too resource demanding. In many cases, it is assumed that admission control has already been carried out.
- *Routing schemes*, depending on the considered architecture and protocols, e.g., *splittable routing* for UDP, or *unsplittable, single path, routing* for MPLS is required for the traffic flows of the embedded VNs.
- *Embedding profits and costs:* in this work, we consider the case that each VN offers a certain profit if it is accepted. Additionally, utilization costs for used node and link capacities can be taken into account as well.
- *Network design* and *rent-at-bulk aspects*: the optimization problem may include the dimensioning of the substrate, the physical resources are often obtained in bulks with volume discounts.

Throughout our work, we assume a *single path unsplittable routing scheme*, which is often preferred to a splittable one so to avoid packet reordering issues, see, e.g., Yu et al. [125] for further comments in this direction. Nevertheless, many results which we present here can be generalized to the splittable case.

3.1.2 Literature

Network ossification One of the purposes of virtualization concepts is to overcome ossification problems in communication networks as described by Anderson et al. [10] and by Turner and Taylor [119]. We give a brief introduction into this matter on the use case of the Internet. The Internet is hugely successful and literally created a world wide network. However, the infrastructure required to support this service is gigantic, multi-faceted and provided, respectively controlled, by many different parties, see e.g., Malik [93] for a list of the worlds biggest broadband companies.

As a consequence, structural updates like introducing IPv6 take tremendous time to conduct, as is reported by Vaughan-Nichols [120]. I.e., the pure size and the thus introduced dispersion of such infrastructure make the network difficult to manage. The same problem arises in many other aspects as well, e.g., when considering routing- (as in Feamster et al. [51]) or security questions (as in Sekar et al. [116]). In particular, even the adaption of new architectures as described by Braden et al. [24] and by Clark et al. [38] is endangered or even completely prevented.

This phenomenon is called *ossification* and hampers the forthcome of the technology. It is one of the main reasons that the Internet "only just works" and cannot offer "better" solutions for numerous tasks, e.g., the implementation of more secure or more reliable routing protocols, see Handley [69], is often not possible.

We conclude this paragraph with an example taken from Handley [69]: The ARPAnet as the predecessor of the Internet introduced TCP/IP as a new technology and switched over to the new protocol in a flag-day on January 1^{st} in 1983. The switchover concerned about 400 computers in the network, all of which were updated to the new standard at this date. Apparently, such flag-day for a similar change is not possible for the Internet as of today. At this point, virtualization concepts are believed to be able to replace these flag-days, this way being a critical enabler for future Internet technologies. Respectively, as formulated by Feamster et al. [52], they are expected to be an inherent part of the future Internet architecture.

Virtual network embedding Network virtualization concepts surfaced as a tool for managing the increasingly complex digital telecommunication networks within the late 90s. Two of the first works in this area, establishing the basic idea and first concepts, are the ones by Yemini and Da Silva [123] and by Richardson et al. [109]. Subsequently, the topic received increasing interest within the networking community, such that, for example, Google Scholar lists 17600 publications including the term "virtual network embedding" from the year 2000 to May 2016 and, including the term "network virtualization", even 43900 publications can be found in the same time frame. We take these numbers as an indicator for the importance of the VNE problem.

While we have given an introduction, respectively a motivation, to the VNE problem at the beginning of this section, we refer to the works of Turner and Taylor [119], of Anderson et al. [10], and of Feamster et al. [52] for additional background information. For

a general overview on the available literature on VNE, we refer to the excellent surveys of Fischer et al. [53], on the Virtual Network Embedding Problem, and of Chowdhury and Boutaba [35], on network virtualization in general.

In the context of our work, we generally understand the VNE problem as an optimization problem concerned with allocating the scarce resources of a given substrate network to a set of resource demanding virtual networks. In this regard, most of the literature on VNE employs a heuristic approach to such optimization problem, in which node and link mappings are carried out sequentially. Consider, for example, the works by Zhu and Ammar [129], by Yu et al. [125], and by Chowdhury et al. [36]. For a more complete picture, we refer to the aforementioned surveys. In many cases, the VNE problem is also considered in an online setting where the virtual network requests arrive over time. Examples are given by Zhu and Ammar [129], where the authors also account for reconfigurations of a given embedding, and by Chowdhury et al. [34], where deterministic and randomized rounding techniques are employed. Manifold further types of heuristics have also been applied to the VNE problem. We exemplary mention some of them: For a rounding algorithm based on column generation, also encompassing admission control, see Jarray and Karmouch [79]. For a greedy algorithm based on the degree of utilization of the different physical nodes, see Cheng et al. [33]. Also, see Zhang et al. [128] for an energy-efficient version of VNE subject to Gaussian traffic demands. For meta-heuristic approaches, we refer the reader to the works of Fajjari et al. [50] and Pages et al. [101] and the there-in contained references.

Among the few exact approaches as from Inführ and Raidl [78], from Liu et al. [88] and from Trinh et al. [118], we mention that of Houidi et al. [74, 75], see also Section 3.2, where the authors propose a Mixed-Integer Linear Programming (MILP) formulation to carry out each step of an algorithm for the online version of the problem. We also refer to Coniglio et al. [40], where an MILP formulation for the offline version of VNE is illustrated, also accounting for the installation (or rental) of network capacities (with a rent-at-bulk aspect). Finally, we point the reader to the work of Botero et al. [23] and of Houidi et al. [75], who extend the paper of Houidi et al. [74] to energy-aware and fault-tolerant cases.

Since VNE is very relevant for practical applications, network resilience in the context of data uncertainty, and on the background of quality of service (QoS) guarantees, is a very active topic. We point out that the author contributed in the works of Coniglio et al. [41, 42] where data uncertainty for VNE was tackled by a Γ-robust model, yielding exact and heuristic solution algorithms for the problem, see also Section 3.4. A similar situation, albeit for the VPN problem, is analyzed by Altın et al. [4] and in Kumar et al. [87], where the authors investigate the problem under data uncertainty with respect to the hose model. The scenario of potential link or node failure in the substrate is, e.g., investigated by Habib et al. [68]. Among others, Chen et al. [32] and Yu et al. [124] published further works in this direction.

To the best of our knowledge, most of the papers where the computational complex-

ity of VNE is mentioned claim that the problem is $\mathcal{NP}$-hard by reduction from the k- Multiway Separator Problem, citing an (unpublished) technical report by Andersen [9], devoted to the (therein called) Testbed Allocation Problem. We point out that the report is formulated very imprecise and that it lacks sufficient details to verify the correctness of its content. Alternatively, VNE is shown to be weakly $\mathcal{NP}$-hard by reduction from the 0-1 Knapsack Problem by Fischer et al. [53], respectively in Yu et al. [125], the authors show strong $\mathcal{NP}$-hardness by a straightforward reduction from the unsplittable Multi-Commodity Flow Problem. However, the latter only holds for the special case of VNE where the node mapping is already given and an unsplittable routing is required. More detailed results, including inapproximability results and an analysis of the VNE problem in fixed dimensions are provided by Amaldi et al. [5], see also Section 3.3. In this context, we point out that the special case of embedding virtual stars is an important topic in the application, i.e., in the context of virtual cluster embedding. We refer to the work of Rost et al. [113] for additional information on that matter.

Technical background In this work, we consider the VNE problem purely as a given optimization problem. Therefore, we mention some use cases and point the reader to some experimental realizations. For further information, we refer to the cited articles and the therein contained references.

Network visualization concepts can be realized in many different settings, see for example the work of Chowdhury and Boutaba [35] for a description of four different incarnations. One such way to provide a virtual networking environment is, for example, via so called layer 3 Virtual Private Networks (L3VPN), employing level three protocols as (IP or MPLS). In particular, we refer to Rosen and Rekhter [111] for a detailed description of the BGP/MPLS L3VPN model.

Currently, network virtualization concepts are employed in a number of research testbeds, for example, in G-Lab, see Schwerdel et al. [115] and in 4WARD, see Brunner et al. [25]. In this context, we mention the work of Carapinha and Jiménez [28] where especially the 4WARD architecture is described in detail.

As final reference, we point the reader to the work of McKeown et al. [96] and the OpenFlow protocol, supported by the Open Networking Foundation [99]. There, the authors describe a virtual networking environment, namely OpenFlow, which was, at first, realized within two buildings of the Stanford University, and which was later adopted by Google for the use within their data centers as described by Hoelzle [73]. The example underlines both, the importance of virtualization concepts and the strong interest of the industry into the topic.

3.2 Formalizing virtual network embedding

3.2.1 Notation and definitions

For the remainder of the chapter, we introduce the following notation. Let R be the set of VN requests. In the following, the superscript 0 denotes data pertaining to the physical network, while we adopt the superscript $r \in R$ for VN requests.

Let the directed graph $G^0 = (V^0, A^0)$ represent the physical substrate network. The node set V^0 describes physical nodes symbolizing, e.g., computers in a data center, and the arc set A^0 represents the possible interconnections between those. We assume that the substrate is endowed with capacity restrictions, i.e., with node capacities c_i^0 for all $i \in V^0$ and with link capacities k_{ij}^0 for all $a_{ij} \in A^0$.

Let $R = 1, \ldots, |R|$ be the set of VN requests, with profits $p^r \geq 0$ for all $r \in R$, where each VN $r \in R$ is given as a directed graph $G^r = (V^r, A^r)$.

For each VN $r \in R$, V^r describes the set of virtual nodes and, for each virtual node $v \in V^r$, the parameter ω_v^r denotes the corresponding node demand, i.e., the amount of resources the virtual node v consumes if it is embedded on a physical node. In this context, we encode the locality constraints in the set $V^0(r, v) \subseteq V^0$. That is, $V^0(r, v)$ contains exactly the physical nodes onto which the virtual node $v \in V^r$ is (only) permitted to be mapped.

Similar, the virtual arc set A^r describes the required connection between the virtual nodes and, for each arc $vw \in A$, the parameter $d_{vw}^r \geq 0$ denotes the corresponding traffic demand between the virtual nodes $v, w \in V^r$. That is, assuming that the virtual nodes v and w are mapped on some physical nodes $i \in V^0$ and $j \in V^0$, d_{vw}^r describes the amount of flow which has to be sent from node i to node j in the substrate. However, if v and w are embedded onto the same physical node, or not at all, no traffic has to be sent. We assume that all traffic requirements are encoded in the (possibly sparse) traffic matrix $D^r \in \mathbb{R}_+^{|V^r| \times |V^r|}$.

In the perspective of the IP, the Virtual Network embedding problem arises as follows. We are looking for a *collection* of VNs *maximizing with maximum profit*, such that, there is enough capacity on the substrate to, at the same time: i) map *all* virtual nodes belonging to these VNs on the physical nodes and ii) route *all* traffic demands required by these VNs depending on their node embeddings on the physical arcs. Note that, in this situation, we assume an *unsplittable* routing scheme.

We formalize this in the following definition.

Definition 3.1 (VNE)**.** *Given a directed graph $G^0 = (V^0, A^0)$, capacity functions*

$$c^0 : \quad V^0 \to \mathbb{Z}_+, \quad i \ \mapsto c_i^0, \tag{3.1a}$$
$$k^0 : \quad A^0 \to \mathbb{Z}_+, \quad ij \mapsto k_{ij}^0, \tag{3.1b}$$

a set R of VNs, and for each $r \in R$ a profit $p_r \in \mathbb{Z}_+$ and virtual demand functions

$$w^r : \quad V^r \to \mathbb{Z}_+, \quad v \ \mapsto \omega_v^r, \tag{3.1c}$$
$$d^r \ : \quad A^r \to \mathbb{Z}_+, \quad vw \mapsto d_{vw}^r, \tag{3.1d}$$

*the **Virtual Network Embedding Problem** (VNE) asks for a subset $S \subseteq R$, and for each $r \in S, i \in V^0$ and $ij \in A^0$ for the functions*

$$x_i^r \ : \quad V^r \to \{0,1\}, \quad v \ \mapsto x_{iv}^r \ , \quad \forall r \in S, \forall i \in V^0 \tag{3.1e}$$
$$f_{ij}^r : \quad A^r \to \{0,1\}, \quad vw \mapsto f_{ij}^{vw,r}, \quad \forall r \in S, \forall ij \in A^0 \tag{3.1f}$$

satisfying the properties

$$\sum_{i \in V^0(r,v)} x^r_{vi} = 1 \qquad \forall r \in S, v \in V^r \tag{3.1g}$$

$$\sum_{r \in S} \sum_{\substack{v \in V^r: \\ i \in V^0(r,v)}} \omega^r_v x^r_{vi} \leq c^0_i \qquad \forall i \in V^0 \tag{3.1h}$$

$$\sum_{r \in S} \sum_{v,w \in V^r} d^r_{vw} f^{vw,r}_{ij} \leq k^0_{ij} \qquad \forall ij \in A^0 \tag{3.1i}$$

$$\sum_{ij \in \delta^+(i)} f^{vw,r}_{ij} - \sum_{ji \in \delta^-(i)} f^{vw,r}_{ji} = x^r_{vi} - x^r_{wi} \qquad \forall r \in S, vw \in A^r, i \in V^0 \tag{3.1j}$$

such that $\sum_{r \in S} p^r$ *is maximized.*

The admission control aspect is encoded in the selection of the set S, i.e., for $r \in R$, it is $r \in S$ if and only if r is accepted. Thus, the profit obtained by accepting the VNs in S is described by $\sum_{r \in S} p^r$. Constraints (3.1g) enforce that each virtual node is mapped onto a (single) substrate node if and only if the corresponding request is accepted. Constraint (3.1h) and Constraint (3.1i) guarantee that the capacity of each physical node and link is not exceeded by the link and node mappings of the accepted requests. Constraints (3.1j) are flow balance constraints ensuring that, for each embedded VN, the routing of its virtual traffic matrices takes place. Note that the right hand side of the flow balance is a function of the mapping of the virtual nodes $v, w \in V^r$, as encoded by $x^r_{vi} - x^r_{wi}$. Each physical node $i \in V^0$ acts as a source node if $x^r_{vi} = 1$ and $x^r_{wi} = 0$, as a sink node if $x^r_{vi} = 0$ and $x^r_{wi} = 1$, and as a "regular" intermediate node, i.e., not a source nor a sink node, if $x^r_{vi} = x^r_{wi} = 0$. If $x^r_{vi} = x^r_{wi} = 1$, then the two virtual nodes v, w are *co-located*, i.e., they are mapped to the same physical node and, hence, their traffic demand (for both pairs v, w and w, v) vanishes. That is, the traffic demand does not need to be routed on any physical link. Since $f^{vw,r}_{ij}$ is integer, single path, unsplittable routing scheme is enforced.

This definition, as well as the many of the methods that we will introduce in the remainder of this work, can directly be adapted to the case of splittable routing by relaxing the field of the f variables to $f^{vw,r}_{ij} \in [0, 1]$. In this context, we point out that the VNE problem can as well be defined on an *undirected* substrate network. In that case, the flows encoded in the functions f have to be indexed over both directions of the underlying edge for which the flows in both directions share the same edge-capacity. We refer to Subsection 1.3, Inequality (1.8) where the same was discussed for NDP. In the course of this chapter, we point will out when we consider a directed or an undirected substrate network. In the undirected case, we write $G^0 = (V^0, E^0)$ for the substrate.

3.2.2 Virtual network embedding as MILP

A basic formulation As for the NDP problem, see Remark 1.1, a MILP formulation for the VNE problem is derived from the definition. The following remark gives such

formulation for the VNE problem on a directed substrate network. This formulation can easily be adapted for the undirected case. Note that the formulation was first shown by Coniglio et al. [40], although with extra aspects, notably, rent-at-bulk, and appeared subsequently in the works of Coniglio et al. [41, 42].

Remark 3.2. *We employ three groups of decision variables:* y^r, x^r_{vi}, *and* $f^{r,vw}_{ij}$. *Let* $y^r \in \{0,1\}$ *take value* 1 *if the VN of index* $r \in R$ *is accepted and* 0 *otherwise. Let* $x^r_{vi} \in \{0,1\}$ *be equal to* 1 *if the virtual node* $v \in V^r$, *pertaining to VN* $r \in R$, *is mapped onto the physical node* $i \in V^0$, *with* $x^r_{vi} = 0$ *otherwise. Let* $f^{r,vw}_{ij}$ *take value* 1 *if the traffic between the two virtual nodes* $v, w \in V^r$, *for a request* $r \in R$, *is routed over the physical link* $ij \in A^0$, *and 0 otherwise. We cast VNE as MILP:*

$$\max \quad \sum_{r \in R} p^r y^r \tag{3.2a}$$

$$\text{s.t.} \quad \sum_{i \in V^0(r,v)} x^r_{vi} = y^r \qquad \forall r \in R, v \in V^r \tag{3.2b}$$

$$\sum_{r \in R} \sum_{\substack{v \in V^r: \\ i \in V^0(r,v)}} \omega^r_v x^r_{vi} \leq c^0_i \qquad \forall i \in V^0 \tag{3.2c}$$

$$\sum_{r \in R} \sum_{v,w \in V^r} d^r_{vw} f^{vw,r}_{ij} \leq k^0_{ij} \qquad \forall ij \in A^0 \tag{3.2d}$$

$$\sum_{ij \in \delta^+(i)} f^{vw,r}_{ij} - \sum_{ji \in \delta^-(i)} f^{vw,r}_{ji} = x^r_{vi} - x^r_{wi} \qquad \forall r \in R, v, w \in V^r, i \in V^0 \tag{3.2e}$$

$$y^r, x^r_{vi}, f^{vw,r}_{ij} \in \{0,1\}. \tag{3.2f}$$

The MILP constraints are basically the same as the ones in Definition (3.1), *with the difference that the* y *variables now encode the admission control aspect, i.e., they model what is contained in the set* S *and what not. The profit induced by the accepted VNs is given by the* y *variables and given by the Objective Function* (3.2a). *As in Definition 3.1, Constraints* (3.2b) *enforce that each virtual node is mapped onto a single substrate node if and only if the corresponding VN is accepted. Constraints*(3.2c) *and Constraint* (3.2d) *guarantee that the capacity of each physical node and link is not exceeded. Constraints* (3.2e) *are flow balance constraints ensuring that, for each embedded VN, the routing of the traffic matrices takes place.*

We point out, that the y variables can be projected away. However, we prefer such extended formulation for the sake of clearness and readability.

VNE with rent-at-bulk In the following remark, the above model is compared to the formulation presented by Coniglio et al. [40]. In the cited work, the formulation is expanded to include network design elements, as well. This way, in contrast to the previous assumptions, the substrate is not given, respectively fixed, but has to be constructed or rented by the IP at certain costs. Consequentially, this version of the VNE problem entails an (optimal) decision of the dimensioning of the substrate together with an (optimal) decision of embedding a collection of the VNs.

Rent-at-bulk aspects can be incorporated in Problem (3.2a)– (3.2f) as follows. Assume that, in the VNE problem, the IP has to construct the substrate network as well. Hereby, similar to the NDP problem, potential node and link locations and capacity modules for both locations are specified. Then, the IP has to decide, which node and link capacity modules are to be installed where in the network. Note that a node does not need to have any capacity installation to be able to forward traffic.

In this context, let U^0 and Q^0 be the set of capacity bulks of different size for the physical nodes and links and let α^{u_i} and β^q_{ij}, for $u_i \in U^0$ and $q_{ij} \in Q^0$ be the corresponding rent-at-bulk costs for one batch of capacity for the nodes, respectively of the capacity for the links. Economies of scales dictate

$$\frac{\alpha^{u_i}}{u_i} \geq \frac{\alpha^{u_j}}{u_j} \qquad \forall u_i, u_j \in U^0 : u_i \leq u_j, \tag{3.3a}$$

$$\text{and} \quad \frac{\beta^{q_{ij}}}{q_{ij}} \geq \frac{\beta^{q_{kl}}}{q_{kl}} \qquad \forall q_{ij}, q_{kl} \in Q^0 : q_{ij} \leq q_{kl}. \tag{3.3b}$$

That is, we assume decreasing unit costs for larger bulks of rented capacity. We further assume that, per physical node $i \in V^0$, at most B^0_i units of capacity can be installed and, for each arc $ij \in A^0$, at most K^0_{ij} units can be installed. The following remark gives the adapted formulation:

Remark 3.3. *We adapt the formulation for the VNE problem by adding, for $i \in V^0$, the variables $g^u_i \in \mathbb{Z}_+$ for all $u \in U^0$ and for $ij \in A^0$, the variables $h^q_{ij} \in \mathbb{Z}_+$ for all $q \in Q^0$, indicating the number of bulks of type u, respectively of type q, installed on node i, respectively on arc ij. Then, replacing the Objective Function* (3.2a)*, the Node Capacity Constraint* (3.2c)*, and the Link Capacity Constraint* (3.2d)*, the formulation with rent-at-bulk reads:*

$$\max \quad \sum_{r \in R} p^r y^r - \sum_{i \in V^0} \sum_{u \in U^0} \alpha^u g^u_i - \sum_{ij \in A^0} \sum_{q \in Q^0} \beta^q h^q_{ij} \tag{3.4a}$$

$$\text{s.t.} \quad (3.2b), (3.2e) \tag{3.4b}$$

$$\sum_{r \in R} \sum_{\substack{v \in V^r: \\ i \in V^0(r,v)}} \omega^r_v x^r_{vi} \leq \sum_{u_i \in U^0} u_i g^u_i \qquad \forall i \in V^0 \tag{3.4c}$$

$$\sum_{r \in R} \sum_{v,w \in V^r} d^r_{vw} f^{vw,r}_{ij} \leq \sum_{q_{ij} \in Q^0} q_{ij} h^q_{ij} \qquad \forall ij \in A^0 \tag{3.4d}$$

$$\sum_{u_i \in U^0} u_i g^u_i \leq B^0_i \qquad \forall i \in V^0 \tag{3.4e}$$

$$\sum_{q_{ij} \in Q^0} q_{ij} h^q_{ij} \leq K^0_{ij} \qquad \forall ij \in A^0 \tag{3.4f}$$

$$(3.2f)\ , g^u_i, h^q_{ij} \in \mathbb{Z}_+. \tag{3.4g}$$

The objective now includes the rent-at-bulk cost and the capacity constraints have no fixed right hand side any more but depend on the actual capacity installation. The two new

inequalities (3.4e) *and* (3.4f) *guarantee that the overall available capacity installation for each node and for each link is satisfied.*

In comparison to the MILP models available in the literature, we point out that, although the formulation proposed by Houidi et al. [74, 75] can be used in the off-line setting as well, it does not allow for admission control, it lacks the rent-at-bulk scheme for physical resources, and it only allows for a splittable routing. Since such formulation aggregates, for each VN, all the flow between pairs of virtual nodes that are mapped on the same pair of physical nodes, by imposing integrality on the corresponding flow variables we would introduce extra routing constraints, thus incorrectly forcing all such heterogeneous flows to share the same physical path.

3.2.3 VNE split into two phases

The VNE problem is composed by two subproblems, a *node-mapping* and a *link-mapping* problem. A set $S \subseteq R$ is a feasible solution to the VNE problem if and only if a feasible node-embedding for all virtual nodes of all VNs in S can be found such that, based on this mapping, a feasible link-embedding for the virtual traffic matrices of all VNs in S can be found as well. In the literature, e.g., in the work of Zhu and Ammar [129] or in the work of Chowdhury et al. [36], and also in the heuristics that we will propose in Section 3.4.3, this structure is often exploited in heuristic approaches to the VNE problem. That is, the two sub-problems are decoupled and then solved in sequence: at first, the node mapping sub-problem is solved, yielding a set S^1 of accepted requests. Following, a second stage problem is solved, possibly again with admission control. There, y^r is fixed to zero for all $r \in R, r \notin S_1$, and all node-embeddings of the first phase are fixed. Then, the second stage problem looks for a feasible, simultaneous link mapping for all accepted VNs.

In the following, the two subproblems are also called *phases*. Thereby, phase I refers to the node-mapping problem and phase II to the subsequent link-mapping problem. For later reference, we give a formal description of the two phases as MILPs in the following. We start with the *first phase*:

Remark 3.4. *The first phase problem is obtained by removing in Problem* (3.2a)– (3.2f) *all constraints related to the link mapping, i.e., we remove Constraint* (3.2d) *and Constraint* (3.2e) *and the flow variables* $f_{ij}^{vw,r}$ *reported in* (3.2f). *We obtain*

$$\max \quad \sum_{r \in R} p^r y^r \tag{3.5a}$$

$$\text{s.t.} \quad (3.2b), (3.2c) \tag{3.5b}$$

$$y^r, x_{iv}^r \in \{0, 1\}. \tag{3.5c}$$

From a combinatorial point of view, we refer to this subproblem as a Multiple Knapsack Problem with Grouped items (MKP-G). It is a Multiple Knapsack Problem (MKP), i.e., an extension of the classical Knapsack Problem where multiple knapsacks are present, in which the items have to be grouped so that, if an item is put into one

of the knapsacks, then all the other items in the same group have to be put in some knapsack as well. From a VNE perspective, each group of items corresponds to the set of virtual nodes belonging to one virtual network request, the items them-self are the virtual nodes, and the knapsacks correspond to the physical nodes.

The *second phase* problem is based on a solution of the first phase problem:

Remark 3.5. *Denote the solution of the first phase problem as* (x^*, y^*)*. The second phase problem is obtained by fixing* $x^r_{vi} = (x^r_{vi})^*$ *for all* $r \in R, v \in V^r$*, and* $i \in V^0(r, v)$ *and solving Problem* (3.2a)– (3.2f) *only in the* f *variables. In detail, the second phase problem writes*

$$\max \quad 1 \tag{3.6a}$$

$$\text{s.t.} \quad (3.2d) \tag{3.6b}$$

$$\sum_{ij \in \delta^+(i)} f^{vw,r}_{ij} - \sum_{ji \in \delta^-(i)} f^{vw,r}_{ji} = (x^r_{vi})^* - (x^r_{wi})^* \quad \forall r \in R, vw \in A^r, i \in V^0 \tag{3.6c}$$

$$f^{vw,r}_{ij} \in \{0, 1\}. \tag{3.6d}$$

Since the x and y variables are fixed, the second phase problem boils down to a Multi-Commodity Flow Problem. If the optimal solution value is equal to one, the solution of the first phase directly yields an optimal solution for the VNE problem. However, if the second phase problem is infeasible, no non-trivial solution for the VNE problem is obtained. An improvement in this direction can be made by extending the second phase problem to include admission control as well:

Remark 3.6. *Let* $S := \{r \in R \mid y^r = 1\}$*. The second phase problem with admission control writes:*

$$\max \quad \sum_{r \in S} p^r y^r \tag{3.7a}$$

$$\text{s.t.} \quad (3.2d) \tag{3.7b}$$

$$\sum_{ij \in \delta^+(i)} f^{vw,r}_{ij} - \sum_{ji \in \delta^-(i)} f^{vw,r}_{ji} = \begin{cases} y^r & \text{if } (x^r_{vi})^* = 1 \neq (x^r_{wi})^* \\ -y^r & \text{if } (x^r_{wi})^* = 1 \neq (x^r_{vi})^* \\ 0 & \text{else} \end{cases} \quad \begin{matrix} \forall r \in R, \\ i \in V^0, \\ vw \in A^r \end{matrix} \tag{3.7c}$$

$$y^r, f^{vw,r}_{ij} \in \{0, 1\}. \tag{3.7d}$$

In this problem, the VNs accepted in the first phase can still be rejected, but if they are accepted, the corresponding node-mapping found in phase one is kept. This way, a solution of the second phase problem can accept less VNs as were accepted in the first phase but will yield a feasible solution in any case. In the following, we refer Problem (3.7a)– (3.7d) *as to the second phase (problem).*

The second subproblem will be referred to as to an Unsplittable Multi-Commodity Flow Problem with Admission Control (UMCF-AC). In it, certain demands can be neglected and a linear function which associates a profit to each accepted group of demands

Algorithm 3.2 VNE: two-phase heuristic

Solve Phase I subproblem via the MILP (3.6a)–(3.5c)
if Second Phase includes admission control **then**
  Let $\tilde{R} := \{r \in R : (y^r)^* = 1\}$
  Replace $(x^r_{vi})^* - (x^r_{wi})^*$ by y^r appropriately
  Solve Phase II subproblem via the MILP (3.7a)–(3.7d)
else
  Solve Phase II subproblem via the MILP (3.6a)–(3.6d)
end if

is maximized. The problem is also subject to "grouping" constraints by which, if a demand for a pair of virtual nodes is routed, then all the demands between pairs of virtual nodes belonging to the same group must be routed as well.

We conclude by pointing out that, the first subproblem is a relaxation of VNE, whereas the second one is a restriction. This way, solving both phases subsequently yields both, an upper bound and a feasible solution for the VNE problem. For a brief description, we refer to Algorithm 3.2. Clearly, such heuristic can be improved in numerous ways. In particular, we refer to Section 3.4 where, in a robust setting, additional constraints (restrictions) are added to the first phase problem to obtain an overall higher rate of re-acceptance in the second phase. Naturally, the same results can be applied to the non-robust problem as well. For further information, we refer to our computational experiments in Section 3.5.

The algorithm, as presented here, relies on the solution of two MILPs. While we will observe in Section 3.5 that these programs do scale well with increasing problem size, there is clearly room for improvement in this direction, if required. That is, if the two subproblems are still desired to be solved via mixed integer linear programming algorithms, cutting planes, etc. could be added to improve the solution process. Furthermore, if solution–time is critical, both subproblems, i.e., the respective combinatorial problems, can as well be tackled by custom heuristics. This way, the first-phase problem no longer yields a dual bound to the VNE problem. However, by incorporating such heuristics into the MILP solution process, the advantages of both approaches can be combined. We point out that, even when tackling both phases by MILPs, an optimal solution of the second phase does not necessarily yield an optimal solution for the complete problem. In any case, after the second phase problem, the obtained solution will be feasible for the VNE problem, compare our results in Section 3.5.

Other extensions of such two-phase approach, for instance, in an iterative setting, are also possible. For example, in Section 3.4, we present, in a robust setting, an adaptive heuristic, which, based on the second phase solution refines the first phase solution by adding additional constraints and then iterates both phases.

3.3 Computational complexity

In this section, we analyze the computational complexity of VNE. If not stated otherwise, we consider the VNE problem on an undirected substrate network, i.e., we assume that $G^0 = (V^0, E^0)$. We speculate that many of the results presented here can be generalized to the directed case. For example, this applies to some of the results from Subsection 3.3.4 and from Subsection 3.3.5 where we consider dynamic programming approaches, respectively where we refer to the maximum flow problem. Other results e.g., the ones relying on the maximum stable set problem (STAB), require further analysis to be carried over.

3.3.1 The two induced subproblems

We discuss the strong $\mathcal{NP}$-hardness of the two subproblems presented above. At first, we focus on the First Phase Problem (3.2a)– (3.2f). As stated in the previous section, this problem is equivalent to the MKP-G problem. We present the following result:

Lemma 3.1. *MKP-G is strongly $\mathcal{NP}$-hard.*

Proof. In the Multi-Knapsack Problem (MKP), we are given p knapsacks with capacities b_j, for $j = 1, \ldots, p$, and q items with item weights and profits w_i and p_i, for $i = 1, \ldots, q$. One has to find a subset of the items that can be put into the knapsacks, subject to weights and capacities, such that the profit of the item-set is maximized.

In the reduction, it suffices to consider VNs, respectively groups of items, containing only a single node (item). In this case, MKP-G is equivalent to MKP, whose strong $\mathcal{NP}$-hardness is shown by Geng et al. [61]. □

Considering the second phase problem, we point out that, if admission control is not included, the problem is equivalent to the Multi-Commodity Flow Problem. In this case, it is $\mathcal{NP}$-hard for unsplittable flows, see Theorem 1.4, but it is in $\mathcal{P}$, see Theorem 1.2, if a splittable routing scheme is assumed. However, when including admission control, we obtain the UMCF-AC Problem, for which the following result holds.

Lemma 3.2. *UMCF-AC is strongly $\mathcal{NP}$-hard.*

Proof. Consider the *Edge Disjoint Path Problem* (EDPP). In an EDPP instance, given an undirected graph $G = (V, E)$ and k pairs of nodes (s_1, t_1) to $(s_k, t_k) \in V \times V$, one has to decide whether there exists k edge disjoint paths between the source-destination pairs. We refer to the work of Middendorf and Pfeiffer [97], for the strong $\mathcal{NP}$-hardness of EDPP. We show a reduction to UMCF-AC.

Given an instance of EDPP with a graph $G = (V, E)$, we construct an instance of UMCF-AC where $G^0 = G$, with zero capacity on the nodes and unit capacity on the edges. For each pair (s_i, t_i), with $1 \leq i \leq k$, we introduce a VN consisting of two virtual nodes and a virtual edge. That is, let $V^r = \{v_1, v_2\}$ with $d^r_{v_1,v_2} = 1$ and let v_1 and v_2 have zero node capacity requirements for all $1 \leq r \leq k$. Assume that the first phase solution mapped v_1 on s_r and v_2 on t_r.

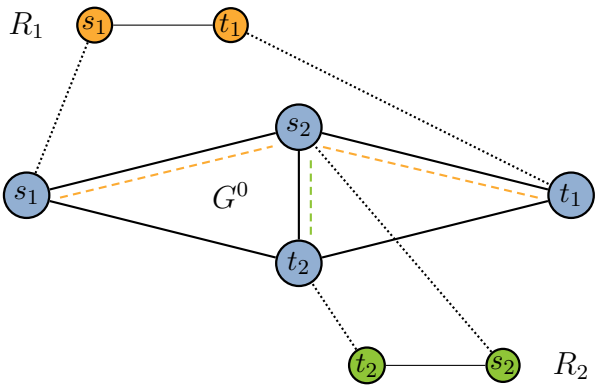


Figure 3.4: An example for the reduction from EDPP with $k = 2$. In this example, two VNs corresponding to paths are embedded without crossing edges due to the presence of unit link demands and capacities.

Clearly, this instance of UMCF-AC admits a solution where all the VNs are embedded if and only if the graph G of the instance of EDPP admits k edge disjoint paths connecting the source-destination pairs. □

For an sketch of the reduction, see Figure 3.4. Note that, when assuming $|V^0(r, v)| = 1$ for all $r \in R$ and $v \in V^r$, i.e., extreme locality, VNE becomes an instance of UMCF-AC. Therefore, the lemma also implies that VNE is strongly $\mathcal{NP}$-hard (when locality conditions are present) by reduction from EDPP.

We point out that the reduction entails even more information. That is, even when considering special cases, i.e., graph classes where many combinatorial optimization problems are solvable in polynomial time, EDPP is still difficult. By this reduction, these results carry over to the UMCF-AC (VNE) problem, as well.

Remark 3.7. *EDPP is $\mathcal{NP}$-hard even for planar graphs, see Chekuri et al. [31], and for series-parallel graphs, see Middendorf and Pfeiffer [97]. By the above reduction, the same holds for the VNE problem.*

The two propositions do not only illustrate that VNE is a hard problem, but also that both of its naturally occurring subproblems are difficult to solve, at least from a theoretical point of view. Hence, heuristics which decompose VNE into the two subproblems, are somewhat counter-intuitive from this perspective. When decomposing VNE, even if VNE was polynomially solvable for that certain class of instances, as a consequence of considering a fixed node mapping, the problem can become $\mathcal{NP}$-hard in the second phase. However, computational experiments have indicated that both phases can be solved at a manageable time investment in practical circumstances, see Section 3.5, such that this decomposition is suitable for practical approaches to VNE.

3.3.2 Strong $\mathcal{NP}$-hardness and inapproximability results

In the previous subsection, we have shown that VNE is strongly $\mathcal{NP}$-hard and, although the reduction is correct, it requires strong locality. We now propose a reduction from the *Stable Set Problem* (STAB), which can be easily extended to the more general case without locality constraints and which also implies a strong inapproximability result.

In STAB, given an undirected graph $G = (V, E)$, and a positive integer $k \in \mathbb{Z}_+$ we look for a stable set, i.e., a subset of nodes that are not pairwise adjacent, of size k. We first propose the reduction for VNE with extreme locality constraints and then extend it to the less restrictive case where each virtual node can be mapped to any physical node. An illustration of the reductions for the two cases with extreme locality and without locality constraints is provided in Figure 3.5 (a) and (b).

Theorem 3.1. *VNE with extreme locality is strongly $\mathcal{NP}$-hard even under the assumption that all VNs are restricted to stars.*

Proof. Consider the decision version of STAB which, given a graph $G = (V, E)$ and a positive integer $k \in \mathbb{Z}_+$, asks whether G contains a stable set of cardinality at least k. We describe a polynomial time reduction from this problem to VNE with extreme locality constraints. For any instance of the decision version of STAB, we construct, in linear time, the VNE instance as follows.

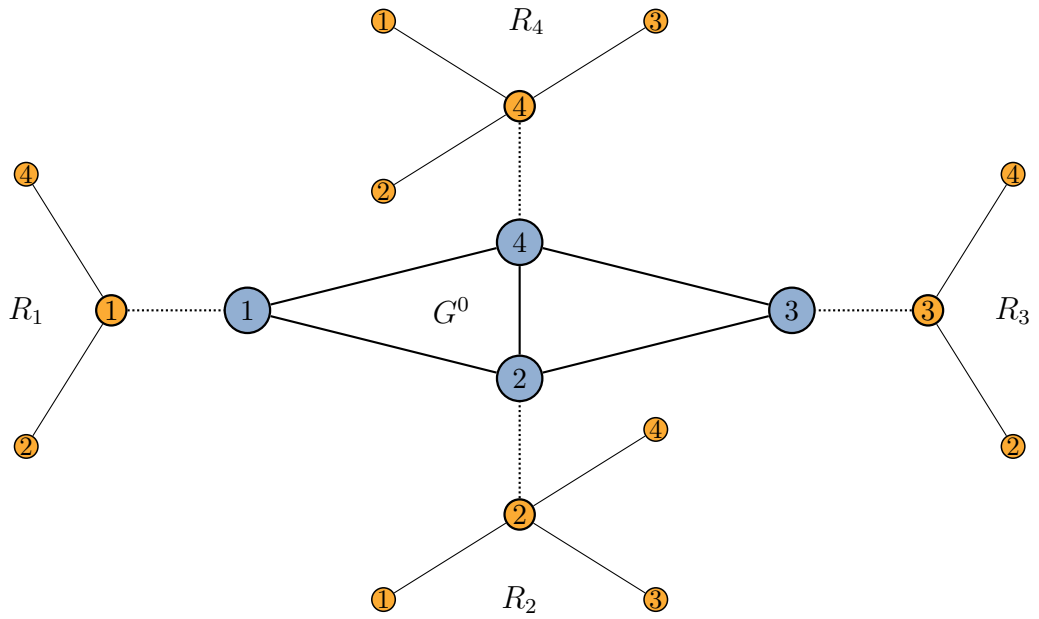
Consider a VNE instance with substrate network $G^0 = G$ with unit node and edge capacities, and $|V^0|$ VN requests with unit profits. Let each VN request $r \in R$ correspond to exactly one node $i \in V^0$. Further, let the VN $r \in R$ corresponding to the physical node $i \in V^0$ be isomorphic to the closed neighborhood of node i. That is, the VN is a star graph with $1+|\delta(i)|$ nodes, a central virtual node corresponding to $i \in V^0$ and $|\delta(i)|$ virtual leaf nodes, one for each neighbor $j \in \delta(i)$. The central virtual node has a unit demand, while all the virtual leaf nodes have a demand of $\frac{1}{\Delta}$, where $\Delta = \max_{i \in V^0} |\delta(i)|$.

The extreme locality constraints are as follows: for each $r \in R$, the central node of the VN G^r corresponding to node $i \in V^0$ can only be mapped to node i, while each virtual leaf node may only be mapped to its corresponding neighbor $j \in \delta(i)$ in G^0. See Figure 3.5 for an illustration, where the labels of the nodes of G^0 and of the VNs indicate the extreme node mapping constraints.

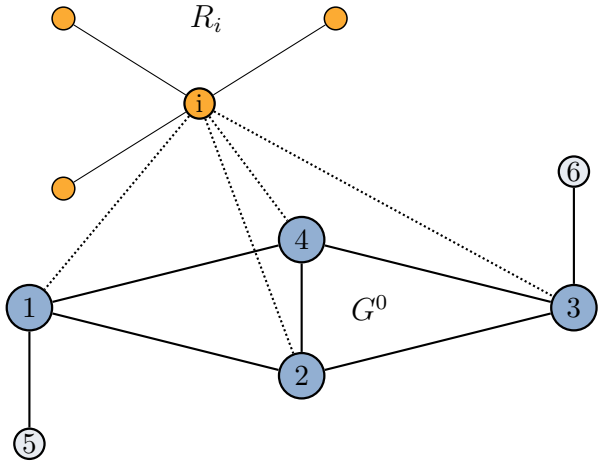
Due to the unit node and edge capacities, if a VN of index $r \in R$ whose central virtual node corresponds to the physical node $i \in V^0$ is accepted, then no other VN of index $r' \in R$ with a center corresponding to some $j \in \delta(i)$ can be accepted. Therefore, the VNE instance admits a feasible solution of total profit k, i.e., a solution where k VNs are simultaneously embedded, if and only if the graph G of the STAB instance contains a stable set of cardinality at least k. □

As above, we point out that the reduction implies $\mathcal{NP}$-hardness even for "simple" cases of the substrate network:

Remark 3.8. *STAB is strongly $\mathcal{NP}$-hard even for planar graphs of maximum degree three as shown by Poljak [105], and for triangle-free graphs, i.e., graphs with chord-less cycles of size four or more, as shown by Garey et al. [59]. Hence, we directly have that*



(a) VNE with strong locality, the node labels indicate the only feasible node mapping.



(b) VNE without locality conditions on an extended (light blue) substrate.

Figure 3.5: Illustrations of the reductions used in Theorem 3.1 (a) and Corollary 3.1 (b). The VNE problem with (a) and without (b) strong locality is used to determine a stable set of a certain size via the amount of embedded stars (orange). In a), the node labels indicate the extreme locality constraints. In b), each node of the original network is extended to have maximum degree with respect to the substrate (blue).

with extreme locality constraints, VNE is strongly $\mathcal{NP}$-hard even when G^0 is a planar graph of maximum degree three or a triangle-free graph.

We extended Theorem 3.1 to the case without extreme locality restrictions:

Corollary 3.1. *VNE is strongly $\mathcal{NP}$-hard even without locality constraints.*

Proof. We modify the reduction in the proof of Theorem 3.1 as follows: Consider the substrate network G^0 where, for each node $i \in V^0$, we add $\Delta - |\delta(i)|$ leaf nodes connected solely to i. These additional nodes have a node capacity of $\frac{1}{\Delta}$, all edges connected to them have capacity one. This way, all nodes of the original graph have the same degree.

All the VNs are identical stars with exactly Δ leafs, with unit traffic demands, a node demand of one for the central node, and one of $\frac{1}{\Delta}$ for all the leaf nodes. See Figure 3.5 (b) for an illustration.

Because of the node demand requirements, no two virtual nodes of the same VN can be mapped onto the same physical node and no central node of a VN can be mapped onto an appended leaf. Due to the traffic requirements, no central node of a VN corresponding to a request r can be mapped next (adjacent) to a node where the central node of another request r' has been mapped. □

As far as the approximability of VNE is concerned, the reduction in the proof of Theorem 3.1 and the inapproximability result for STAB by Hastad [71] imply that:

Corollary 3.2. *Let $\epsilon > 0$. With extreme locality constraints, VNE cannot be approximated in polynomial time within a factor of $|V^0|^{1/2-\epsilon}$ unless $\mathcal{P} = \mathcal{NP}$ or within a factor of $|V^0|^{1-\epsilon}$ unless $\mathcal{ZPP} = \mathcal{NP}$.*

Proof. Consider the same reduction as in Theorem 3.1. A polynomial-time algorithm capable of certifying the existence of a VNE solution where at least $|V^0|^{1/2-\epsilon}$ or $|V^0|^{1/4-\epsilon}$ stars are embedded can also certify in polynomial-time that $G = (V, E)$ has a stability number of, at least, $|V|^{1/2-\epsilon}$ or $|V|^{1/4-\epsilon}$, since by construction, it is $|V| = |V^0|$. □

For the more general case without extreme locality, the following holds:

Corollary 3.3. *Let $\epsilon > 0$. VNE cannot be approximated in polynomial time within a factor of $|V^0|^{1/4-\epsilon}$ unless $\mathcal{P} = \mathcal{NP}$ or within a factor of $|V^0|^{1/2-\epsilon}$ unless $\mathcal{ZPP} = \mathcal{NP}$.*

Proof. Let $G = (V, E)$ be an instance of STAB. Consider the reduction in Corollary 3.1. There, G^0 is a copy of G extended with, at most, $O(|V|^2)$ extra nodes, i.e., if G is a star, exactly $(|V| - 1)^2$ extra nodes are added, yielding $|V^0| = |V| + (|V| - 1)^2 \in O(|V|^2)$. In other words, $|V^0| = O(|V|^2)$ and $|V| = O(|V^0|^{1/2})$. Since the optimal solution values of STAB on G and VNE on G^0 coincide, the existence of a polynomial time approximation algorithm for VNE with a factor of $|V^0|^{1/2-\epsilon}$ or $|V^0|^{1/4-\epsilon}$ would imply the existence of a polynomial time approximation algorithm for STAB with a factor of $|V|^{1-\epsilon}$ or $|V|^{1/2-\epsilon}$, which is impossible unless, respectively, $\mathcal{P} = \mathcal{NP}$ or $\mathcal{ZPP} = \mathcal{NP}$. □

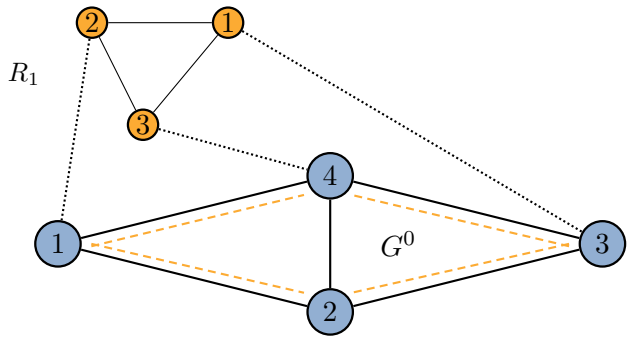


Figure 3.6: Reduction from MCMP. We consider a VNE instance with a single VN and with all unit capacities and demands. If the VNE is embedded we have found a clique minor of size three.

3.3.3 Cases with a constant dimension

We present complexity results for three special cases where one of the three dimensions of VNE, namely, the number of requests $|R|$, the size of the VNs $|V^r|$ for $r \in R$, or the size of the substrate $|V^0|$, is assumed to be a constant.

VNE with a single VN request The first result is obtained when fixing the number of requests to one. We present two alternative ways to prove $\mathcal{NP}$-hardness.

The first result relies on the graph minor containment problem and, specifically, on the *Maximum Clique Minor Problem* (MCMP). Given a graph G, MCMP calls for a subgraph G' which, after edge contraction, is isomorphic to a clique of cardinality $k \in \mathbb{Z}_+$. The $\mathcal{NP}$-hardness of MCMP is established in Eppstein [49].

Lemma 3.3. *VNE is strongly $\mathcal{NP}$-hard even when $|R| = 1$.*

Proof. Consider an MCMP instance with a given graph G and an integer $k \in \mathbb{Z}_+$. Construct the VNE instance with $G^0 = G$, with unit node and edge capacities, and a single VN request with k nodes, a complete demand matrix with unit entries and unit node demands, see Figure 3.6 for a sketch.

G contains a clique minor of size k if and only if the VN is embedded. □

For the second reduction, we rely, again, on the STAB problem.

Lemma 3.4. *VNE is strongly $\mathcal{NP}$-hard even when $|R| = 1$ (and the demand matrix of the VN is a "meta-star", i.e., a tree of depth two).*

Proof. We extend the construction we have used in Corollary 3.1. Consider an instance $G = (V, E)$ with $k \in \mathbb{Z}_+$ of STAB. For the VNE instance, we adopt G as the substrate network, again extended by adding $\Delta - |\delta(i)|$ leaf nodes connected solely to i, for each node $i \in V$, with the same capacities as described above. Additionally, we add a node m, connected to all nodes i corresponding to the node set V of G. The node m and its outgoing edges have node, respectively link capacity ϵ, for ϵ sufficiently small.

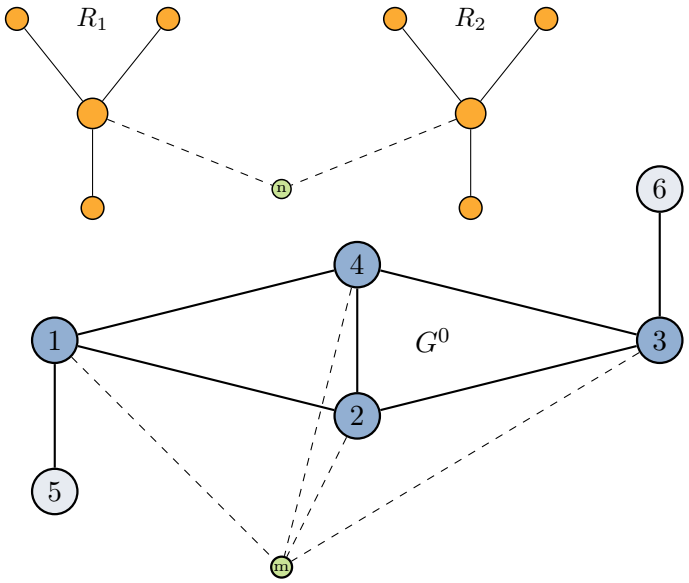


Figure 3.7: Illustration of the reduction used in Lemma 3.4. STAB is reduced to a VNE instance with a single VN. Therefore, additional connecting nodes and links (green) have been added to the substrate (blue) and to the VNs (orange).

The VN is constructed as follows. The VN consists of k identical stars with exactly Δ leafs, with unit traffic demands, a demand of one for the central node, and one of $\frac{1}{\Delta}$ for all the leaf nodes. It also contains an additional node n which is connected to all the centers of the k stars. Node n has a demand of ϵ and the centers of the stars have a traffic demand of ϵ to n. See Figure 3.7 for an illustration.

By the same arguments as above, the VNE instance admits a feasible solution of total profit one, i.e., a solution where the VN is embedded, if and only if the graph G of the STAB instance contains a stable set of cardinality at least k. □

VNE where all VNs contain a single node The second result concerns VNs consisting of a single virtual node, and relies on the strong $\mathcal{NP}$-hardness of the *Multi-Knapsack Problem* (MKP) as shown by Geng et al. [61]:

Lemma 3.5. *VNE is strongly $\mathcal{NP}$-hard even when $|V^r| = 1$ for all $r \in R$.*

Proof. Given any MKP instance with p knapsacks and q items, with knapsack capacities b_j, for $j = 1, \ldots, p$, and item weights and profits w_i and p_i, for $i = 1, \ldots, q$, it suffices to construct a VNE instance with a physical node for each knapsack j in MKP, with the same capacity b_j, and as many VN requests as there are MKP items. Let each VN request r consist of a single virtual node v with demand w_v^r equal to the weight w_i of the corresponding MKP item, and with profit p_i. The MKP instance has a solution of value k if and only if a subset of VN requests with a total profit of k can be embedded. □

Recall that MKP admits a *PTAS*, see Chekuri and Khanna [30]. Furthermore, it is well known that the standard dynamic programming algorithm for the (single) *Knapsack Problem* (KP) can be extended to the MKP. Although the algorithm becomes exponential in the number of knapsacks, it is pseudo-polynomial whenever the number of knapsacks is a constant. We will extend on that in the next subsection.

VNE where the size of the substrate is limited We start with the most restricted case where the substrate network consists of s single node.

Proposition 3.1. *VNE with $|V^0| = 1$ is weakly $\mathcal{NP}$-hard.*

Proof. The result follows by a straightforward reduction from the Knapsack Problem (KP). Since there is a single physical node, each virtual network $r \in R$ collapses into a single virtual node with node demand equal to the total node demand $\sum_{v \in V^r} w_v^r$ and with zero traffic demand. Clearly, the KP instance has a solution of value at least k if and only if a subset of VN requests with a total profit of at least k can be embedded. □

This immediately implies that the VNE problem on a single physical node can be solved in pseudo-polynomial time. The result holds regardless of the size or topology of the different VNs which, without loss of generality, can be transformed into single nodes, as there is only a single physical node. If we additionally assume that $|R| = 1$, the problem becomes trivial:

Remark 3.9. *Consider the case where $|V^0| = 1$ and $|R| = 1$. VNE can be solved in polynomial time by verifying $\sum_{v \in V^r} w_v^r \leq B_i$, for the unique physical node $i \in V^0$.*

The weak $\mathcal{NP}$-hardness is immediately lost as soon as we increase the (constant) number of nodes in the physical network to two:

Lemma 3.6. *VNE is strongly $\mathcal{NP}$-hard even when $|V^0| = 2$ (and $|R| = 1$).*

Proof. Consider the *Minimum Cut into Equal Sized Subsets* (EQUICUT) problem which, given an undirected graph $G = (V, E)$ with $|V|$ even, a weight function $c : E \to \mathbb{N}$, two nodes $s, t \in V$, and an integer $k \in \mathbb{Z}_+$, calls for a partition of V into two subsets $S, V \setminus S$ of equal size with $s \in S$ and $t \in V \setminus S$ such that $\sum_{e \in \delta(S)} c_e \leq k$.

For the reduction, we construct a VNE instance with a single VN, topologically equal to G, with unit node demands for any node in $V \setminus \{s, t\}$, a node demand of $|V|$ for s and t, edge demands c_{ij} for all $\{i, j\} \in E$, and a unit profit. Let the physical network consist of two nodes, both of capacity $|V| + \frac{|V|-2}{2}$, and let them be connected by a single edge of capacity k. By construction, s and t have to be embedded on two different nodes and the physical networks has enough node capacity at accommodate all virtual nodes.

If VNE admits a solution of value one, the embedding of the VN induces a partition of V into two sets of equal size and, as the capacity on the single physical link is not exceeded, the cut given by this partition is of weight at most k. Conversely, if EQUICUT does not admit a solution, then, in any mapping of the virtual nodes onto the two physical nodes, the total traffic demand between pairs that are not mapped onto the same physical node exceeds k. The hardness result follows due to EQUICUT being strongly $\mathcal{NP}$-hard, as shown by Garey et al. [59] by a reduction from 3-SAT. □

Table 3.1: The tableau for the dynamic program for the KP with a capacity of four and three candidate items. Note that, excluding "trivial" cells, the tableau contains $3 \cdot 4 = 12$ entries.

Items	**Capacity**				
	0	1	2	3	4
$\emptyset$	0	0	0	0	0
$\{1\}$	0	·	·	·	·
$\{1,2\}$	0	·	·	·	·
$\{1,2,3\}$	0	·	·	·	·

3.3.4 Dynamic programming approaches

In this subsection, we highlight the intrinsic knapsack-like nature of VNE. Based on this structure, we derive a dynamic programming approach for the VNE problem. In particular, we extend the dynamic program for the (multi-) KP to VNE. Therefore, we start with the following example:

Example 3.3. *The KP can be solved by dynamic programming. For the KP, the table of such program contains an entry for each each number in $0, \ldots, B$, where B is the capacity of the KP, and for all potential numbers of packed items from 0 to n, where n is the number of items of the KP. Hereby, each entry states the optimal solution value of the induced KP instance given by a part of the capacity and a subset of the items. See Table 3.1 for an example of a KP with capacity of four and three items.*

In total, $O(nB)$ values have to be computed to completely fill in this tableau and the optimal solution value is given by the right most entry at the bottom. Each entry is recursively derived of two previously computed values. That is, the optimal solution value of an entry, respectively of an induced instance, is either the optimal solution value given if that the item of the currently largest index is put into the knapsack or the optimal solution value given if that the item is not packed. In total, this yields a pseudo polynomial time algorithm.

For the MKP, the dynamic program of the KP can be extended by adding an additional dimension for every further knapsack on the MKP. Let m be the number of the knapsacks and let B be the maximal capacity of all of them. In this case, the recursion for evaluating the tableau is exponential as the tableau contains $O(nB^m)$ entries and for each entry, a maximum over $m+1$ terms has to be evaluated (enumerating the possibilities that an item is packed on any of the m knapsacks or not at all). As long as m is fixed, the algorithm is still pseudo-polynomial.

Note that the same algorithm can directly be applied to the special cases in Lemma 3.5 and in Proposition 3.1. We introduce some additional notation to extend these results:

Definition 3.2. *Let the tuple $\mathcal{I} := (V^0, E^0, \mathcal{B}, R)$ be an instance of the VNE problem, and let $\mathcal{B}$ denote the vector of "stacked" node and edge capacities of the substrate, i.e.,*

$$\mathcal{B} := (B_1, \dots, B_{|V^0|}, K_1, \dots K_{|E^0|}), \tag{3.8}$$

for some enumeration of the physical nodes and links. Let β be the maximum entry of $\mathcal{B}$. We adopt the notation $\mathcal{I} = (\mathcal{B}, R)$ if the substrate network is clear in the context.

Denote with Δ_r the set of all possible embeddings of the request $r \in R$ which are feasible for $(\mathcal{B}, \{r\})$ and denote by $c(\eta_r) \in \mathbb{R}^{|V^0|+|E^0|}$ the vector of the capacity requirements induced by the embedding $\eta_r \in \Delta^r$. We write η_r^* for an optimal embedding of r with respect to $\mathcal{I}$ and $P(\mathcal{I})$ as the thereof induced profit.

As for the KP, we can give a recursion on the optimal solution value P:

Lemma 3.7. *Let $\mathcal{I}$ be a VNE instance where $\mathcal{B}$ units of capacity are available on the substrate. For any $\tilde{R} \subseteq R$ and $r \in \tilde{R}$, the following recursion holds:*

$$P(\mathcal{B}, \tilde{R}) = \max\left\{\mathcal{P}(\mathcal{B}, \tilde{R} \setminus \{r\}), \max_{\eta_r \in \Delta^r}\left\{\mathcal{P}(\mathcal{B} - c(\eta_r), \tilde{R} \setminus \{r\}) + p_r\right\}\right\}. \tag{3.9}$$

Proof. The optimal solution of the VNE instance is given by either rejecting or embedding r. The first part of the maximization describes the first possibility. The latter part enumerates all possible embeddings of r onto the substrate. □

Recursion (3.9) can, in principle, be evaluated in the same way as the dynamic program for the (M)KP by, in the subproblem, enumerating all possible embeddings of each VN. In theory, this yields an exact solution procedure for the VNE problem. We point out some apparent weaknesses of this approach.

Similar to the MKP case, each capacity entry (for the nodes and for the links) obtains its own dimension in the tableau, such that, in total, the tableau contains $O(|R|\beta^{|V^0|+|E^0|})$ entries. This way, the algorithm is at least exponential in the size of the substrate network. Furthermore, if the substrate consist of at least two nodes, there are exponentially many possibilities to map the nodes of a VN onto the physical nodes. Even when neglecting the link-mapping where exponentially man possibilities occur as well, the number of potential node embeddings of a VN is exponential, making an enumerative approach prohibitively expensive for the general case.

In this context, we recall that just *checking* if a single VN r is embeddable is for any G^0 and G^r (any topology, variable-size) $\mathcal{NP}$-hard and inapproximable, compare Lemma 3.4. Even for any G^r (any topology, variable-size) and a fixed-size G^0, the problem is still strongly $\mathcal{NP}$-hard, see Lemma 3.6. However, there are special cases under which the dynamic program yields an pseudo-polynomial algorithm:

Remark 3.10. *VNE can be solved in pseudo-polynomial time if G^0 has fixed size and if the number of embeddings of each VN Δ^r is pseudo-polynomially bounded, respectively if the subproblem in Equation* (3.9) *is pseudo-polynomially solvable.*

If G^0 has fixed size, the dynamic program has a pseudo-polynomial-sized table. If Δ^r is pseudo-polynomially bounded for all $r \in R$, the subproblem in Equation (3.9) *can be solve by complete enumeration.*

The corollary applies for example if V^0 and $|V^r|$ are constant. Note, though, that it relies on the flows being unsplittable as, in this case, the amount of paths between two physical nodes is a constant as G^0 is of fixed size. If the flows are splittable, the subproblem is not pseudo-polynomially solvable by enumeration.

3.3.5 Star topologies

As indicated by Rost et al. [113], the special case of embedding virtual star-networks is an important topic for many applications, as for instance the virtual cluster embedding described by Ballani et al. [12]. In the referred work, Rost et al. propose a polynomial time algorithm to embed a uniform star on a general graph. Such VN is isomorphic to a star, where all traffic demands are either going to, or are coming from the center. Since the substrate network is undirected, without loss of generality we focus on the first case. In contrast to the VNE problem considered in this work, the embedding of the virtual nodes (links) induces objective costs, such that the resulting embedding has to be optimal with respect to those costs as well. In the following, we present further results on our variant of the VNE problem, where only star networks are to be embedded.

We have already observed, compare Lemma 3.1, respectively Corollary 3.1, that embedding multiple, general stars is a strongly $\mathcal{NP}$-hard case. Therefore, we restrict ourselves to the case that only a *single star* is to be embedded or to be rejected. We start with a proper definition of the subproblem.

A single star

Definition 3.3 (VNES)**.** *Let an instance of the VNE problem be given with $|R| = 1$ and let the virtual request be isomorphic to a star graph with center z and with $k \in \mathbb{Z}_+$ leaf nodes $h = 1, \dots, k$. We refer to the resulting problem as the* ***Virtual Network Embedding Problem of a Star*** *(VNES). Further, denoting $\mathcal{B}$ as the vector of stacked capacities of the substrate with maximum entry β, we write $\mathcal{I} := (V^0, E^0, \mathcal{B}, k)$ for an instance of VNES.*

As we have already observed by a reduction from the KP, see Proposition 3.1, the VNES problem is weakly $\mathcal{NP}$-hard if the substrate network consists of a single node. The same holds, if the substrate network consists of a single edge:

Lemma 3.8. *VNES is* weakly $\mathcal{NP}$*-hard, even when G^0 is a single edge.*

Proof. Consider the Partition Problem (PART): Given a set N of n items with weights $w_i \in \mathbb{Z}_+$ for all $i = 1, \dots, n$, is there a subset $S \subseteq N$ of items with weight

$$C := \left\lceil \frac{\sum_{i \in N} w_i}{2} \right\rceil ? \tag{3.10}$$

PART is weakly $\mathcal{NP}$-hard. We show a reduction to VNES.

Let the substrate network be a single edge where both nodes have capacity C and let there be a sufficiently large edge capacity between them. Let the VN be a star with n

leaves. Let the node demand be w_i for the leaf nodes $i = 1, \dots, n$, and let the node demand be zero for the central node. The edge demands are arbitrary. The PART instance is a "yes" instance if and only if the VN is embeddable. □

By this result, the VNES problem with bounded node set V^0 is at least weakly $\mathcal{NP}$-hard. In the following, we extend this result and show that this problem is at most weakly $\mathcal{NP}$-hard. For this purpose, observe that instead of solving the VNES problem, we can equivalently ask how many nodes of the VN can maximally be embedded. By introducing a profit of one for the embedding of each leaf we can give a dynamic program especially tailored to this case. We adapt Definition 3.2:

Definition 3.4. *Let the tuple $\mathcal{I} := (V^0, E^0, \mathcal{B}, k)$ be an instance of the VNES problem, where the center of the VN has already been mapped on $i_0 \in V^0$ and where k leaves remain to be embedded.*

Let $\mathcal{B}$ denote the vector of "stacked" node and edge capacities of the substrate, i.e.,

$$\mathcal{B} := (B_1, \dots, B_{|V^0|}, K_1, \dots K_{|E^0|}), \tag{3.11}$$

for some enumeration of the physical nodes and links. Let β be the maximum entry of $\mathcal{B}$. We adopt the notation $\mathcal{I} = (\mathcal{B}, n)$ if the substrate network is clear in the context.

Denote by Δ^h the set of all possible embeddings of the h^{th} leaf, $h = 1, \dots, k$ which are feasible for $(\mathcal{B}, h)$ and denote by $c(\eta_h)$ the vector of the capacity requirements induced by the embedding $\eta_h \in \Delta^h$. An optimal embedding of h with respect to $\mathcal{I}$ is denoted as η_h^*. We write η_r^* for an optimal embedding of r with respect to $\mathcal{I}$ and $P(\mathcal{I})$ as the thereof induced profit.

The following lemma gives a recursion for the embedding of the star:

Lemma 3.9. *Let $\mathcal{I}$ be a VNES instance where $\mathcal{B}$ units of capacity are available on the substrate. Let the central node z be mapped onto node $i \in V^0$. For $h = 1, \dots, k$ it is*

$$\begin{aligned} &\mathcal{P}(V^0, E^0, \mathcal{B}, h) \\ &= \max \left\{ \mathcal{P}(V^0, E^0, \mathcal{B}, h-1), \max_{\eta_h \in \Delta^h} \{\mathcal{P}(\mathcal{B} - c(\eta_h), h-1) + 1\} \right\}. \end{aligned} \tag{3.12}$$

Proof. An optimal solution of the VNES instance, with respect to the profit obtained when embedding the leaves, is given by either rejecting or embedding the h^{th} leaf. The first part of the maximization describes the first possibility. The latter parts enumerates all possible embeddings of the h^{th} leaf onto the substrate. □

The recursion requires the central node to be embedded beforehand, such that, if it is employed to solve the VNES problem, it has to be evaluated for all $|V^0|$ potential mappings of the center. Similar as in the previous subsection, if evaluated by dynamic programming, the tableau requires $O(k\beta^{|V^0|+|E^0|})$ entries, such that the algorithm is exponential, even when ignoring the complexity of the subproblem.

However, since we assume an unsplittable routing scheme, we can rewrite Recursion (3.12). Therefore, for node $i \in V^0$, denote by P_i all paths in G^0 from node i_0 to

node i. For $i \in V^0$, $p \in P_i$ and $h = 1, \ldots, k$, denote by $(w_i, p)_h$ the vector of the capacity requirements induced by the embedding the h^{th} leaf of the VN on the physical node i and routing the traffic to the center on path p. Then, the inner maximization term of Recursion (3.12) can be replaced by

$$\max_{\substack{i \in V^0 \\ p \in P_i}} \left\{ \mathcal{P}(V^0, E^0, \mathcal{B} - (w_i, p)_h, h-1) + 1 \right\}. \tag{3.13}$$

By limiting the size of the substrate network, we obtain:

Corollary 3.4. *VNES can be solved in pseudo-polynomial time if V^0 has fixed size.*

Proof. The dynamic program is evaluated $|V^0|$ times, once for every possible embedding of the center. The dynamic programming table contains $O(k\beta^{|V^0|+|E^0|})$ entries. For each entry, the maximum in Recursion (3.12) as to be evaluated. Since V^0 has fixed size, the number of terms, i.e., the number of different embeddings of a leaf with respect to V^0, over which the maximum is determined is bounded by a constant. □

In the dynamic program, the subproblem is (pseudo-) polynomial even for non-fixed-size G^0 if it has a (pseudo-) polynomial (or constant) number of paths, e.g., if G^0 is an edge, a path, a tree, a cycle, or a cactus graph with fixed-size cycle decomposition. In contrast to the dynamic program proposed in Lemma 3.7, the size of the virtual network does not need to be restricted.

Basically the same program can be employed to tackle the problem of deciding whether k stars can be embedded onto the substrate, i.e., whether all leafs of all the k stars can be embedded or not. Therefore, all combinations of the potential embeddings the centers of the k stars have to be enumerated and thus, the algorithms becomes exponential in the number of VNs. Again, if the substrate and the number of stars is limited, the algorithm becomes pseudo-polynomial.

However, if the size of the substrate is not restricted, VNES is strongly $\mathcal{NP}$-hard:

Lemma 3.10. *The VNES problem is strongly $\mathcal{NP}$-hard.*

Proof. Consider the 3-Partition Problem (3-PART): Given a set S of $n = 3m$ positive integers $k_1, \ldots, k_n$ and an integer B where $\sum_{i \in S} k_i = mB$, can S be partitioned into m triplets $S_1, S_2, \ldots, S_m$ such that

$$\sum_{k \in S_j} k = B \quad \forall\, j = 1, \ldots, m? \tag{3.14}$$

3-PART is strongly $\mathcal{NP}$-hard, see Garey et al. [59]. We show a reduction to VNES:

Let the substrate network consist of m nodes of capacity B, interconnected with sufficiently large edge capacities. Let the VN be a star with n leaves. Let the node demand be $\omega_i^1 = k_i$ for the leaf nodes $i = 1, \ldots, n$, and let the node demand be zero for the central node. The edge demands are arbitrary. The 3-PART instance is a "yes" instance if and only if the VN is embeddable. □

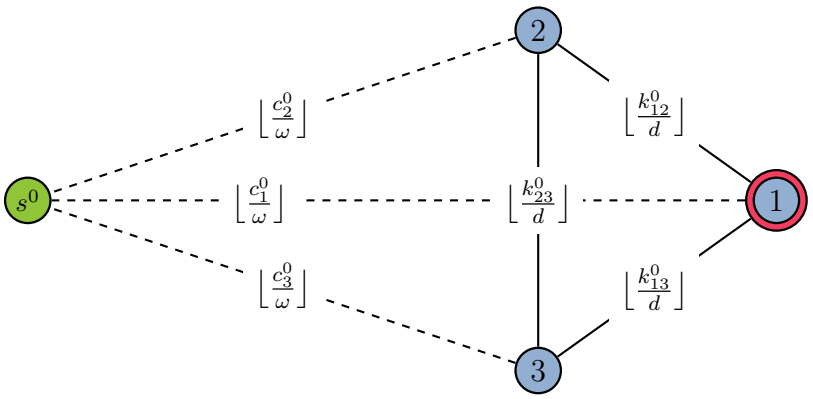


Figure 3.8: Consider a VNES-U instance with a three node substrate network (blue) and assume that the center of the star is already mapped on node 1 (red). The (modified) edge capacities are indicated. VNES-U has a feasible solution if and only if a s^0 (green) to 1 (red) flow exists of value equal to the amount of the leafs of the star.

Uniform stars When restricting the VNES problem to the embedding of a uniform star, we obtain the result by Rost et al. [113], that is, a polynomially solvable case.

Definition 3.5 (VNES-U)**.** *Let an instance of the VNES problem be given and let the VN be isomorphic to a star graph with center z and $k \in \mathbb{N}$ with leaf nodes $h = 1, \dots, k$. We refer to the resulting VNE problem as the* ***Virtual Network Embedding Problem of an Uniform Star*** *(VNES-U) if the demand values are uniform, i.e., if for fixed $\omega, d \in \mathbb{Z}_+$, it is $\omega_v^1 = \omega$ and $d_{vz}^1 = d$ for all $v = 1, \dots, k$.*

As is shown by Rost et al. [113], VNES-U can be solved by a polynomial time algorithm, if no locality conditions are present:

Lemma 3.11 (Rost et al. [113])**.** *VNES-U is in $\mathcal{P}$ if no locality conditions are present.*

In this work, the authors show that a uniform star can be embedded in a substrate network G^0 of any topology and capacities in polynomial time. The algorithm works as follows: We re-define the edge capacities as

$$k_{ij}^0 := \left\lfloor \frac{k_{ij}^0}{d} \right\rfloor \qquad \forall ij \in E^0 \tag{3.15}$$

and add a node s^0 to the substrate, connected to all nodes $i \in V^0$ with a capacity of

$$k_{s^0,i}^0 := \left\lfloor \frac{c_i^0}{\omega} \right\rfloor. \tag{3.16}$$

Assuming that the center is mapped on some $i \in V^0$, the algorithm looks for a flow of value k (or for a minimum cost flow of that value) from s^0 to i. See Figure 3.8 for a

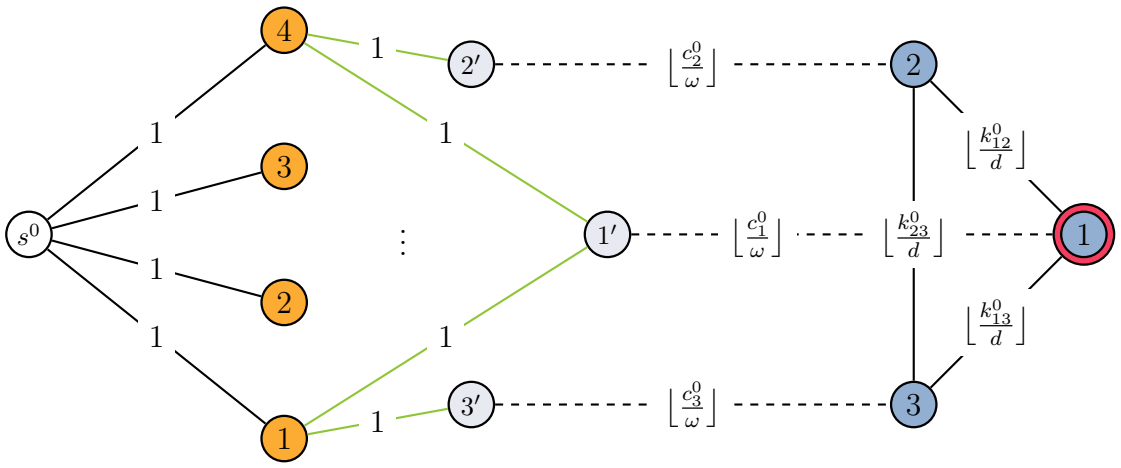


Figure 3.9: Consider a VNES-U instance with a three node substrate network (blue), where a star with four leafs (orange) is to be embedded and assume that the center of the star is already mapped on node 1 (red). The modified edge capacities are shown, the locality constraints are modeled with the green edges, e.g., the leaf 1 (orange) may not be mapped on the substrate node 2. VNES-U has a feasible solution if a $s^0 - 1$ flow of value four exists.

sketch of the construction. A solution is found by applying the algorithm $|V^0|$ times, once per possible mapping of the center. Note that, because of the capacities being integral, the resulting flow will be integer. Since we have normalized all capacities with respect to one unit of traffic, respectively node, demand, the flows from the different leafs are not split. Hence, the resulting maximum flow yields an embedding of the VN with unsplittable routing.

We can extend this result to include locality conditions as follows:

Corollary 3.5. *VNES-U is in $\mathcal{P}$.*

Proof. We consider the same construction as for Lemma 3.11, extended as follows: The edge-capacities k_{ij}^0 of the substrate are redefined as described above. We copy the nodes of V^0 and add them again to the substrate network. We call these new nodes $(V^0)'$. Each copied node $i \in (V^0)'$ is connected to exactly one corresponding original node $j \in V^0$ with edge capacity

$$k_{ij}^0 := \left\lfloor \frac{c_i^0}{\omega} \right\rfloor. \tag{3.17}$$

We further add k nodes, and connect each of them to $i \in (V^0)'$ with capacity one if $i \in V^0(r,k)$, i.e., if the k^{th} leaf of the star was allowed to be embedded on the to i corresponding node $j \in V^0$. Denote the set of this nodes as K. Finally, we add a source node and connect it to all nodes $k \in K$ with capacity one.

Assuming that the center is mapped on some $\bar{\imath} \in V^0$, the algorithm looks for a flow of value k (or for a minimum cost flow of that value) from s^0 to $\bar{\imath}$. See Figure 3.9 for

Table 3.2: An overview on the complexity results on the VNE problem as presented in Section 3.3. The Column "R./A." denotes which problem (reduction), respectively which algorithm, is used to derive the result.

Variant	**Complexity**	**R./A.**
General	strong $\mathcal{NP}$-hard	STAB
Subproblem phase I	strong $\mathcal{NP}$-hard	MKP
Subproblem phase II	strong $\mathcal{NP}$-hard	EDPP
$\lvert R\rvert = 1$	strong $\mathcal{NP}$-hard	STAB
$\lvert V^r\rvert = 1 \forall r \in R$	strong $\mathcal{NP}$-hard	MKP
$\lvert V^0\rvert = 1$	weak $\mathcal{NP}$-hard	KP
$\lvert V^0\rvert = 2$	strong $\mathcal{NP}$-hard	EQUICUT
$\lvert V^0\rvert$ const. & $\lvert V^r\rvert$ const. $\forall r \in R$	weakly $\mathcal{NP}$-hard	Dyn. Prog.
VNE with mult. stars	strong $\mathcal{NP}$-hard	STAB
VNES	strong $\mathcal{NP}$-hard	3-PART
VNES & $\lvert V^0\rvert$ const.	weak $\mathcal{NP}$-hard	Dyn. Prog.
VNES-U	polynomial	Max. Flow.

a sketch of the construction. A solution is found by applying the algorithm $|V^0|$ times, once per possible mapping of the center. □

We conclude this section by an overview on our results in Table 3.2.

3.4 Addressing the case of data uncertainty

In this section, we focus on the VNE problem subject to data uncertainty. In many practical applications, it is reasonable to assume that, for each VN, the actual demand for computing resources and traffic may vary, often substantially, over time. For instance, an online gaming or a movie streaming service may have more or less customers, and therefore, a different resource consumption, depending on its popularity, which clearly changes over time (with, e.g., peaks for new content releases, after an advertising campaign, and so on). This poses a problem from a network reliability point of view, as it can lead to traffic congestion, quality of service degradations, or, even, service disruptions.

In this regard, we assume that some parameters of the VNE problem, namely the demand values ω_v^r and d_{vw}^r, are not known *a priori*. Consequently, any solution to the VNE problem has to be found without exact knowledge of these parameters. As indicated in Section 1.4, classical approaches to account for data uncertainty typically consider a so-called *worst case* setting, so to guarantee that the network will be operational even for peak values of traffic. Although guaranteeing feasibility, this practice comes at an often unnecessary cost as, in many cases, it is very unlikely for every demand in every VN to

simultaneously be at its peak. Indeed, in a number of practical cases, it is reasonable to assume that the probability that *all* demands simultaneously reach their peak values is fairly small. This is reasonable, in our example, when assuming that new content releases and advertising campaign do not take place for *all* services at the same time.

Consequently, the idea is to look for a solution where the different VNs are provisioned for demands which are smaller than their peak values, at the same time guaranteeing that the substrate network has sufficient capacity for *almost all* the traffic configurations, only neglecting a few unlikely cases. This way, we are likely to obtain more profitable solutions where more VN requests are embedded, thus avoiding costly issues of over-provisioning. For this purpose, we introduce a chance-constrained version of the VNE problem in the next subsection, from which we derive a Γ-robust MILP formulation in the following. We conclude by presenting heuristics for the Γ-robust problem.

3.4.1 A chance-constrained MILP formulation

One natural way of taking demand uncertainties into account is of interpreting ω_v^r and d_{vw}^r not as constants, but as (bounded) random variables and requiring that each constraint holds with a certain probability. In the reminder of this work, we will assume that, for any $r \in R$, each uncertain node demand ω_v^r is an independent random variable taking value in the symmetric interval $[\bar{\omega}_v^r - \hat{\omega}_v^r, \bar{\omega}_v^r + \hat{\omega}_v^r]$, with *nominal* value $\bar{\omega}_v^r$ and *maximum deviation* $\hat{\omega}_v^r$. Similarly, we assume that each uncertain link demand $d_{vw}^r \in D^r$ takes values in a symmetric interval, i.e., we assume that $d_{vw}^r \in [\bar{d}_{vw}^r - \hat{d}_{vw}^r, \bar{d}_{vw}^r + \hat{d}_{vw}^r]$, centered around the nominal value $\bar{d}_{vw}^r$, with a maximum deviation $\hat{d}_{vw}^r$.

In this context, let $\epsilon \in [0, 1]$ be a given probability with which each constraint is required to be satisfied. Formalizing this for the Constraints (3.2c) and (3.2d), we obtain the following chance-constrained MILP formulation of VNE:

$$\max \quad \sum_{r \in R} p^r y^r \tag{3.18a}$$

$$\text{s.t.} \quad \Pr\left(\sum_{r \in R} \sum_{\substack{v \in V^r: \\ i \in V^0(r,v)}} \omega_v^r x_{vi}^r \leq c_i^0 \right) \geq \epsilon \qquad \forall i \in V^0 \tag{3.18b}$$

$$\Pr\left(\sum_{r \in R} \sum_{v,w \in V^r} d_{vw}^r f_{ij}^{vw,r} \leq k_{ij}^0 \right) \geq \epsilon \qquad \forall (i,j) \in A^0 \tag{3.18c}$$

$$(3.2b), (3.2e), (3.2f). \tag{3.18d}$$

Note that, if the deterministic Formulation (3.2a)–(3.2f) is solved with *worst case* data, i.e., by setting each random variable to its maximum value, we obtain an embedding which is also a feasible solution to the chance-constrained Formulation (3.18a)–(3.18d) when solved with any ϵ, and an optimal one for $\epsilon = 1$ (assuming that the worst case can occur). As we will see with our computational experiments, which we report in Section 3.5, the objective function value of such solutions is typically very poor.

Clearly, Formulation (3.2a)–(3.2f) can also be solved with *nominal* data, substituting for each random variable its nominal value. Although typically yielding much larger objective function values, this choice is in practice only feasible for the setting $\epsilon = 0$.

3.4.2 The Γ-robust VNE problem

For most non-trivial probability distributions of the corresponding random variables, chance-constrained problems are, in general, very hard to solve. This is because, to rely on mathematical programming tools, they require a closed-form solution to the integrals corresponding to each probabilistic constraint the derivation of which is, usually, not known. An attractive way to circumvent this drawback, also successfully applied to a number of networking problems as, e.g., by Koster et al. [85], is of recurring to robust optimization and, specifically, to a Γ-robust approach.

As lined out in Section 1.4, the Γ-robustness model by Bertsimas and Sim [17, 18]assumes that, in any possible realization of the uncertain data (i.e., of the random variables of the chance-constrained model), at most Γ coefficients will simultaneously deviate from their nominal value. This model naturally meets the features of VNE if we assume that the number of demands simultaneously reaching their peak values is bounded by Γ. That is, Γ-robust solutions are guaranteed to be feasible for any realization of the uncertain coefficients with at most Γ deviations. Thus, the Γ-robust model allows to approximate the chance-constrained formulation for an arbitrary ϵ by selecting a suitable value for Γ. Most interestingly, the Γ-robust model leads to problems which are computationally much more tractable than those involving chance constraints, as we will show in the following.

Let us now show how to derive a Γ-robust MILP formulation for VNE. We will address the case where, for each node and link, at most Γ demands deviate from their nominal value. We recall that, for $\Gamma = 0$, the robust problem corresponds to the original problem with nominal data while, for $\Gamma = \infty$, it corresponds to the original problem with worst case data where all the coefficients simultaneously deviate to their maximum value. For the remainder of the section, assume that $\Gamma \in \mathbb{Z}_+$ is given.

Γ-robust MILP formulation for VNE I: node demands For convenience, define $\mathcal{V}_i^N := \{(r,v) \in R \times \cup_{r \in R} V^r : i \in V^0(r,v)\}$. For any physical node $i \in V^0$, the set $\mathcal{V}_i^N$ corresponds to all the request-node pairs (r,v) where the virtual node $v \in V^r$ can be mapped to the physical node i. The Γ-robust counterpart to Constraint (3.18b) is:

$$\underbrace{\sum_{(r,v)\in\mathcal{V}_i^N} \bar{\omega}_v^r x_{vi}^r}_{\text{nominal LHS}} + \underbrace{\max_{\substack{T\subseteq\mathcal{V}_i^N \\ |T|\leq\Gamma}} \sum_{(r,v)\in T} \hat{\omega}_v^r x_{vi}^r}_{\text{maximum deviation}} \leq c_i^0 \qquad \forall\, i \in V^0. \tag{3.19}$$

The constraint accounts for the scenario where the Γ coefficients with the largest value of $\hat{\omega}_v^r x_{vi}^r$ (those in the set T) simultaneously deviate. If the constraint is satisfied, the

nominal constraint will then be satisfied for any realization in the uncertainty set. We apply Theorem 1.5 to obtain a reformulation by linear constraints:

Corollary 3.6. *Let $\Gamma \in \mathbb{Z}_+$. Let $\pi_i \geq 0$ for all $i \in V^0$, and let $\rho_i^{rv} \geq 0$, for all $i \in V^0, r \in R, v \in V^r$. Constraint* (3.19) *can be substituted by*

$$\sum_{(r,v)\in\mathcal{V}_i^N} (\bar{\omega}_v^r x_{vi}^r + \rho_i^{rv}) + \Gamma\pi_i \leq c_i^0 \qquad \forall\, i \in V^0 \tag{3.20a}$$

$$\pi_i + \rho_i^{rv} \geq \hat{\omega}_v^r x_{vi}^r \qquad \forall\, i \in V^0, r \in R, v \in V^r : i \in V^0(r,v) \tag{3.20b}$$

$$x_{vi}^r, \pi_i, \rho_i^{rv} \geq 0. \tag{3.20c}$$

Proof. Consider (1.25a)–(1.25d). □

Γ-robust MILP formulation for VNE II: traffic (link) demands Denote all triples of VNs $r \in R$ and pairs of their virtual nodes $v, w \in V^r$ by

$$\mathcal{V}^L := \{(r,v,w) : r \in R, v, w \in V^r\}. \tag{3.21}$$

The Γ-robust counterpart to Constraint (3.18c), for all $(i,j) \in A^0$, reads:

$$\underbrace{\sum_{(r,v,w)\in\mathcal{V}^L} \bar{d}_{vw}^r f_{ij}^{vw,r}}_{\text{nominal LHS}} + \underbrace{\max_{\substack{T\subseteq\mathcal{V}^L \\ |T|\leq\Gamma}} \sum_{(r,v,w)\in T} \hat{d}_{vw}^r f_{ij}^{vw,r}}_{\text{maximum deviation}} \leq k_{ij}^0 \qquad \forall\,(i,j) \in A^0. \tag{3.22}$$

Similarly to the node case, a linear reformulation can be obtained by Theorem 1.5:

Corollary 3.7. *Let $\Gamma \in \mathbb{Z}_+$. Let $\pi_{ij}, \rho_{ij}^{vw,r} \geq 0$ for all $r \in R, v, w \in V^r$ and for all $(i,j) \in A^0$. Constraint* (3.19) *can be substituted by*

$$\sum_{(r,v,w)\in\mathcal{V}^L} (\bar{d}_{vw}^r f_{ij}^{vw,r} + \rho_{ij}^{vw,r}) + \Gamma\pi_{ij} \leq k_{ij}^0 \qquad \forall\,(i,j) \in A^0 \tag{3.23a}$$

$$\pi_{ij} + \rho_{ij}^{vw,r} \geq \hat{d}_{vw}^r f_{ij}^{vw,r} \qquad \forall (i,j) \in A^0, r \in R, v, w \in V^r \tag{3.23b}$$

$$f_{ij}^{r,vw}, \pi_{ij}, \rho_{ij}^{r,vw} \geq 0. \tag{3.23c}$$

Proof. Consider (1.25a)–(1.25d). □

Combining both results, we obtain a Γ-robust MILP formulation for the VNE problem:

Remark 3.11. *A Γ-robust formulation for VNE is obtained from* (3.2a)–(3.2f) *by substituting for Constraint* (3.2c) *and Constraint* (3.2d) *their robust counterparts:*

$$\max \quad \sum_{r\in R} p^r y^r \tag{3.24a}$$

$$\text{s.t.} \quad \sum_{i\in V^0(r,v)} x^r_{vi} = y^r \qquad \forall r \in R, v \in V^r \tag{3.24b}$$

$$\sum_{(r,v)\in\mathcal{V}^N_i} (\bar{\omega}^r_v x^r_{vi} + \rho^{rv}_i) + \Gamma\pi_i \le c^0_i \qquad \forall\, i \in V^0 \tag{3.24c}$$

$$\pi_i + \rho^{rv}_i \ge \hat{\omega}^r_v x^r_{vi} \qquad \begin{array}{l}\forall\, i \in V^0,\\ \forall\, r \in R, v \in V^r : i \in V^0(r,v)\end{array} \tag{3.24d}$$

$$\sum_{ij\in\delta^+(i)} f^{vw,r}_{ij} - \sum_{ji\in\delta^-(i)} f^{vw,r}_{ji} = x^r_{vi} - x^r_{wi} \qquad \forall r \in R, v, w \in V^r, i \in V^0 \tag{3.24e}$$

$$\sum_{(r,v,w)\in\mathcal{V}^L} (\bar{d}^r_{vw} f^{vw,r}_{ij} + \rho^{vw,r}_{ij}) + \Gamma\pi_{ij} \le k^0_{ij} \qquad \forall\, (i,j) \in A^0 \tag{3.24f}$$

$$\pi_{ij} + \rho^{vw,r}_{ij} \ge \hat{d}^r_{vw} f^{vw,r}_{ij} \qquad \forall (i,j) \in A^0, r \in R, v, w \in V^r \tag{3.24g}$$

$$y^r, x^r_{vi}, f^{vw,r}_{ij} \in \{0,1\} \tag{3.24h}$$

$$\pi_i, \rho^{rv}_i, \pi_{ij}, \rho^{r,vw}_{ij} \ge 0. \tag{3.24i}$$

3.4.3 Heuristics for the Γ-robust VNE problem

Although much more tractable than its original chance-constrained version (as it "only" requires the solution of an MILP), the Γ-robust version of VNE is still, as we will see in Section 3.5, very hard to solve for large instances within a reasonable computing time. Hence, in this subsection we propose two heuristics approaches to produce good-quality, robust solutions at a smaller computational effort. Both approaches rely on splitting the VNE problem into the robust counterparts of the two subproblems mentioned in Section 3.2, which are then solved sequentially.

A two-phase heuristic

First, let us outline our two-phase method. In the first phase, we carry out admission control and Γ-robust node embedding, but neglect link mapping and link capacities. In the second phase, we complete the partial solution found in phase one by searching for a Γ-robust link mapping for the accepted VNs (assuming that their node mapping is fixed to the one found in the first phase), while still allowing for VN rejections. This way, the heuristic is the natural extension to the robust case of the heuristic approach mentioned in Subsection 3.2.3. In this sense, due to entailing the solution of two MILPs at each iteration, our algorithm could be classified as a *mathheuristic*.

Phase one subproblem In the first phase, we consider the first phase problem, see Remark 3.4, and replace the node capacity constraints with their robust counterparts:

Remark 3.12. *Substituting Constraint* (3.5b) *with Constraints* (3.20a) *and* (3.20b)*, the* Γ*-robust first phase problem, i.e., the* Γ*-robust node embedding subproblem with admission control, amounts to*

$$\max \quad \sum_{r\in R} p^r y^r \tag{3.25a}$$

$$\text{s.t.} \quad \sum_{i\in V^0(r,v)} x^r_{vi} = y^r \qquad \forall r \in R, v \in V^r \tag{3.25b}$$

$$\sum_{(r,v)\in\mathcal{V}^N_i} (\bar{\omega}^r_v x^r_{vi} + \rho^{rv}_i) + \Gamma\pi_i \leq c^0_i \qquad \forall\, i \in V^0 \tag{3.25c}$$

$$\pi_i + \rho^{rv}_i \geq \hat{\omega}^r_v x^r_{vi} \qquad \begin{array}{l}\forall\, i \in V^0,\\ \forall\, r \in R, v \in V^r : i \in V^0(r,v)\end{array} \tag{3.25d}$$

$$y^r, x^r_{vi}, \pi_i, \rho^{rv}_i \geq 0. \geq 0. \tag{3.25e}$$

This subproblem is the Γ-robust counterpart for the MKP-G problem (Subsection 3.2.3). While we have shown that MKP-G is strongly $\mathcal{NP}$-hard, it is fairly easier to solve than the whole Γ-robust VNE problem, as we will see in Section 3.5. By construction, optimal solutions to this problem provide upper bounds to the Γ-robust version of VNE.

Phase two subproblem In the second phase, we consider the second phase problem, see Remark 3.6, and replace the link capacity constraints with their robust counterparts:

Remark 3.13. *Substituting Constraint* (3.7b) *with Constraints* (3.23a) *and* (3.23b)*, the* Γ*-robust second phase problem, i.e., the* Γ*-robust link embedding subproblem with admission control, amounts to*

$$\max \quad \sum_{r\in s} p^r y^r \tag{3.26a}$$

$$\text{s.t.} \quad \sum_{ij\in\delta^+(i)} f^{vw,r}_{ij} - \sum_{ji\in\delta^-(i)} f^{vw,r}_{ji} = \begin{cases} y^r & \text{if } (x^r_{vi})^* = 1 \neq (x^r_{wi})^* \\ -y^r & \text{if } (x^r_{wi})^* = 1 \neq (x^r_{vi})^* \\ 0 & \text{else} \end{cases} \begin{array}{l}\forall r \in R,\\ i \in V^0,\\ vw \in A^r\end{array} \tag{3.26b}$$

$$\sum_{(r,v,w)\in\mathcal{V}^L} (\bar{d}^r_{vw} f^{vw,r}_{ij} + \rho^{vw,r}_{ij}) + \Gamma\pi_{ij} \leq k^0_{ij} \qquad \forall\, (i,j) \in A^0 \tag{3.26c}$$

$$\pi_{ij} + \rho^{vw,r}_{ij} \geq \hat{d}^r_{vw} f^{vw,r}_{ij} \qquad \forall (i,j) \in A^0, r \in R, v, w \in V^r \tag{3.26d}$$

$$y^r, f^{r,vw}_{ij}, \pi_{ij}, \rho^{vw,r}_{ij} \geq 0. \tag{3.26e}$$

This problem is the Γ-robust counterpart of the UMCF-AC problem (Subsection 3.2.3). In spite of its strong $\mathcal{NP}$-hardness, as we will see in Section 3.5, this problem will be much easier to solve, for all instances that we will consider, than MKP-G.

By solving the phase one and phase two subproblems in sequence with the same value of Γ, the heuristic that we have introduced always provides a lower bound, i.e., a feasible solution to the Γ-robust VNE problem.

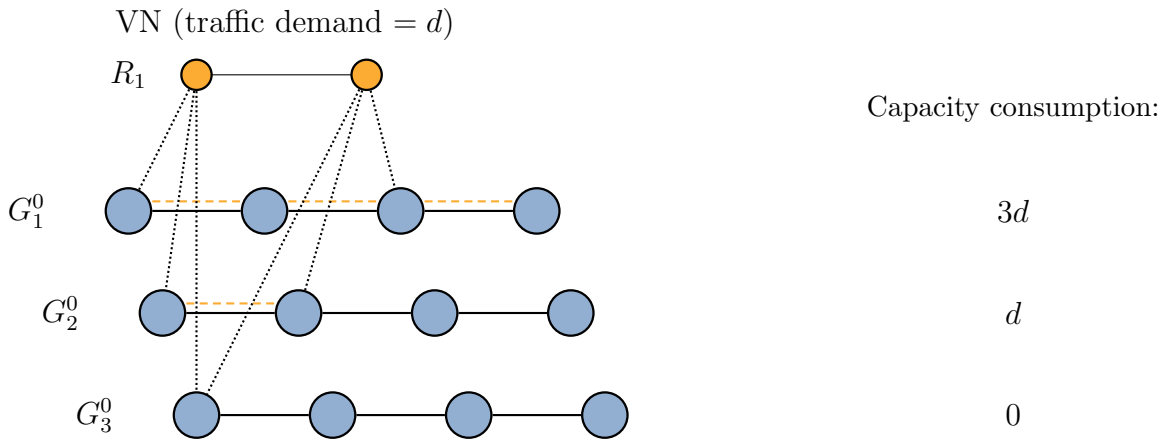


Figure 3.10: Three different embeddings of a single VN requiring a different amount of physical link capacity. Note how, in case of co-location, the flow vanishes as the source node and the sink node coincide.

Revised phase one subproblem Preliminary experiments have shown that, in many cases, more than 50% of the requests accepted in the first phase are then discarded in the second phase. This is a consequence of the fact that the phase one subproblem is oblivious to the routing aspect. Among its feasible solutions, we would indeed prefer one where pairs of virtual nodes sharing a traffic demand are mapped to physical nodes which are as close as possible. This is because, the more links are used in the routing, the higher the consumption of link capacity in the substrate network will be. A sketch of this simple observation can be found in Figure 3.10.

To circumvent this drawback, we restrict the feasible region of the first phase subproblem to solutions where pairs of virtual nodes sharing a traffic demand are mapped to physical nodes that are not too far away from each other, thus, hopefully, reducing the number of rejections in phase two. For this purpose, we cluster the virtual node pairs $v, w \in V^r$ of any VN request of index $r \in R$ into a set $\mathcal{C}$ of categories, depending on their traffic demand values d^r_{vw}. For each category $C \in \mathcal{C}$, we introduce a parameter z_C which describes the maximum distance, in terms of number of links, that we allow between the physical nodes onto which v and w can be mapped.

More formally, for any two physical nodes $i, j \in V^0$, let $\sigma(i, j)$ be the number of hops between them. We partition the different pairs of virtual nodes into three categories, based on the magnitude of their traffic demands: L (for *low*), M (for *medium*), and H (for *high*). Given the three corresponding parameters $z_L, z_M, z_H \in \mathbb{Z}_+$, we introduce the following *distance-bounding* constraints to the phase one problem:

$$x^r_{vi} + x^r_{wj} \leq 1 \quad \forall\, r \in R, v, w \in V^r, i, j \in V^0 : \begin{cases} \sigma(i,j) > z_L \wedge \{v, w\} \in L \\ \sigma(i,j) > z_M \wedge \{v, w\} \in M \\ \sigma(i,j) > z_H \wedge \{v, w\} \in H. \end{cases} \tag{3.27}$$

For a brief description of the robust, revised two-phase heuristic, we refer to Algorithm 3.3. The results obtained with this heuristic will be discussed in Section 3.5.

Algorithm 3.3 VNE: Γ-robust & revised two-phase heuristic

Input: Categories L, M, H; Parameters z_L, z_M, z_H
Add Constraint (3.27) to the Phase I problem
Solve the Γ-robust phase I subproblem via the MILP (3.25a)–(3.25e), (3.27)
Let $\tilde{R} := \{r \in R : (y^r)^* = 1\}$
Replace $(x^r_{vi})^* - (x^r_{wi})^*$ by y^r appropriately
Solve the Γ-robust phase II subproblem via the MILP (3.26a)–(3.26e)

An adaptive heuristic

While the two-phase heuristic provides good quality solutions in a short amount of computing time, see Section 3.5, its success heavily depends on a good choice of its input parameters z_L, z_M, z_H. To avoid the need for finding a suitable choice of such parameters *a priori*, we propose an adaptive algorithm in which a suitable parameter setting is chosen automatically.

The method solves a sequence of Γ-robust phase one and phase two subproblems. If, at any iteration, the phase two subproblem terminates and accepts all the requests that were accepted in phase one, it halts. If not, it looks for a virtual link which, if its VN was accepted, would consume the largest quantity of link capacity and adds a constraint similar to Constraint (3.27) to the phase one subproblem to prevent the mapping of this virtual link. Thus, the added constraint reduces the link capacity consumption in the next iteration. The algorithm iterates until a maximum alloed time is reached.

For any virtual node v, denote by $i(v) \in V^0$ the physical node to which v has been mapped in the Phase I subproblem. At each iteration and for each request $r \in R$, we associate to each pair of virtual nodes $\{v, w\}$, with $v, w \in V^r$, a value equal to the product between their traffic demand and the distance $\sigma(i(v), i(w))$ between the corresponding physical nodes $i(v), i(w)$ in terms of number of links. Then, for each request of index $r \in R$, we identify the virtual node pair which induced the largest capacity consumption:

$$(v', w') := \underset{v,w \in V^r}{\operatorname{argmax}} \Big\{ \max\{d^r_{vw}, d^r_{wv}\} \cdot \sigma(i(v), i(w)) \Big\}. \tag{3.28}$$

Due to employing the shortest path measure in term of number of links, the expression which is maximized corresponds to the *minimum* physical resource consumption that would correspond to any pair of virtual nodes embedded as defined in the first phase. Then, to impose a mapping to a closer pair of physical nodes, we add to the first phase problem, for the current triple (r, v', w'), the constraints:

$$x^r_{v'i} + x^r_{w'j} \leq 1 \quad \forall i, j \in V^0 : \sigma(i, j) > \left\lceil \frac{\sigma(i(v'), i(w'))}{2} \right\rceil \tag{3.29a}$$

Algorithm 3.4 VNE: adaptive heuristic

```
while time limit not reached do
    Solve Phase I subproblem via the MILP (3.25a)–(3.25e)
    Let R̃ := {r ∈ R : y^r = 1}
    Replace (x^r_vi)* − (x^r_wi)* by y^r appropriately
    Solve Phase II subproblem via the MILP (3.26a)–(3.26e)
    if y^r = 1 ∀r ∈ R̃ then
        terminate
    end if
    for r ∈ R do
        (v', w') := argmax_{v,w∈V^r} { max{d^r_vw, d^r_wv} · σ(i(v), i(w)) }
        for i, j ∈ V^0 do
            if σ(i(v'), i(w')) > 4 then
                Add Constraint (3.29a) to the Phase I problem
            else
                Add Constraint (3.29b) to the Phase I problem
            end if
        end for
    end for
end while
```

if $\sigma(i(v'), i(w')) > 4$, and, if $\sigma(i(v'), i(w')) \leq 4$, the constraints:

$$x^r_{v'i} + x^r_{w'j} \leq 1 \quad \forall i, j \in V^0 : \sigma(i, j) > \sigma(i(v'), i(w')) - 1. \tag{3.29b}$$

We use two different classes of constraints to prevent a too fast decline in the allowed distance for an embedding of a virtual link. The pseudo-code for the adaptive heuristic is reported in Algorithm 3.4.

3.5 Computational studies

In this section, we present computational experiments on the VNE problem. In particular, we focus on the VNE problem with data uncertainty, i.e., on an evaluation of the robust problem. During our computations, we consider the VNE problem on a directed substrate.

We start with an introduction of the dataset in Subsection 3.5.1. In Subsection 3.5.2, we briefly discuss the *deterministic* problem, i.e., the case where all data is certain. For example, we sketch how such problem can be tackled by exact and heuristic approaches, postponing a more in-depth discussion of well suited parameter choices of the heuristics and generally, a more detailed discussion of the results, to the robust case.

In Subsection 3.5.3, we consider the Γ-robust VNE problem. We extensively discuss parameter settings for the two-phase heuristic and compare the heuristic and the exact

Table 3.3: Characteristics of the substrate networks.

	Name	$\|V^0\|$	$\|E^0\|$	Request Set
SNDLIB	Abilene	12	30	$\{5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 20, 24, 28, 32\}$
	Atlanta	15	44	
	Nobel-us	14	42	
	Polska	12	36	
Internet Top. Zoo	Fatman	17	42	$\{8, 9, 10, 12, 14, 16, 18, 20, 24, 28, 32, 35, 40, 45, 50\}$
	Digex	31	70	
	Cernet	41	116	
	Bellsouth	51	132	
	Intellifiber	73	190	
	RedBestel	84	186	
	Deltacom	113	322	
	Cogentco	197	486	

solution approaches. We conclude by discussing the scalability of the two-phase heuristic when the size of the substrate networks increases beyond medium sized networks.

3.5.1 The dataset

Throughout this section, we consider two different datasets, based on two different sets of substrate networks. The first group of substrate networks is taken from the SNDLIB [100] and consists of small to medium sized networks. The second group is taken from the Internet Topology Zoo [83] and contains rather large sized networks. Especially with regard to our exact solution approaches via mixed integer linear programming, we employ the first set of instances as our main dataset, and the latter one only to evaluate the scalability of our heuristic approaches.

For the first group of instances, we consider four networks of similar size and density, and transformed into directed graphs with anti-parallel links: Abilene, Atlanta, Nobel-us, and Polska. For the second group, we consider eight networks, again transformed into directed graphs with anti-parallel links: Fatman, Digex, Cernet, Bellsouth, Intellifiber, RedBestel, Deltacom, and Cogentco. We refer to Table 3.3 for some information on these networks. The physical node capacities are randomly drawn from the tuple $(10, 50, 100, 500)$, with a probability of $(0.1, 0.4, 0.4, 0.1)$. Physical link capacities are set to 500 for all the edges.

For these substrate networks, we consider VN requests with 12 virtual nodes, a profit chosen uniformly at random between 20 and 100, and a random topology with a link density of 0.5. As to the *locality* aspect, we construct each set $V^0(r, v)$ by first sampling uniformly at random a cardinality factor α_v^r from the interval $[\frac{1}{2}, 1]$ and then adding node $i \in V^0$ to $V^0(r, v)$ with a probability α_v^r.

For the virtual node and traffic demands, we mimic a case where historical data is available, creating a *historical* sequence of 100 data sets. First, for each uncertain coefficient ω_v^r or d_{vw}^r, we sample a value from the tuple $(10, 50, 100, 500)$ (scaled by 0.04 for nodes and 0.06 for links) uniformly at random, with a probability of $(0.1, 0.4, 0.4, 0.1)$. The historical sequence for that coefficient is constructed by adding to the previously sampled value a Gaussian error with zero mean and a standard deviation equal to three times the original value, for each of the 100 snapshots, forcing any demand value thus obtained to 0 if negative. Finally, the nominal node and traffic demands $\bar{w}_v^r$ and $\bar{d}_{vw}^r$ are computed as the arithmetic average over the 100 snapshots, computing $\hat{\omega}_v^r$ and $\hat{d}_{vw}^r$ as the largest deviations with respect to $\bar{w}_v^r$ and $\bar{d}_{vw}^r$ over the historical sequence.

We generate the instances with an increasing number of requests, that is

$$|R| \in \{5, 6, 7, 8, 9, 10, 12, 14, 16, 18, 20, 24, 28, 32\} \tag{3.30a}$$

for the instances of the SNDLIB and

$$|R| \in \{8, 9, 10, 12, 14, 16, 18, 20, 24, 28, 32, 35, 40, 45, 50\} \tag{3.30b}$$

for the instances from the Internet Topology Zoo. The VNs are constructed incrementally for each topology, so that every instance of a given substrate with r requests contains the same requests as an instance with the same topology and $r' < r$ requests, plus $r - r'$ additional ones. As a consequence, the value of an optimal solution for any given topology is a nondecreasing function of $|R|$. During the generation of the instances, we vary the random seed so that the sequence of VN requests is different for each physical topology. The data set thus constructed is composed of 56 instances from the SNDLIB and 129 instances from the Internet Topology Zoo.

We point out that all data used in this section, for the deterministic as well as for the robust case, is available for download and is briefly described on the website [39].

In the following, we compare the solutions obtained via the different methods with respect to their objective function value and their *empirical* protection level. The latter is defined as the number of snapshots, in the historical sequence of each instance, for which *no* node or link capacity constraint is violated by the solution that we have found. This way, the protection level corresponds to the ϵ in the chance constrained formulation (3.18a)-(3.18d), i.e., given a solution, the protection level empirically determines the maximum ϵ for which the solution was feasible for the chance constrained problem. Clearly, the higher the protection level of a solution, the "better" the solution is and, in general, there is a trade-off between protection level and objective value.

All our computations are carried out on an Intel(R) Core(TM) i7-3770 CPU @ 3.40 GHz with 32 GB RAM. We employ the state-of-the-art MILP solver CPLEX 12.4 [45], relying on AMPL [6] as modeling language. We set a time limit of 3600 seconds for the exact (Γ-robust) MILP formulations, adopting a much shorter time limit of 300 seconds per subproblem in both the two-phase and the adaptive heuristics. The latter is run for, at most, 12 iterations.

3.5.2 The deterministic VNE problem

We evaluate the deterministic problem, i.e., the problem where the input data is certain. Recall that we have described two approaches to the VNE problem, an exact MILP approach and a heuristic approach where node- and link-mapping is carried out in sequence, either by a two-phase or by an adaptive algorithm. We present solutions obtained for *worst-case* demands and for *average* demands. We start with the exact approach where VNE is tackled by the MILP as described in Remark 3.2.

Exact nominal and worst case solutions The complete results for the full data set are reported in Table 3.4. At the first glance, the table shows that the instances with the worst case data are harder to solve than those with average data. That is, the instances with worst-case data have an average optimality gap of 32% versus one of 0%, and an average computing time 3.5 times larger, thus showing that the problem gets more difficult for a higher load. More precisely, out of 56 instances, only 8 instances cannot be solved to optimality with average data (with an average gap of 2.75%), whereas this number increases to 33 instances with worst case data (with an average gap of 53%).

The same trend is observed in the objective values: as more available VNs imply more possibilities for profit, with rising $|R|$, higher objective values should be achieved. This is always the case for the average-case solution, even though some instances are not solved to optimality, but e.g., for the POLSKA instance with worst case data, the trend does not hold true for $|R| = 16$ and $|R| = 18$.

Clearly, for both types of data, the difficulty with respect to solution time of the instances rises with increasing problem size. That is with rising $|R|$ more solution time is required, respectively optimality gaps occur. Consider for example, the ATLANTA instance with average data. It can be solved for all $|R|$ up to 24 but not for $|R| = 28$ and $|R| = 32$. To a lesser extent the same holds for increasing sizes of the substrate networks. That is, the instances corresponding to the smallest substrate, i.e., to the ABILENE network, can be solved to optimality for all but two cases, whereas the other instances have significantly less cases for which an optimal solution can be found within the time limit.

All in all, the exact approach is well suited for small instances with low load. However, for larger instances, the optimality gaps become severe. Therefore, in the next paragraph, we evaluate heuristic solution approaches.

Heuristic nominal and worst case solutions In comparison to the exact approach above, we now consider a heuristic solution to the VNE problem. That is, we employ deterministic versions of the two-phase and the adaptive heuristic to our data. At first, we consider the two-phase heuristic:

We apply the two-phase heuristic as described in Subsection 3.2.3, that is, with a second phase problem which includes admission control. In the context of Section 3.4.3, we add Distance-Bounding Constraints (3.27) to the first phase to improve the results of the heuristic. As described in Section 3.4.3, we cluster the virtual node pairs into

Table 3.4: Results for the deterministic MILP formulation obtained within 3600 seconds with nominal and worst case data. All entries rounded to the nearest integer.

		$\|R\|$	5	6	7	8	9	10	12	14	16	18	20	24	28	32	**Avg**
Objective Fct.	Nominal	Abil	342	381	438	471	500	500	512	512	595	632	632	632	664	715	**538**
		Atl	402	444	523	559	623	695	853	970	1061	1197	1273	1324	1443	1447	**915**
		Nob	346	393	441	532	570	629	734	810	894	917	1011	1165	1211	1282	**781**
		Pol	265	312	398	452	550	644	782	875	968	1082	1082	1065	1219	1284	**784**
		Avg	**339**	**383**	**450**	**504**	**561**	**617**	**720**	**792**	**880**	**957**	**1000**	**1047**	**1134**	**1182**	**755**
	W.-Case	Abil	68	68	125	126	126	126	126	126	163	163	163	163	151	151	**132**
		Atl	251	332	344	210	301	301	194	263	184	174	336	220	227	348	**263**
		Nob	225	225	180	98	189	228	130	146	228	193	228	215	185	237	**193**
		Pol	169	169	217	245	277	277	277	301	301	252	238	238	348	263	**255**
		Avg	**178**	**199**	**217**	**170**	**223**	**233**	**182**	**209**	**219**	**196**	**241**	**209**	**228**	**250**	**211**
Opt. Gap (%)	Nominal	Abil	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
		Atl	0	0	0	0	0	0	0	0	0	0	0	0	0	2	**0**
		Nob	0	0	0	0	0	0	0	4	0	0	0	1	2	4	**1**
		Pol	0	0	0	0	0	0	0	0	0	0	0	5	3	1	**1**
		Avg	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**2**	**1**	**2**	**0**
	W.-Case	Abil	0	0	0	0	0	0	0	0	0	0	0	0	21	21	**3**
		Atl	32	0	0	81	26	26	96	58	127	145	25	97	100	31	**60**
		Nob	0	0	52	179	44	20	110	87	20	41	20	42	74	35	**52**
		Pol	0	22	13	0	0	0	0	0	0	29	36	36	0	33	**12**
		Avg	**8**	**5**	**16**	**65**	**18**	**11**	**51**	**36**	**37**	**54**	**20**	**44**	**49**	**30**	**32**
Comp. Time (s)	Nominal	Abil	3	14	7	13	168	853	507	364	244	78	16	71	1394	896	**331**
		Atl	1	1	1	4	5	5	10	35	47	15	518	177	3600	3600	**573**
		Nob	1	1	2	6	4	8	12	3600	33	502	3112	3600	3600	3600	**1291**
		Pol	0	1	1	1	2	4	5	8	143	24	1484	3600	3600	3600	**891**
		Avg	**1**	**4**	**3**	**6**	**44**	**218**	**133**	**1002**	**117**	**155**	**1283**	**1862**	**3049**	**2924**	**771**
	W.-Case	Abil	1	1	226	688	692	1659	1249	1357	31	29	44	34	3600	3600	**944**
		Atl	3600	242	105	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	**3111**
		Nob	30	176	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	**3100**
		Pol	14	3600	3600	3207	2738	2149	2109	255	1890	3600	3600	3600	3298	3600	**2661**
		Avg	**911**	**1005**	**1883**	**2774**	**2657**	**2752**	**2640**	**2203**	**2280**	**2707**	**2711**	**2708**	**3525**	**3600**	**2454**
Protection Level	Nominal	Abil	1	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
		Atl	27	19	1	3	0	0	0	0	0	0	0	0	0	0	**4**
		Nob	54	19	1	0	0	0	0	0	0	0	0	0	0	0	**5**
		Pol	38	29	22	14	7	2	0	0	0	0	0	0	0	0	**8**
		Avg	**30**	**17**	**6**	**4**	**2**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**4**
	W.-Case	Abil	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		Atl	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		Nob	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		Pol	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		Avg	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**

Table 3.5: Results for the deterministic two-phase heuristic with nominal (average) and worst case data. Entries are rounded to the nearest integer. "Sol. Quality" shows the relation of the solution value in comparison to the best solution value found by the exact method.

		$\|R\|$	5	6	7	8	9	10	12	14	16	18	20	24	28	32	**Avg**
Objective Fct.	Nominal	Abil	342	301	339	304	332	431	304	318	342	379	464	396	413	436	**364**
		Atl	402	444	481	523	517	536	582	622	684	726	665	688	633	709	**587**
		Nob	346	393	441	532	570	629	697	617	705	728	755	766	858	866	**636**
		Pol	265	312	370	424	492	539	623	585	585	504	506	473	535	646	**490**
		Avg	**339**	**363**	**408**	**446**	**478**	**534**	**552**	**536**	**579**	**584**	**598**	**581**	**610**	**664**	**519**
	W.-Case	Abil	0	0	0	0	0	0	0	0	0	0	0	94	0	0	**7**
		Atl	89	95	95	89	36	79	95	89	79	95	79	71	89	89	**84**
		Nob	180	180	98	98	180	180	130	180	98	180	180	98	185	98	**148**
		Pol	93	38	93	93	93	93	93	93	93	73	73	93	97	97	**87**
		Avg	**91**	**78**	**72**	**70**	**77**	**88**	**80**	**91**	**68**	**87**	**83**	**89**	**93**	**71**	**81**
Sol. Quality	Nominal	Abil	1.00	0.79	0.77	0.65	0.66	0.86	0.59	0.62	0.57	0.60	0.73	0.63	0.62	0.61	**0.69**
		Atl	1.00	1.00	0.92	0.94	0.83	0.77	0.68	0.64	0.64	0.61	0.52	0.52	0.44	0.49	**0.71**
		Nob	1.00	1.00	1.00	1.00	1.00	1.00	0.95	0.76	0.79	0.79	0.75	0.66	0.71	0.68	**0.86**
		Pol	1.00	1.00	0.93	0.94	0.89	0.84	0.80	0.67	0.60	0.47	0.47	0.44	0.44	0.50	**0.71**
		Avg	**1.00**	**0.95**	**0.91**	**0.88**	**0.85**	**0.87**	**0.76**	**0.67**	**0.65**	**0.62**	**0.62**	**0.56**	**0.55**	**0.57**	**0.75**
	W.-Case	Abil	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.58	0.00	0.00	**0.04**
		Atl	0.35	0.29	0.28	0.42	0.12	0.26	0.49	0.34	0.43	0.55	0.24	0.32	0.39	0.26	**0.34**
		Nob	0.80	0.80	0.54	1.00	0.95	0.79	1.00	1.23	0.43	0.93	0.79	0.46	1.00	0.41	**0.80**
		Pol	0.55	0.22	0.43	0.38	0.34	0.34	0.34	0.31	0.31	0.29	0.31	0.39	0.28	0.37	**0.35**
		Avg	**0.43**	**0.33**	**0.31**	**0.45**	**0.35**	**0.35**	**0.46**	**0.47**	**0.29**	**0.44**	**0.33**	**0.44**	**0.42**	**0.26**	**0.38**
Comp. Time (s)	Nominal	Abil	6	6	6	6	6	8	8	11	6	6	6	6	7	8	**7**
		Atl	11	12	11	11	11	13	11	14	19	14	31	37	56	37	**20**
		Nob	9	8	9	10	9	9	12	16	13	22	21	17	28	41	**16**
		Pol	6	6	6	6	7	7	8	10	17	8	8	11	11	11	**9**
		Avg	**8**	**8**	**8**	**8**	**8**	**9**	**10**	**13**	**14**	**13**	**17**	**18**	**26**	**24**	**13**
	W.-Case	Abil	7	6	6	7	6	6	6	6	6	6	7	6	6	7	**6**
		Atl	12	12	12	12	11	10	10	11	12	11	10	13	14	13	**12**
		Nob	9	10	10	11	10	9	32	10	10	13	11	9	9	9	**12**
		Pol	7	7	6	6	8	6	7	7	7	7	8	8	7	7	**7**
		Avg	**9**	**9**	**9**	**9**	**9**	**8**	**14**	**8**	**9**	**9**	**9**	**9**	**9**	**9**	**9**
Protection Level	Nominal	Abil	1	0	0	0	0	1	0	0	0	0	1	0	0	0	**0**
		Atl	75	72	55	1	1	0	2	0	0	0	0	0	0	0	**15**
		Nob	59	49	1	1	0	0	2	0	0	0	0	0	0	0	**8**
		Pol	96	86	79	0	0	1	0	0	0	0	0	1	0	0	**19**
		Avg	**58**	**52**	**34**	**1**	**0**	**1**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**10**
	W.-Case	Abil	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		Atl	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		Nob	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		Pol	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		Avg	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**

the three categories L, M, H (low, medium, and high). Each pair $v, w \in V^r$ belongs to H if $d^r_{vw} \geq 50$, to M if $10 \leq d^r_{vw} < 50$, and to L otherwise. For example, we set the corresponding parameters to $z_L = |V^0|$, $z_M = 2$, and $z_H = 1$. For a more involved discussion on these parameter values in the robust setting, we refer to Subsection 3.5.3.

The complete results for the two-phase heuristic on the full data set are given in Table 3.5. As expected, we directly observe that the two-phase heuristic provides, on average, worse solutions than the exact approach. That is, for nominal data, the heuristic achieves an average objective value of 519, respectively, for the worst case data, a value of 81, compared to the 755, respectively the 211, of the exact approach. Expressed differently, the heuristics achieves, on average, only 75% respectively 38% of the exact approach. In particular, the instances with worst case data seem to be the most difficult ones as, in all cases, at most one or two VNs are accepted and, for the smallest instances, ABILENE, only the zero solution can be found in all but one case.

The same trend is also supported by the "Sol. Quality" measure, which shows the ratio of the given solution value and the best solution value found by the exact method. As we can see, the heuristic does rarely achieve a ratio above 1.00, which further shows that the solution values obtainedare of average quality.

On the positive side, the time consumption of the heuristic approach is significantly lower with, on average, 13, respectively 9 seconds, than the one of the exact approach (771 and 2454 seconds). Interestingly, when focusing on the average values, the time consumption seems to scale with the number of VNs for the nominal data whereas it seems to stay constant for the worst case data.

When focusing on the adaptive heuristic, see Table 3.6, we observe a similar trend. On average, the objective values are worse than the ones from the exact approach but better than the ones from the two-phase heuristic, the same holds true for the "Sol. Quality" measure. I.e., the adaptive heuristic can only generate solutions which have on average 88% respectively 50% of the objective value of the exact approach. The solution times are longer than the ones required by the two-phase heuristic but still significantly shorter than the ones of the exact approach. Again, the solution time scales with the amount of VNs for nominal data but shows no particular trend for the worst case figures.

Both heuristics offer a reasonable alternative to the exact approach for the nominal data. That is, they offer a trade-off in objective value and computing time with respect to the MILP formulation. This trade-off seems to be more beneficial for the adaptive heuristic, however, if solution time is critical, the two-phase heuristic offers comparable objective function values at a much lower time investment. When considering worst-case data, the trade-off between solution quality and time requirement is not as clear. The time requirement of the MILP is an order of magnitude higher than the one for the heuristics but the same holds for the objective value. In this case, the heuristics perform not good enough to be an alternative choice compared to the exact approach.

Remarks on the protection level As we have anticipated in Section 3.4, although the objective function values with average data are much larger than those for the worst

Table 3.6: Results for the deterministic Adaptive Heuristic with nominal (average) and worst case data. Entries are rounded to the nearest integer. "Sol. Quality" shows the relation of the solution value in comparison to the best solution value found by the exact method.

		$\|R\|$	5	6	7	8	9	10	12	14	16	18	20	24	28	32	**Avg**
Objective Fct.	Nominal	Abil	322	381	438	450	361	480	472	472	509	465	564	564	489	510	**463**
		Atl	402	444	523	559	623	695	853	790	819	852	938	993	1029	912	**745**
		Nob	346	393	441	532	570	629	734	794	842	879	890	949	979	1198	**727**
		Pol	265	312	398	452	550	644	707	691	671	725	632	562	651	785	**575**
		Avg	**334**	**383**	**450**	**498**	**526**	**612**	**692**	**687**	**710**	**730**	**756**	**767**	**787**	**851**	**627**
	W.-Case	Abil	68	68	57	69	69	69	69	69	0	0	0	94	0	0	**45**
		Atl	95	95	95	156	95	95	89	150	89	89	79	168	79	79	**104**
		Nob	114	116	117	91	91	91	189	189	91	98	137	136	94	164	**123**
		Pol	169	93	121	121	121	121	121	121	93	93	93	143	143	93	**118**
		Avg	**112**	**93**	**98**	**109**	**94**	**94**	**117**	**132**	**68**	**70**	**77**	**135**	**79**	**84**	**97**
Sol. Quality	Nominal	Abil	0.94	1.00	1.00	0.96	0.72	0.96	0.92	0.92	0.86	0.74	0.89	0.89	0.74	0.71	**0.87**
		Atl	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.81	0.77	0.71	0.74	0.75	0.71	0.63	**0.87**
		Nob	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.98	0.94	0.96	0.88	0.81	0.81	0.93	**0.95**
		Pol	1.00	1.00	1.00	1.00	1.00	1.00	0.90	0.79	0.69	0.67	0.58	0.53	0.53	0.61	**0.81**
		Avg	**0.99**	**1.00**	**1.00**	**0.99**	**0.93**	**0.99**	**0.96**	**0.88**	**0.82**	**0.77**	**0.77**	**0.75**	**0.70**	**0.72**	**0.88**
	W.-Case	Abil	1.00	1.00	0.46	0.55	0.55	0.55	0.55	0.55	0.00	0.00	0.00	0.58	0.00	0.00	**0.41**
		Atl	0.38	0.29	0.28	0.74	0.32	0.32	0.46	0.57	0.48	0.51	0.24	0.76	0.35	0.23	**0.42**
		Nob	0.51	0.52	0.65	0.93	0.48	0.40	1.45	1.29	0.40	0.51	0.60	0.63	0.51	0.69	**0.68**
		Pol	1.00	0.55	0.56	0.49	0.44	0.44	0.44	0.40	0.31	0.37	0.39	0.60	0.41	0.35	**0.48**
		Avg	**0.72**	**0.59**	**0.48**	**0.68**	**0.45**	**0.42**	**0.72**	**0.70**	**0.30**	**0.35**	**0.31**	**0.64**	**0.32**	**0.32**	**0.50**
Comp. Time (s)	Nominal	Abil	6	11	13	22	34	26	15	23	53	45	52	48	65	65	**34**
		Atl	9	9	11	25	52	71	109	162	255	307	462	278	476	648	**205**
		Nob	8	8	8	8	8	8	59	111	134	202	151	224	252	364	**110**
		Pol	5	6	16	15	41	44	78	89	101	114	99	108	91	130	**67**
		Avg	**7**	**8**	**12**	**18**	**34**	**37**	**65**	**96**	**136**	**167**	**191**	**165**	**221**	**302**	**104**
	W.-Case	Abil	8	7	13	14	14	14	16	16	13	13	14	20	26	27	**15**
		Atl	34	39	44	75	75	89	121	348	55	52	351	83	92	87	**110**
		Nob	22	25	126	66	347	349	856	683	1795	1044	2647	353	97	64	**605**
		Pol	8	21	25	24	36	35	39	42	39	39	39	40	39	42	**34**
		Avg	**18**	**23**	**52**	**45**	**118**	**122**	**258**	**272**	**476**	**287**	**763**	**124**	**63**	**55**	**191**
Protection Level	Nominal	Abil	1	0	0	2	0	0	0	1	0	0	0	0	0	0	**0**
		Atl	75	72	75	3	4	0	0	1	0	0	0	0	0	0	**16**
		Nob	59	49	1	1	0	0	1	0	0	0	0	0	0	0	**8**
		Pol	96	86	41	0	1	0	0	0	1	0	0	0	0	1	**16**
		Avg	**58**	**52**	**29**	**2**	**1**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**10**
	W.-Case	Abil	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		Atl	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		Nob	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		Pol	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		Avg	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**

case, for both the heuristic and the exact approaches, for the majority of instances the corresponding solutions are infeasible in almost all the snapshots of the historical sequence. On the contrary, the solutions with worst case data are always feasible, at the expense of a very poor objective function value. This simple observation is illustrated, for the ABILENE instances, with solutions obtained from the exact approach, in Figure 3.11.

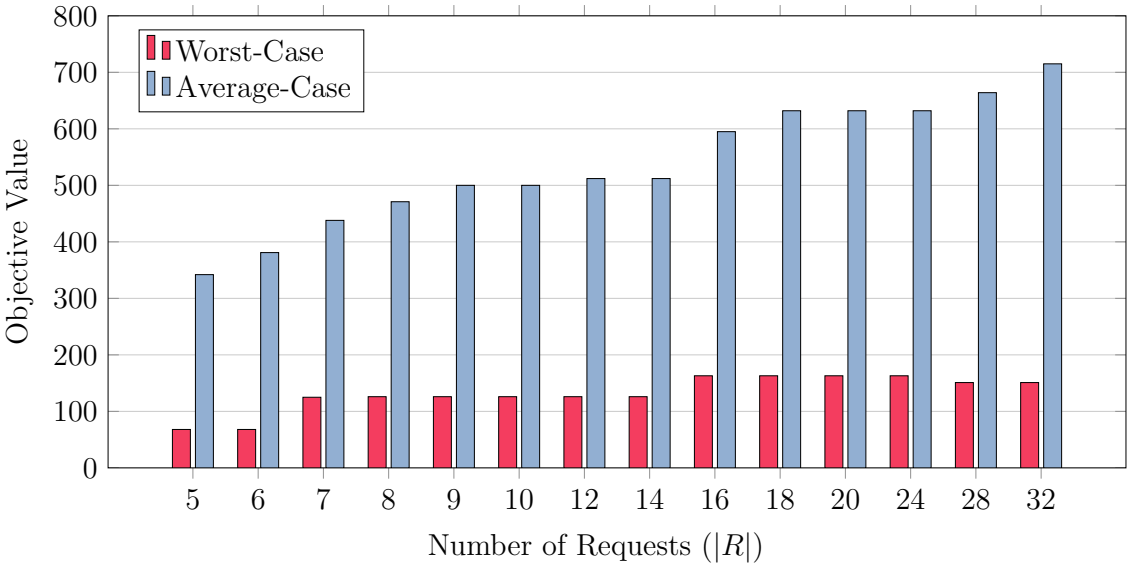
From a practical perspective, this behavior is troublesome as it offers no trade-off between both extremes. That is, any solution is either infeasible with high probability or induces a very conservative payoff. This way, even if feasibility in 90% of all cases is sufficient, one is forced to consider the solution corresponding to the worst-case. Clearly, from an economic point of view one looses potential profit this way. In the next subsection, we evaluate robust optimization approaches yielding this desired trade-off.

Conclusions In this subsection, we have investigated the deterministic VNE problem. We have observed that exact solution approaches are auspicious but require a substantial time investment, even for small to medium sized instances. In particular, most instances cannot be solved to optimality, but promising primal solutions can be obtained. We have seen that heuristic approaches can partially circumvent this issue, often offering competitive solutions at a much lower time investment. However, such behavior heavily depends on the problem data, i.e., for worst-case data the heuristic's solution quality was very low. We point out that for a better performance of the two phase heuristic, other parameter settings should be explored. A more in-depth discussion on this matter is given in the next subsection for the robust case.

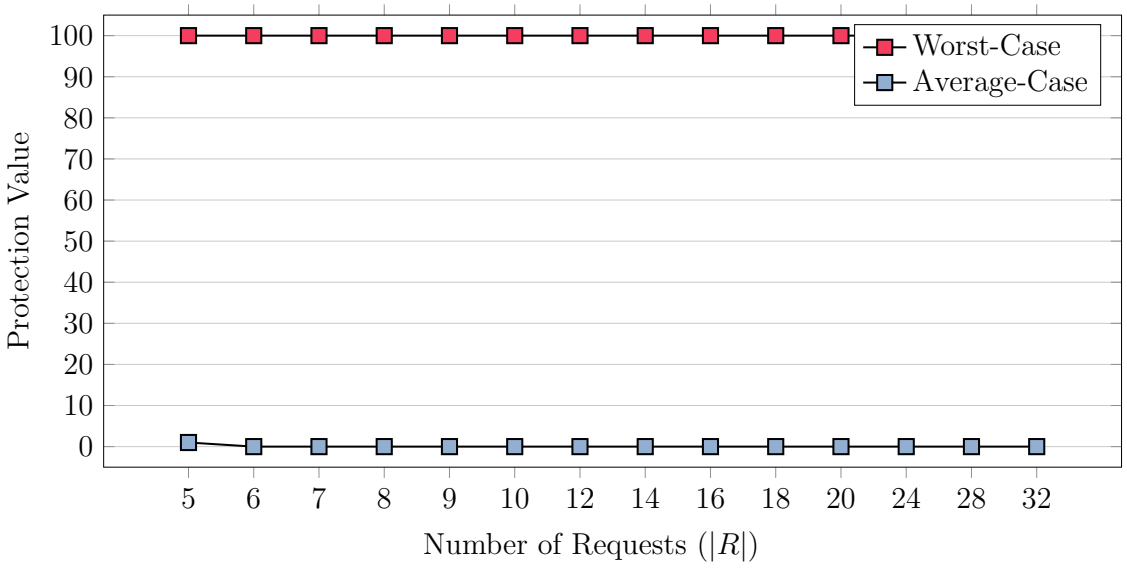
In the context of data uncertainty, we have briefly commented on the lack of the deterministic model to offer a trade-off between solution value and the corresponding protection value as it can either produce extremely optimistic or extremely conservative solutions. This way, the deterministic model is insufficient if data uncertainty occurs. Means to better cope with such uncertainty are evaluated in the next subsection.

3.5.3 The Γ-robust VNE problem

In this subsection, we focus on the VNE problem with data uncertainty. In particular, we assume that the uncertainty takes places in the virtual demand values of the VNs, namely in the parameters w_v^r and d_{vw}^r, while the substrate network and the number of VNs is certain. We tackle the uncertainty by Γ-robustness as described in Section 1.4. Recall that we have described two approaches to the Γ-robust VNE problem, an exact MILP approach and an heuristic approach where node- and link mapping are carried out in sequence, either by a two phase or by an adaptive algorithm. We start with by evaluating the exact approach, that is, where the VNE problem is tackled by the MILP as described in Remark 3.11, and evaluate the heuristic approaches afterwards. In particular for the two-phase heuristic, we perform extensive computations to determine the best parameter setting.



(a) Objective Function Values



(b) Protection Values

Figure 3.11: In (a): objective function values and in (b) protection values for worst-case and average-case data, reported as a function of $|R|$, for the ABILENE instances, obtained by the (deterministic) MILP.

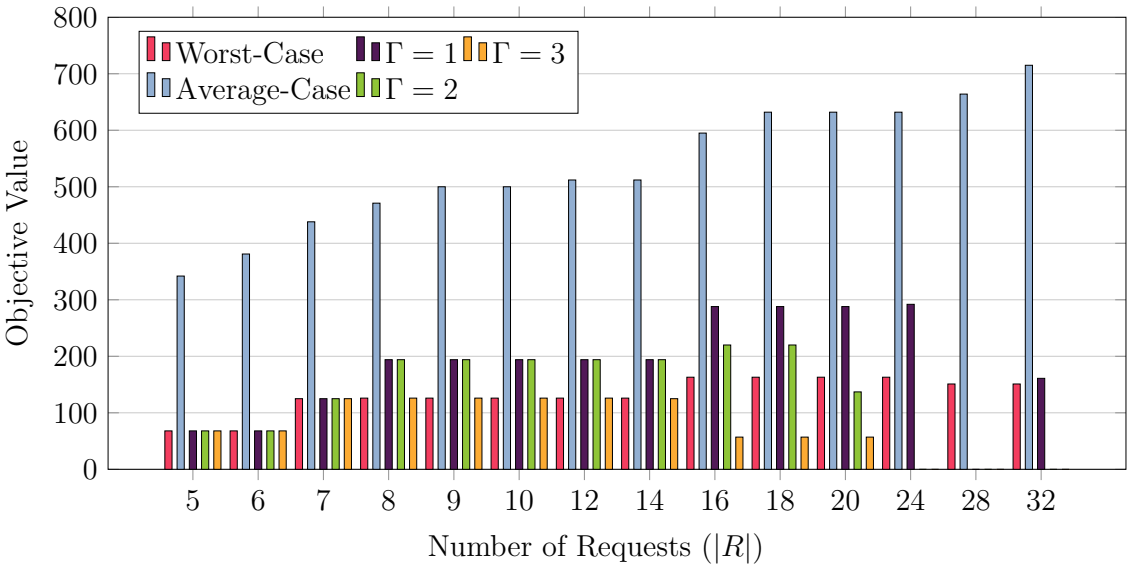
Exact solutions via a Γ-robust MILP

We illustrate the results obtained with the Γ-robust MILP formulation, first focusing on the ABILENE instances. We assume $\Gamma \in \{1, 2, 3\}$, adopting the same value for both the node and link capacity constraints. We do not report results for larger values of Γ as, in our experiments, we achieve a very high empirical protection level already for $\Gamma = 3$.

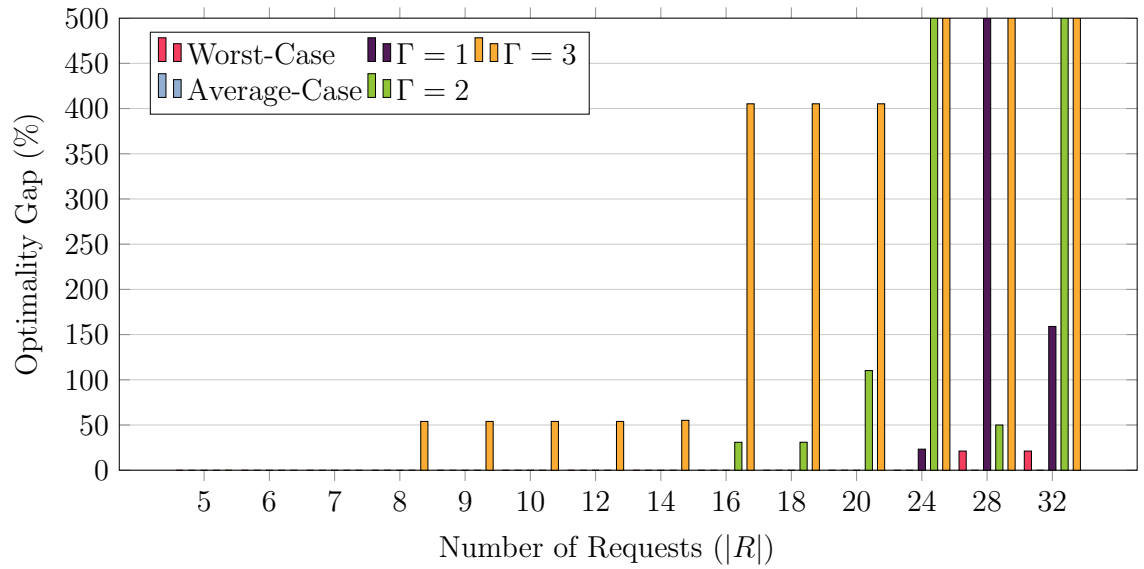
Figure 3.12 (a) reports the objective function value for different values of Γ as a function of $|R|$. Note that the value of an optimal Γ-robust solution should be between the optimal average and worst case solutions and that such value should be larger for smaller values of Γ. As we can see, this is not always the case. E.g., we observe that, for $|R| = 16$ and $\Gamma = 3$, as a consequence of prematurely reaching the time limit, we achieve a strictly smaller objective function value than for the worst case. In general, it seems that, with many requests ($|R| > 14$) and for increasing values of Γ, the Γ-robust formulations are more and more difficult to solve. As an example, observe that, for $|R| = 28$, no solution is found for $\Gamma \in \{1, 2, 3\}$, while the solution for $|R| = 32$ and $\Gamma = 1$ has a smaller value than that for the same Γ and $|R| = 24$. This is better shown in Figure 3.12 (b), which reports the optimality gap in percent, i.e., the difference between the best known integer solution (BI) and the best bound divided by the BI, of the solutions (truncated to 500 for illustration purposes), showing that, for instances with a large $|R|$ and for larger values of Γ, the exact approach does not scale well at all.

We obtain qualitatively comparable results also for the other topologies, as reported in Table 3.7. Indeed, when considering the full data set (56 instances) with three values of Γ (168 VNE problems in total), in 119 cases we cannot find an optimal solution within the time limit, registering an average gap of 76% (only considering the instances where the gap is finite). In 101 cases, not even a non-trivial solution, i.e., one where at least a single VN is accepted, is found within the time limit and thus, the gap is infinite. This shows that, compared to the deterministic problems with average and worst case data, with Γ-robustness we obtain much harder problems. In particular, this emphasizes the need of heuristic approaches as the heuristics invoked during the MILP solution process are remarkably unsuccessful in the robust case. Therefore, we consider heuristic solution approaches in the next paragraph.

We conclude by pointing out that, in the cases where the robust MILP provides non trivial solutions, in particular the $\Gamma = 1$ setting offers an interesting trade-off between the objective value and the induced protection level which cannot be achieved by the deterministic model. For example, considering the ABILENE instance with $|R| = 16$, the Γ-robust model yields a solution with value 288 and a protection level of 81 compared to solutions of value 596 and a protection level of 0 (nominal data), and a solution of value 163 and a protection level of 100 (worst case data). As we have mentioned before, from an economical point of view, solutions realizing such trade-offs are very interesting for the application. We take the previously mentioned example as a proof of concept that the robust model is, theoretically, well suited to obtain such alternative solutions. However, the solution quality of the overall approach is too low, such that, for most instances (and higher level of Γ), the model fails to provide them. Therefore, we refer to



(a) Objective Function Values



(b) Optimality gaps (in percent)

Figure 3.12: In (a): objective function values and in (b): optimality gaps for the ABILENE instances, obtained with the exact models (time limit of 3,600 seconds) and reported as a function of $|R|$. In (b), the bars are capped at 500 for illustration purposes.

Table 3.7: Results for the exact MILP Γ-robust formulation obtained within 3600 seconds. Entries are rounded to the nearest integer.

		$\lvert R\rvert$	5	6	7	8	9	10	12	14	16	18	20	24	28	32	**Avg**
Objective Fct.	$\Gamma=1$	**Abil**	68	68	125	194	194	194	194	194	288	288	288	292	-	161	**196**
		Atl	332	374	453	489	553	-	-	-	-	-	-	-	-	-	**440**
		Nob	294	341	389	480	518	518	-	-	-	-	-	-	-	-	**423**
		Pol	265	312	398	452	550	550	-	-	-	-	-	-	-	-	**421**
		Avg	**240**	**274**	**341**	**404**	**454**	**421**	**194**	**194**	**288**	**288**	**288**	**292**	**-**	**161**	**295**
	$\Gamma=2$	**Abil**	68	68	125	194	194	194	194	194	220	220	137	-	-	-	**164**
		Atl	332	374	-	-	-	-	-	-	-	-	-	-	-	-	**353**
		Nob	294	294	294	98	-	-	-	-	-	-	-	-	-	-	**245**
		Pol	265	312	321	-	104	76	-	-	-	-	-	-	-	-	**180**
		Avg	**240**	**262**	**247**	**97**	**149**	**135**	**194**	**194**	**220**	**220**	**137**	**-**	**-**	**-**	**190**
	$\Gamma=3$	**Abil**	68	68	125	126	126	126	126	125	57	57	57	-	-	-	**96**
		Atl	332	-	-	-	-	-	-	-	-	-	-	-	-	-	**332**
		Nob	225	225	222	-	-	-	-	-	-	-	-	-	-	-	**224**
		Pol	197	-	-	-	-	-	-	-	-	-	-	-	-	-	**197**
		Avg	**206**	**147**	**174**	**126**	**126**	**126**	**126**	**125**	**57**	**57**	**57**	**-**	**-**	**-**	**121**
Opt. Gap (%)	$\Gamma=1$	**Abil**	0	0	0	0	0	0	0	0	0	0	0	23	∞	159	**14**
		Atl	0	0	0	0	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	**0**
		Nob	0	0	0	0	0	0	∞	∞	∞	∞	∞	∞	∞	∞	**0**
		Pol	0	0	0	0	0	0	∞	∞	∞	∞	∞	∞	∞	∞	**0**
		Avg	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**23**	**-**	**159**	**14**
	$\Gamma=2$	**Abil**	0	0	0	0	0	0	0	0	31	31	110	∞	∞	∞	**16**
		Atl	0	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	**0**
		Nob	0	0	16	319	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	**84**
		Pol	0	0	24	∞	429	624	∞	∞	∞	∞	∞	∞	∞	∞	**215**
		Avg	**0**	**0**	**13**	**160**	**214**	**312**	**0**	**0**	**31**	**31**	**110**	**-**	**-**	**-**	**79**
	$\Gamma=3$	**Abil**	0	0	0	54	54	54	54	55	405	405	405	∞	∞	∞	**135**
		Atl	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	**-**
		Nob	0	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	**0**
		Pol	0	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	∞	**0**
		Avg	**0**	**0**	**0**	**54**	**54**	**54**	**54**	**55**	**405**	**405**	**405**	**-**	**-**	**-**	**135**
Comp. Time (s)	$\Gamma=1$	**Abil**	2	1	2	61	3	33	5	49	6	671	146	3600	3600	3600	**841**
		Atl	1	3	6	3600	10	3600	3600	3600	3600	3600	3600	3600	3600	3600	**2573**
		Nob	2	3	3	2068	13	19	3600	3600	3600	3600	3600	3600	3600	3600	**2208**
		Pol	3	270	338	11	1975	2890	3600	3600	3600	3600	3600	3600	3600	3600	**2449**
		Avg	**2**	**69**	**87**	**1435**	**500**	**1635**	**2701**	**2712**	**2702**	**2868**	**2737**	**3600**	**3600**	**3600**	**2018**
	$\Gamma=2$	**Abil**	4	5	36	425	784	156	399	441	3601	3600	3600	3600	3600	3600	**1704**
		Atl	186	2192	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	**3256**
		Nob	8	433	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	**3117**
		Pol	144	2502	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	**3275**
		Avg	**86**	**1283**	**2709**	**2806**	**2896**	**2739**	**2800**	**2810**	**3600**	**3600**	**3600**	**3600**	**3600**	**3600**	**2838**
	$\Gamma=3$	**Abil**	12	31	37	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	**2834**
		Atl	876	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	**3405**
		Nob	97	751	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	**3146**
		Pol	180	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	3600	**3356**
		Avg	**291**	**1996**	**2709**	**3600**	**3600**	**3600**	**3600**	**3600**	**3600**	**3600**	**3600**	**3600**	**3600**	**3600**	**3185**
Protection Level	$\Gamma=1$	**Abil**	96	99	94	83	89	88	83	97	81	60	80	85	-	88	**86**
		Atl	98	93	86	63	22	-	-	-	-	-	-	-	-	-	**84**
		Nob	92	86	94	72	64	78	-	-	-	-	-	-	-	-	**81**
		Pol	95	85	66	76	33	42	-	-	-	-	-	-	-	-	**73**
		Avg	**95**	**91**	**85**	**33**	**77**	**69**	**83**	**97**	**81**	**60**	**80**	**85**	**-**	**88**	**82**
	$\Gamma=2$	**Abil**	100	100	99	100	98	99	100	99	100	100	100	-	-	-	**100**
		Atl	100	100	-	-	-	-	-	-	-	-	-	-	-	-	**100**
		Nob	100	99	100	100	-	-	-	-	-	-	-	-	-	-	**100**
		Pol	100	96	100	-	100	100	-	-	-	-	-	-	-	-	**99**
		Avg	**100**	**99**	**100**	**100**	**99**	**99**	**100**	**99**	**100**	**100**	**100**	**-**	**-**	**-**	**100**
	$\Gamma=3$	**Abil**	100	100	100	100	100	100	100	100	100	100	100	-	-	-	**100**
		Atl	100	-	-	-	-	-	-	-	-	-	-	-	-	-	**100**
		Nob	100	100	100	-	-	-	-	-	-	-	-	-	-	-	**100**
		Pol	100	-	-	-	-	-	-	-	-	-	-	-	-	-	**100**
		Avg	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**-**	**-**	**-**	**100**

the next paragraph where we derive such Γ-robust solutions by heuristics, particularly tailored to generate solutions in a tighter time-frame.

Heuristic solutions

Let us now consider the two heuristic methods. Differently from the exact case, where we consider $\Gamma \in \{0, 1, 2, 3\}$ for both node and link capacity constraints, in the two heuristic methods, we always set $\Gamma = 0$ for the second subproblem (which carries out the link-mapping). This is because, even without explicitly accounting for robustness in it, we still obtain solutions with a very high empirical protection level, as we will better illustrate in the following.

Two-phase heuristics: performance variability Let us first focus on the two-phase heuristic. As described in Section 3.4.3, we cluster the virtual nodes pairs into the three categories L, M, H (low, medium, and high). Each pair $v, w \in V^r$ belongs to H if $d^r_{vw} \geq 50$, to M if $10 \leq d^r_{vw} < 50$, and to L otherwise. To assess the *sensitivity* of the heuristic with respect to the parameters z_L, z_M, and z_H, we illustrate the results obtained for all the combinations of $z_L, z_M, z_H \in P := \{|V^0|, \frac{|V^0|}{2}, \frac{|V^0|}{4}, 2, 1\}$, with $z_L \leq z_M \leq z_H$.

Table 3.8 reports a comparison over all parameter settings, aggregated over all substrate networks and all Γ values. Let $SOL(\Gamma, s, r, p)$ be the solution value of the two-phase heuristic for a given value of Γ, a substrate network s, a number of requests r, and the parameter setting $p \in P \times P \times P$, where

$$\Gamma \in G := \{0, 1, 2, 3\}, \tag{3.31a}$$

$$s \in S := \{\text{Abilene, Atlanta, Nobel-us, Polska}\}, \tag{3.31b}$$

$$r = |R| \in \{5, \ldots, 32\}. \tag{3.31c}$$

For a fixed $r \in R$ and parameter setting $p_0 \in P$, each entry of Table 3.8 is computed as

$$\frac{1}{|G|\,|S|\,|P|} \sum_{\Gamma \in G} \sum_{s \in S} \sum_{p \in P} \frac{SOL(\Gamma, s, r, p)}{SOL(\Gamma, s, r, p_0)}. \tag{3.32}$$

Thus, each entry describes the relative quality of the solutions obtained with the parameter setting p_0, compared to the solutions obtained with all other parameter settings. The lower the value, the better p_0 performs with respect to all other parameter settings. In this table the results for the best setting, on average, over all the instances are underlined. As we can see, there is a unique "winner": the setting where

$$z_L = |V^0|, \quad z_M = 2, \quad \text{and } z_H = 1. \tag{3.33}$$

Such evaluation requires an extensive amount of computations. Hence, selecting the best suited setting in this manner may be infeasible in practice. In this context, it is worth mentioning that, if one is not able to find such good parameter setting beforehand, the variability of the two-phase heuristic in terms of solution quality can be quite high. This phenomenon is better shown in Table 3.9, which reports the minimum and

Table 3.8: Performance ratio of each parameter setting $z_L, z_M, z_H \in \{|V^0|, \frac{|V^0|}{2}, \frac{|V^0|}{4}, 2, 1\}$ with $z_L \leq z_M \leq z_H$, for each $|R|$. The results for the, on average, best setting over all the instances are underlined.

z_L	z_M	z_H	$\|R\|$ 5	6	7	8	9	10	12	14	16	18	20	24	28	32	**Avg**
1	1	1	1.41	1.31	1.41	1.56	1.60	1.39	1.43	1.44	1.48	1.36	1.48	1.34	1.10	1.01	**1.38**
2	1	1	1.16	1.09	1.15	1.20	1.21	1.17	1.17	1.15	1.00	0.98	1.01	1.00	0.93	0.89	**1.08**
2	2	1	0.97	0.96	1.02	0.99	0.98	0.98	0.95	0.94	0.96	0.93	0.91	0.91	0.93	0.89	**0.95**
2	2	2	0.97	0.97	1.01	0.98	0.98	0.99	0.97	0.96	0.97	0.94	0.92	0.94	0.94	0.99	**0.97**
$\|V^0\|$	1	1	1.16	1.08	1.15	1.19	1.20	1.17	1.16	1.14	1.01	0.99	0.99	0.97	0.92	0.89	**1.07**
$\|V^0\|$	2	1	0.96	0.96	0.99	0.96	0.95	0.96	0.93	0.93	0.93	0.93	0.93	0.89	0.94	0.91	**0.94**
$\|V^0\|$	2	2	0.96	0.96	1.01	0.97	0.98	0.99	0.97	0.95	0.96	0.97	0.99	0.97	0.98	0.98	**0.97**
$\|V^0\|$	$\|V^0\|$	1	0.96	0.96	0.98	0.97	0.96	0.96	1.05	0.96	0.99	0.98	0.98	0.99	0.96	1.01	**0.98**
$\|V^0\|$	$\|V^0\|$	2	0.96	0.96	0.96	0.97	0.98	0.98	1.04	1.02	1.03	1.07	1.05	1.05	1.08	1.04	**1.01**
$\|V^0\|$	$\|V^0\|$	$\frac{\|V^0\|}{2}$	0.96	0.98	1.00	1.07	1.06	1.05	1.12	1.17	1.13	1.14	1.14	1.15	1.16	1.10	**1.09**
$\|V^0\|$	$\|V^0\|$	$\frac{\|V^0\|}{4}$	0.96	0.98	0.99	0.99	1.00	0.99	1.03	1.05	1.09	1.13	1.10	1.11	1.12	1.08	**1.05**
$\|V^0\|$	$\|V^0\|$	$\|V^0\|$	0.96	0.98	1.00	1.07	1.06	1.05	1.12	1.17	1.13	1.14	1.14	1.15	1.16	1.10	**1.09**
$\|V^0\|$	$\frac{\|V^0\|}{2}$	1	0.96	0.96	0.98	0.96	0.96	0.96	0.96	0.96	0.98	0.98	0.98	0.99	0.97	1.00	**0.97**
$\|V^0\|$	$\frac{\|V^0\|}{2}$	2	0.96	0.96	0.96	0.97	0.98	1.00	1.03	1.02	1.03	1.07	1.05	1.06	1.10	1.04	**1.02**
$\|V^0\|$	$\frac{\|V^0\|}{2}$	$\frac{\|V^0\|}{2}$	0.96	0.98	1.00	1.07	1.06	1.05	1.11	1.16	1.12	1.14	1.15	1.15	1.14	1.11	**1.08**
$\|V^0\|$	$\frac{\|V^0\|}{2}$	$\frac{\|V^0\|}{4}$	0.96	0.98	0.99	0.99	1.00	0.99	1.01	1.05	1.07	1.13	1.10	1.12	1.13	1.08	**1.04**
$\|V^0\|$	$\frac{\|V^0\|}{4}$	1	0.96	0.96	0.96	0.97	0.94	0.97	0.95	0.95	0.96	0.95	0.97	0.93	0.95	0.94	**0.95**
$\|V^0\|$	$\frac{\|V^0\|}{4}$	2	0.97	0.96	0.97	0.97	0.98	1.01	1.01	0.99	0.99	0.99	1.02	1.06	1.02	1.05	**1.00**
$\|V^0\|$	$\frac{\|V^0\|}{4}$	$\frac{\|V^0\|}{4}$	0.96	0.97	0.98	0.99	0.99	1.00	0.99	1.03	1.04	1.05	1.02	1.05	1.06	1.67	**1.06**
$\frac{\|V^0\|}{2}$	1	1	1.16	1.08	1.15	1.19	1.20	1.18	1.16	1.14	1.01	0.99	0.99	0.95	0.92	0.88	**1.07**
$\frac{\|V^0\|}{2}$	2	1	0.96	0.96	0.99	0.96	0.95	0.96	0.93	0.92	0.94	0.96	0.93	0.91	0.95	0.90	**0.95**
$\frac{\|V^0\|}{2}$	2	2	0.96	0.96	1.01	0.97	0.98	1.00	0.97	0.95	0.96	0.97	0.99	0.98	0.97	0.99	**0.98**
$\frac{\|V^0\|}{2}$	$\frac{\|V^0\|}{2}$	1	0.96	0.96	0.98	0.96	0.96	0.96	0.96	0.96	0.98	0.98	0.98	0.99	0.97	1.02	**0.97**
$\frac{\|V^0\|}{2}$	$\frac{\|V^0\|}{2}$	2	0.96	0.96	0.96	0.97	0.98	0.98	1.02	1.02	1.04	1.07	1.04	1.06	1.08	1.02	**1.01**
$\frac{\|V^0\|}{2}$	$\frac{\|V^0\|}{2}$	$\frac{\|V^0\|}{2}$	0.96	0.98	1.00	1.07	1.12	1.05	1.11	1.14	1.13	1.14	1.13	1.16	1.15	1.11	**1.09**
$\frac{\|V^0\|}{2}$	$\frac{\|V^0\|}{2}$	$\frac{\|V^0\|}{4}$	0.96	0.98	0.99	0.99	1.00	0.99	1.01	1.05	1.10	1.13	1.10	1.12	1.11	1.09	**1.05**
$\frac{\|V^0\|}{2}$	$\frac{\|V^0\|}{4}$	1	0.96	0.96	0.96	0.97	0.94	0.97	0.95	0.95	0.96	0.95	0.96	0.94	0.96	0.94	**0.96**
$\frac{\|V^0\|}{2}$	$\frac{\|V^0\|}{4}$	2	0.96	0.96	0.97	0.97	0.98	1.03	1.01	0.99	0.99	0.99	1.02	1.06	1.02	1.05	**1.00**
$\frac{\|V^0\|}{2}$	$\frac{\|V^0\|}{4}$	$\frac{\|V^0\|}{4}$	0.96	0.96	0.98	0.99	0.99	1.00	0.99	1.02	1.04	1.05	1.02	1.04	1.05	1.65	**1.05**
$\frac{\|V^0\|}{4}$	1	1	1.16	1.08	1.15	1.19	1.20	1.18	1.17	1.13	1.02	1.00	0.99	0.96	0.94	0.90	**1.08**
$\frac{\|V^0\|}{4}$	2	1	0.97	0.97	1.02	0.96	0.95	0.96	0.93	0.92	0.94	0.94	0.92	0.95	0.98	0.93	**0.95**
$\frac{\|V^0\|}{4}$	2	2	0.97	0.97	1.01	0.96	0.96	0.97	0.95	0.95	0.95	0.95	0.98	0.95	1.03	1.03	**0.97**
$\frac{\|V^0\|}{4}$	$\frac{\|V^0\|}{4}$	1	0.96	0.96	0.96	0.96	0.96	0.96	0.94	0.94	0.93	0.94	0.95	0.95	0.96	0.92	**0.95**
$\frac{\|V^0\|}{4}$	$\frac{\|V^0\|}{4}$	2	0.96	0.97	0.96	0.97	0.97	0.97	0.99	0.98	0.97	0.94	1.00	1.00	1.02	1.03	**0.98**
$\frac{\|V^0\|}{4}$	$\frac{\|V^0\|}{4}$	$\frac{\|V^0\|}{4}$	0.96	0.96	0.97	0.99	0.98	1.00	0.95	0.97	1.01	1.01	0.99	1.06	1.35	1.04	**1.02**

Table 3.9: Best (max) and worst (min) solution values obtained via the two-phase heuristic over all parameter settings, i.e., $z_L, z_M, z_H \in \{|V^0|, |V^0|/2, |V^0|/4, 2, 1\}$, with $z_L \leq z_M \leq z_H$. Entries are rounded to the nearest integer.

	$\|R\|$		5	6	7	8	9	10	12	14	16	18	20	24	28	32	$\emptyset$
$\Gamma=0$	ABI	max	342	381	438	471	500	500	512	512	595	632	632	632	664	715	**538**
		min	224	301	339	304	332	332	304	318	342	379	438	396	413	369	**342**
	ATL	max	402	444	523	559	623	695	853	970	1061	1118	1206	1262	1416	1387	**894**
		min	332	363	453	489	517	536	582	622	631	712	665	664	633	701	**564**
	NOB	max	346	393	441	532	570	629	734	846	894	917	1011	1176	1223	1313	**788**
		min	346	393	441	487	522	499	603	617	705	712	660	715	725	786	**587**
	POL	max	265	312	398	452	550	644	782	837	968	1054	961	1011	1082	1141	**747**
		min	265	312	370	424	473	539	614	585	580	504	506	473	535	601	**484**
Avg (max)			**339**	**383**	**450**	**504**	**561**	**617**	**720**	**791**	**880**	**930**	**953**	**1020**	**1096**	**1139**	**742**
Avg (min)			**292**	**342**	**401**	**426**	**461**	**477**	**526**	**536**	**565**	**577**	**567**	**562**	**577**	**614**	**494**
$\Gamma=1$	ABI	max	68	68	125	194	194	194	194	194	288	288	288	360	417	417	**235**
		min	0	0	57	57	57	57	57	57	151	151	151	223	280	280	**113**
	ATL	max	332	374	453	489	553	625	783	900	912	976	955	996	1070	1115	**752**
		min	251	251	330	366	366	438	596	588	552	644	574	622	0	0	**398**
	NOB	max	294	341	389	480	518	518	623	652	631	636	649	744	776	775	**573**
		min	196	243	291	382	420	420	420	464	464	464	519	644	649	539	**437**
	POL	max	265	312	398	452	550	550	600	644	644	687	687	698	793	856	**581**
		min	235	282	360	360	458	435	480	537	557	532	530	525	501	574	**455**
Avg (max)			**240**	**274**	**341**	**404**	**454**	**472**	**550**	**598**	**619**	**647**	**645**	**700**	**764**	**791**	**535**
Avg (min)			**171**	**194**	**260**	**291**	**325**	**338**	**388**	**412**	**431**	**448**	**444**	**504**	**358**	**348**	**351**
$\Gamma=2$	ABI	max	68	68	125	194	194	194	194	194	288	288	288	256	288	288	**209**
		min	0	0	57	57	57	57	57	57	57	57	57	57	114	114	**57**
	ATL	max	332	374	453	489	511	519	610	722	709	714	739	759	784	740	**604**
		min	251	251	330	366	366	438	524	559	548	499	508	449	89	0	**370**
	NOB	max	294	294	342	366	402	402	402	402	402	433	419	512	486	478	**402**
		min	196	196	196	196	234	304	341	278	278	278	278	328	366	0	**248**
	POL	max	265	312	370	422	454	466	494	528	538	573	573	573	624	704	**493**
		min	235	282	340	384	397	397	381	399	367	334	359	404	447	417	**367**
Avg (max)			**240**	**262**	**323**	**368**	**390**	**395**	**425**	**462**	**484**	**502**	**505**	**525**	**546**	**553**	**427**
Avg (min)			**171**	**182**	**231**	**251**	**264**	**299**	**326**	**323**	**313**	**292**	**301**	**310**	**254**	**133**	**261**
$\Gamma=3$	ABI	max	68	68	125	137	137	137	137	137	220	220	220	220	277	277	**170**
		min	0	0	57	57	57	57	57	57	57	57	57	57	114	114	**57**
	ATL	max	332	332	411	447	447	447	459	560	560	560	560	585	648	644	**499**
		min	251	251	330	366	366	366	459	449	451	445	451	499	544	0	**373**
	NOB	max	225	225	273	282	320	320	320	320	320	320	337	457	453	468	**331**
		min	82	129	129	129	129	266	157	158	158	266	158	242	275	327	**186**
	POL	max	197	244	322	346	406	406	456	500	500	554	526	526	571	638	**442**
		min	169	169	283	271	271	271	321	305	305	324	324	356	399	428	**300**
Avg (max)			**206**	**217**	**283**	**303**	**328**	**328**	**343**	**379**	**400**	**414**	**411**	**447**	**487**	**507**	**361**
Avg (min)			**126**	**137**	**200**	**206**	**206**	**240**	**249**	**242**	**243**	**273**	**248**	**289**	**333**	**217**	**229**

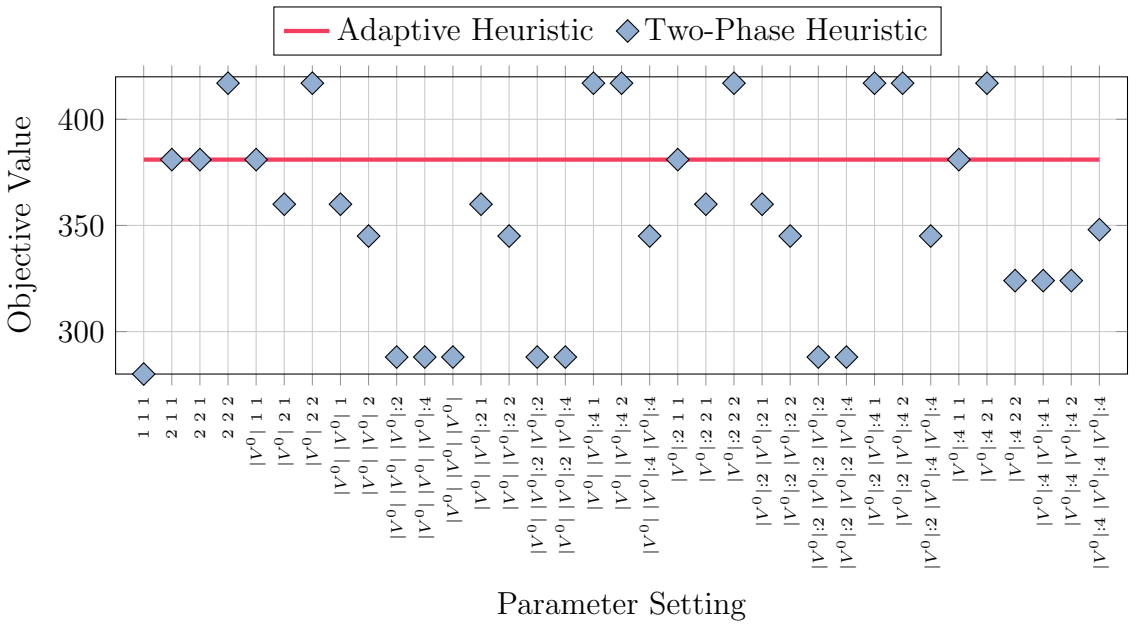


Figure 3.13: Objective function values (blue) of the two-phase heuristic, with $|R| = 28$ and $\Gamma = 1$ with $z_L, z_M, z_H \in \{|V^0|, |V^0|/2, |V^0|/4, 2, 1\}$ and $z_L \leq z_M \leq z_H$ for the ABILENE instances. The corresponding solution obtained via the adaptive heuristic is shown in red.

maximum objective function value achieved with different parameter settings for each instance. Remarkably, the differences in solution quality are quite significant. As an example, for $\Gamma = 2$ we have solution values between, on average, 427 and 261. We point out that this variability also explains the somewhat moderate success, at least with respect to worst-case data, we have observed when tackling the deterministic VNE problem with the two-phase heuristic in the previous subsection. Apparently, the "best" parameter setting for the Γ-robust problem is not as good for the deterministic case.

For a visual depiction of this variability, see Figure 3.13, which reports the solution values for all parameter settings for the ABILENE instance, with $|R| = 28$ and $\Gamma = 1$, as well as the solution value obtained by the adaptive heuristic, which we consider in detail in the next paragraph.

We remark that not only the parameters z_L, z_M, and z_H induce performance variability, but that the partition of the traffic demands into categories as L, M, H has a similar effect. We do not examine this case here, however, we point out that the adaptive heuristic, see below, does not include such user-supplied information.

Two-phase heuristics: the "best" parameter setting We now compare the results of the "winning" parameter setting $z_L = |V^0|$, $z_M = 2$, $z_H = 1$ to the exact approach. We first illustrate the results for the ABILENE instances. In Figure 3.14, we report the ratio between the solution values found heuristically and the best ones obtained via the exact formulation (in case no non-zero solution is found with the exact method, we cap this

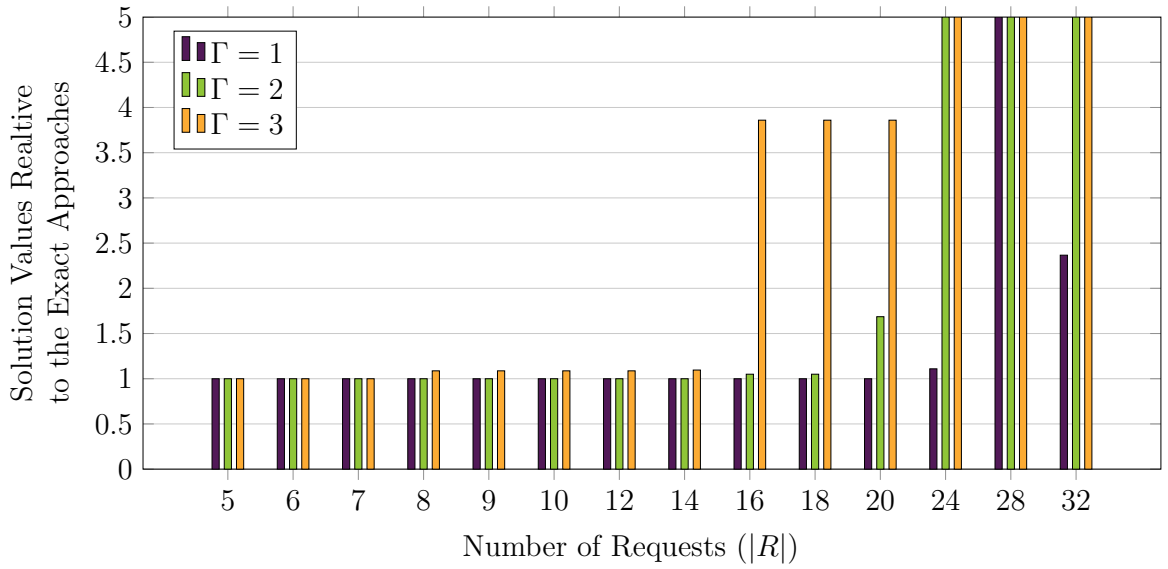


Figure 3.14: Objective function value ratios between the best solutions found via the two-phase heuristic ($z_L = |V^0|$, $z_M = 2$, and $z_H = 1$) and the exact formulation, reported as a function of $|R|$, for the ABILENE instances. The bars are capped at 5.0 for illustration purposes.

ratio at 5.0). Interestingly, for $\Gamma = 1$, the heuristic provides competing solutions to those obtained via the exact approach for all the ABILENE instances. For the first half of the instances (those with $|R| \leq 14$), the heuristic achieves a comparable objective function value (with a ratio close to 1.0), while it clearly outperforms the exact approaches on the harder instances ($|R| \geq 16$) where, for $\Gamma = 3$, the heuristic solutions are better by a factor larger than 3.5.

The results for the complete data set can be found in Table 3.10. The corresponding solution times of the single phases are reported in Table 3.11. We remark that the heuristic method finds non-zero solutions for all the instances and for all values of Γ, whereas, with the exact method, we find nonzero solutions for only 87 cases out of 168. For the cases where a solution to the exact formulation is known, we find solutions which are substantially better than the best ones found via the exact method within the time limit. On average, the heuristic yields solutions with an objective function value that is better by a factor of 1.42. If we restrict to $\Gamma > 0$, this factor goes up to 2.04.

For all the instances that can be solved to optimality with the exact approach, equivalent solution values are obtained with the heuristic approaches. This indicates that, at least on the smaller instances for which the optimal solution value is known, the heuristics are competitive in terms of solution quality. Table 3.11 also illustrates how much easier the second phase problem is with respect to the first one, in terms of computing times. What we can also see there is that the overall time requirements of the two phases is, on average, below 150 seconds, even though many instances meet their time limit of 300 seconds. Recalling that most instances run into their time limit of 3600 seconds in

Table 3.10: Detailed results for the two-phase heuristic with the a priori determined parameter setting $z_L = |V^0|, z_M = 2$, and $z_H = 1$.

		$\|R\|$	5	6	7	8	9	10	12	14	16	18	20	24	28	32	**Avg**
Objective Function	$\Gamma = 0$	ABI	342	381	438	471	500	500	512	512	595	632	632	632	664	715	**538**
		ATL	402	444	523	559	623	695	853	900	988	1001	1119	1091	1179	1090	**819**
		NOB	346	393	441	532	570	629	696	771	782	841	851	997	1105	1128	**720**
		POL	265	312	398	452	522	644	702	718	726	714	795	873	774	866	**626**
		Avg	**339**	**383**	**450**	**504**	**554**	**617**	**691**	**725**	**773**	**797**	**849**	**898**	**931**	**950**	**676**
	$\Gamma = 1$	ABI	68	68	125	194	194	194	194	194	288	288	288	360	360	381	**228**
		ATL	332	374	453	489	553	625	783	900	912	960	949	925	892	903	**718**
		NOB	294	341	389	480	518	518	555	631	621	636	606	724	650	637	**543**
		POL	265	312	398	452	522	550	570	606	642	633	602	573	651	732	**536**
		Avg	**240**	**274**	**341**	**404**	**447**	**472**	**526**	**583**	**616**	**629**	**611**	**646**	**638**	**663**	**506**
	$\Gamma = 2$	ABI	68	68	125	194	194	194	194	194	220	220	220	256	277	277	**193**
		ATL	332	374	453	489	511	519	605	655	686	697	703	683	737	730	**584**
		NOB	294	294	342	366	402	402	402	395	401	360	404	467	411	410	**382**
		POL	265	312	368	394	454	406	478	500	491	526	554	573	601	606	**466**
		Avg	**240**	**262**	**322**	**361**	**390**	**380**	**420**	**436**	**450**	**451**	**470**	**495**	**507**	**506**	**406**
	$\Gamma = 3$	ABI	68	68	68	126	126	126	126	126	220	220	220	220	220	220	**154**
		ATL	332	332	411	447	447	447	459	497	535	560	497	585	604	582	**481**
		NOB	225	225	273	282	320	320	320	319	320	320	330	457	366	354	**317**
		POL	197	244	322	346	397	406	456	500	500	526	526	475	500	585	**427**
		Avg	**206**	**217**	**269**	**300**	**323**	**325**	**340**	**361**	**394**	**407**	**393**	**434**	**423**	**435**	**345**
Protection Level	$\Gamma = 0$	ABI	0	1	1	0	1	2	0	0	0	0	1	1	0	0	**1**
		ATL	3	11	6	8	1	1	2	0	1	0	0	0	0	0	**2**
		NOB	37	39	28	9	6	3	1	0	0	0	0	0	0	0	**9**
		POL	18	4	7	2	2	0	1	0	0	0	0	0	0	0	**2**
		Avg	**15**	**14**	**11**	**5**	**3**	**2**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	4
	$\Gamma = 1$	ABI	100	99	97	92	92	92	96	94	82	94	88	78	77	74	**90**
		ATL	99	86	86	76	70	64	54	49	57	45	43	49	40	60	**63**
		NOB	88	91	84	81	74	75	77	52	62	56	64	59	59	81	**72**
		POL	98	75	82	75	67	64	57	56	69	66	64	71	68	60	**69**
		Avg	**96**	**88**	**87**	**81**	**76**	**74**	**71**	**63**	**68**	**65**	**65**	**64**	**61**	**69**	**73**
	$\Gamma = 2$	ABI	100	100	100	99	100	100	100	100	100	100	100	100	100	100	**100**
		ATL	100	99	97	99	98	99	99	99	95	97	97	99	97	100	**98**
		NOB	100	100	99	99	99	100	98	99	98	100	99	99	99	99	**99**
		POL	99	98	99	99	97	100	97	100	91	98	98	96	94	98	**97**
		Avg	**100**	**99**	**99**	**99**	**99**	**100**	**99**	**100**	**96**	**99**	**99**	**99**	**98**	**99**	99
	$\Gamma = 3$	ABI	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		ATL	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		NOB	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		POL	100	100	100	100	100	100	100	100	100	100	100	100	100	99	**100**
		Avg	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	100

Table 3.11: Computing times for the two-phase heuristic with $z_L = |V^0|$, $z_M = 2$, $z_H = 1$.

		$\|R\|$	5	6	7	8	9	10	12	14	16	18	20	24	28	32	**Avg**
Computing Time Phase I (s)	$\Gamma = 0$	Abilene	0	0	0	0	1	5	6	7	1	2	2	2	1	4	**2**
		Atlanta	0	0	0	0	0	0	0	0	0	0	1	1	14	15	**2**
		Nobel-us	0	0	0	0	0	0	0	300	0	0	9	13	14	95	**31**
		Polska	0	0	0	0	0	0	0	0	1	1	1	8	300	59	**26**
		Avg	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**77**	**1**	**1**	**3**	**6**	**82**	**43**	**15**
	$\Gamma = 1$	Abilene	0	0	0	0	0	0	0	0	0	0	0	1	300	300	**43**
		Atlanta	0	0	0	0	0	0	1	2	300	300	300	300	300	300	**129**
		Nobel-us	0	0	0	0	0	1	1	8	300	300	300	300	300	300	**129**
		Polska	0	0	0	0	0	0	0	300	300	300	300	300	300	300	**150**
		Avg	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**77**	**225**	**225**	**225**	**225**	**300**	**300**	**113**
	$\Gamma = 2$	Abilene	0	0	0	1	0	1	0	1	13	13	22	47	300	272	**48**
		Atlanta	0	0	0	1	2	3	300	300	300	300	300	300	300	300	**172**
		Nobel-us	0	0	0	300	300	300	300	300	300	300	300	300	300	300	**236**
		Polska	0	1	300	300	300	300	300	300	300	300	300	300	300	300	**257**
		Avg	**0**	**0**	**75**	**150**	**151**	**151**	**225**	**225**	**228**	**228**	**230**	**237**	**300**	**293**	**178**
	$\Gamma = 3$	Abilene	0	0	0	1	0	1	1	1	2	2	2	34	68	80	**14**
		Atlanta	0	1	1	5	13	7	6	18	300	300	300	300	300	300	**132**
		Nobel-us	0	0	0	1	3	3	300	300	300	300	300	300	300	300	**172**
		Polska	0	1	300	139	300	300	300	300	300	300	300	300	300	300	**246**
		Avg	**0**	**0**	**76**	**37**	**79**	**78**	**152**	**155**	**226**	**226**	**226**	**233**	**242**	**245**	**141**
Computing Time Phase II (s)	$\Gamma = 0$	Abilene	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
		Atlanta	0	0	0	0	0	0	1	2	3	9	5	13	107	17	**11**
		Nobel-us	0	0	0	0	0	0	1	1	8	1	1	2	2	2	**1**
		Polska	0	0	0	0	1	0	1	1	1	1	1	0	1	1	**1**
		Avg	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**3**	**3**	**2**	**4**	**28**	**5**	**3**
	$\Gamma = 1$	Abilene	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
		Atlanta	0	0	0	0	0	0	0	0	0	0	0	9	0	0	**1**
		Nobel-us	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
		Polska	0	0	0	0	0	0	0	0	0	0	0	1	0	0	**0**
		Avg	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**2**	**0**	**0**	**0**
	$\Gamma = 2$	Abilene	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
		Atlanta	0	0	0	0	1	0	0	0	0	0	0	0	0	0	**0**
		Nobel-us	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
		Polska	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
		Avg	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
	$\Gamma = 3$	Abilene	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
		Atlanta	0	0	0	0	0	0	0	1	0	0	0	0	0	0	**0**
		Nobel-us	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
		Polska	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
		Avg	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
Computing Time Phase I+II (s)	$\Gamma = 0$	Abilene	0	0	0	0	1	5	6	7	1	2	2	2	1	4	2
		Atlanta	0	0	0	0	0	0	1	2	3	9	6	14	121	32	13
		Nobel-us	0	0	0	0	0	0	1	301	9	1	10	15	16	97	32
		Polska	0	0	0	0	1	0	1	1	2	1	2	8	301	61	27
		Avg	**0**	**0**	**0**	**0**	**0**	**1**	**2**	**78**	**3**	**3**	**5**	**10**	**110**	**48**	**19**
	$\Gamma = 1$	Abilene	0	0	0	0	0	0	0	0	0	0	0	1	300	300	**43**
		Atlanta	0	0	0	0	0	0	2	2	300	300	300	309	300	300	**130**
		Nobel-us	0	0	0	0	0	1	1	8	300	300	300	300	300	300	**129**
		Polska	0	0	0	0	1	1	1	300	300	300	300	301	300	300	**150**
		Avg	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**78**	**225**	**225**	**225**	**228**	**300**	**300**	**113**
	$\Gamma = 2$	Abilene	0	0	0	1	0	1	0	1	13	13	22	47	300	272	**48**
		Atlanta	0	0	0	1	3	4	300	300	300	300	300	300	300	300	**172**
		Nobel-us	0	0	0	300	300	300	300	300	300	300	300	300	300	300	**236**
		Polska	0	1	300	300	300	300	300	300	300	300	300	300	300	300	**257**
		Avg	**0**	**0**	**75**	**151**	**151**	**151**	**225**	**225**	**228**	**228**	**231**	**237**	**300**	**293**	**178**
	$\Gamma = 3$	Abilene	0	0	0	1	0	1	1	1	2	2	2	34	68	80	**14**
		Atlanta	0	1	1	5	13	7	6	20	300	300	300	300	300	300	**132**
		Nobel-us	0	0	0	1	4	3	300	300	300	300	300	300	300	300	**172**
		Polska	0	1	300	139	300	300	300	300	300	300	300	300	300	300	**246**
		Avg	**0**	**0**	**76**	**37**	**79**	**78**	**152**	**155**	**226**	**226**	**226**	**234**	**242**	**245**	**141**

the exact approach, this stresses the superiority of the heuristic method in this regard as, given a much tighter time frame, it can usually produce better solutions.

In general, it is to say that with the selected parameter setting, the heuristic outperforms the exact approach dramatically, both with respect to the objective values and in comparison to the required computing time. We point out that, since we obtain non trivial solutions for all instances, the robust model is successful in providing solution values for different levels of protection. This way, this approach offers valuable alternative solutions to the ones obtainable in a deterministic setting, emphasizing the benefit of such model. We will comment on this in more detail, in the next subsection.

The adaptive heuristic Let us now consider the adaptive heuristic which, by design and differently from the two-phase heuristic, does not depend on a user-supplied parameter initialization. The corresponding results are shown in Table 3.12. As we can see there, the adaptive heuristic provides results which are comparable to those obtained with the two-phase heuristic employing the "winning" parameter setting, although, on average, the results for the former are slightly worse (by a mere 3%) than those for the latter. Nevertheless, there are cases where the adaptive heuristic improves over the two-phase heuristic with the "winning" parameter setting $z_L = |V^0|$, $z_M = 2$, and $z_H = 1$, such as for ABILENE with $|R| = 28$ and $\Gamma = 1$, as can be observed in Figure 3.13.

As is the case for the two-phase heuristic, non-trivial solutions can be obtained for all instances such that a multitude of different (with respect to Γ), robust solutions are available, each yielding a different trade-off between protection and objective value. While we will discuss the protection level in the next section, we point out that the adaptive heuristic requires a much larger time investment than the two-phase heuristic, i.e., for $\Gamma = 2$, on average 904 seconds are required, see Table 3.13, while the two-phase heuristic required only 178 seconds. By construction, the heuristic runs for at most 3600 seconds, as does the exact model. The reason for the increased time consumption is the iterative nature of the algorithm, i.e., the single phases are not only solved once but multiple times. In total, the time-limit was met in exactly one case, while the iteration limit was reached in 55 cases. We conclude that the adaptive heuristic yields solutions of similar quality as the two-phase heuristic, requiring more time but no user-provided input.

Overall, both heuristic methods dramatically outperform the exact approach, with the adaptive one being able to do so even without an *a priori* knowledge of a "good" parameter setting although at the cost of a higher computing time when compared to the two-phase heuristic. With respect to performance variability as encountered with the two-phase heuristic, we remark that the adaptive heuristic, of course, depends on some parameter setting as well. This is, for instance, the selection of the embeddings which are prevented in the subsequent iterations, i.e., Constraint (3.28) or the "decay" of the allowed distances in an embedding, Constraint (3.29a), respectively Constraint (3.29b). We do not focus on the performance of the adaptive heuristic with respect to these settings, as we believe that the here presented choice works sufficiently well in general. In

Table 3.12: Results (solution & protection values) for the adaptive heursitic.

		$\|R\|$	5	6	7	8	9	10	12	14	16	18	20	24	28	32	**Avg**
Objective Fct. Values	$\Gamma = 0$	Abi	322	381	438	450	480	480	472	472	566	507	507	564	513	658	**486**
		Atl	402	444	523	559	623	695	783	906	866	929	956	937	990	920	**752**
		Nob	346	393	441	532	570	629	734	770	879	842	836	1046	989	1106	**722**
		Pol	265	312	398	452	550	644	752	739	750	740	597	626	724	798	**596**
		Avg	**334**	**383**	**450**	**498**	**556**	**612**	**685**	**722**	**765**	**755**	**724**	**793**	**804**	**871**	**639**
	$\Gamma = 1$	Abi	68	68	125	194	194	194	194	194	288	288	288	360	381	381	**230**
		Atl	332	374	453	453	553	625	783	762	833	832	850	750	605	602	**629**
		Nob	294	341	389	480	480	473	586	607	636	630	609	728	750	748	**554**
		Pol	265	312	398	452	522	522	553	604	595	640	623	649	572	706	**530**
		Avg	**240**	**274**	**341**	**395**	**437**	**454**	**529**	**542**	**588**	**598**	**593**	**622**	**577**	**609**	**486**
	$\Gamma = 2$	Abi	68	68	125	194	194	194	194	194	288	288	288	256	277	277	**208**
		Atl	332	374	453	447	511	519	610	686	722	660	662	659	552	550	**553**
		Nob	294	294	342	366	402	402	402	395	395	405	401	459	397	467	**387**
		Pol	265	284	368	394	454	454	478	475	491	498	492	548	583	606	**456**
		Avg	**240**	**255**	**322**	**350**	**390**	**392**	**421**	**438**	**474**	**463**	**461**	**481**	**452**	**475**	**401**
	$\Gamma = 3$	Abi	68	68	125	126	126	126	126	126	220	220	220	220	220	220	158
		Atl	332	332	411	447	411	447	459	560	560	560	560	574	580	554	485
		Nob	225	225	260	282	320	320	320	312	320	320	337	420	419	456	324
		Pol	189	244	294	308	369	369	412	437	481	472	472	472	517	585	402
		Avg	**204**	**217**	**273**	**291**	**307**	**316**	**329**	**359**	**395**	**393**	**397**	**422**	**434**	**454**	**342**
Protection Values	$\Gamma = 0$	Abi	1	0	1	2	1	1	3	2	0	1	0	1	0	0	**1**
		Atl	75	72	75	1	1	0	2	0	0	0	1	0	0	0	**16**
		Nob	59	49	1	1	0	0	0	0	0	0	0	0	0	0	**8**
		Pol	96	86	23	2	3	0	0	1	0	0	1	1	0	0	**15**
		Avg	**58**	**52**	**25**	**2**	**1**	**0**	**1**	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**10**
	$\Gamma = 1$	Abi	98	98	98	97	99	99	99	94	89	94	87	71	62	80	**90**
		Atl	91	84	91	90	81	68	54	63	39	42	28	42	50	31	**61**
		Nob	88	84	81	82	75	85	60	56	42	52	54	57	49	61	**66**
		Pol	90	81	75	72	68	68	59	66	69	61	62	59	63	57	**68**
		Avg	**92**	**87**	**86**	**85**	**81**	**80**	**68**	**70**	**60**	**62**	**58**	**58**	**56**	**57**	**71**
	$\Gamma = 2$	Abi	100	100	100	100	100	100	100	100	98	98	98	99	100	100	**100**
		Atl	99	98	99	100	99	99	99	96	96	100	98	98	98	99	**99**
		Nob	100	100	99	99	99	99	99	99	100	97	99	100	98	99	**99**
		Pol	99	99	97	95	100	100	95	98	95	96	99	98	94	96	**97**
		Avg	**100**	**99**	**99**	**99**	**100**	**100**	**98**	**98**	**97**	**98**	**99**	**99**	**98**	**99**	**99**
	$\Gamma = 3$	Abi	100	100	100	100	100	100	100	100	100	100	100	100	99	99	**100**
		Atl	100	100	100	100	100	100	100	100	100	100	100	100	99	100	**100**
		Nob	100	100	100	100	100	100	100	100	100	100	100	100	100	100	**100**
		Pol	100	100	100	100	100	100	100	100	99	100	100	100	100	99	**100**
		Avg	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**	**100**

Table 3.13: Results (computing time) for the adaptive heursitic.

Computing Times (s)	$\|R\|$	5	6	7	8	9	10	12	14	16	18	20	24	28	32	**Avg**
$\Gamma = 0$	Abi	6	14	12	37	29	33	13	34	42	45	49	56	62	67	**36**
	Atl	9	9	11	36	59	67	99	207	267	334	350	355	464	464	**195**
	Nob	8	10	8	8	8	9	44	128	137	211	206	465	486	614	**167**
	Pol	6	7	16	12	27	51	125	97	110	116	90	97	94	119	**69**
	Avg	**7**	**10**	**12**	**23**	**31**	**40**	**70**	**116**	**139**	**177**	**174**	**244**	**276**	**316**	**117**
$\Gamma = 1$	Abi	6	6	6	8	7	8	8	9	7	7	9	41	38	36	**14**
	Atl	11	9	10	10	32	17	127	180	522	2059	3180	3484	623	1231	**821**
	Nob	8	9	9	9	8	11	13	45	310	309	611	613	1518	613	**292**
	Pol	6	6	12	30	34	62	68	213	308	551	569	2798	1891	1973	**609**
	Avg	**8**	**7**	**9**	**14**	**20**	**24**	**54**	**112**	**287**	**732**	**1093**	**1734**	**1017**	**963**	**434**
$\Gamma = 2$	Abi	7	11	9	15	10	11	11	11	31	33	31	911	912	910	**208**
	Atl	10	9	10	10	29	18	1664	3098	3445	2137	3042	3361	622	613	**1291**
	Nob	8	8	8	308	308	309	309	308	309	311	308	308	308	308	**244**
	Pol	6	6	6	367	307	307	2114	3320	3318	3357	3600	2716	3466	3320	**1872**
	Avg	**8**	**9**	**8**	**175**	**164**	**161**	**1024**	**1684**	**1776**	**1459**	**1745**	**1824**	**1327**	**1288**	**904**
$\Gamma = 3$	Abi	6	6	6	306	307	307	307	308	612	607	607	607	1808	1209	**500**
	Atl	9	9	10	10	10	15	15	21	1222	1521	1519	1819	2136	1820	**724**
	Nob	8	8	8	8	8	8	307	308	308	308	308	612	1214	1211	**330**
	Pol	6	6	308	607	2173	1528	2020	3019	2819	3316	3314	3315	3389	2717	**2039**
	Avg	**7**	**7**	**83**	**233**	**625**	**465**	**662**	**914**	**1240**	**1438**	**1437**	**1588**	**2137**	**1739**	**898**

particular, when comparing to the two-phase heuristic, we do not observe a performance deterioration when applying the heuristic to the deterministic case with worst-case data, which supports our claim.

Considerations on the protection level

We now focus on the empirical protection level of the different approaches. Recall that, when adopting worst case or average data, the protection level is by definition always equal to 100% for the former whereas, as it is, in practice, almost always equal to 0% for the latter.

Let us first consider the exact Γ-robust approach for $\Gamma \in \{1, 2, 3\}$, as reported in Table 3.7. We remark that the empirical protection level is induced by the feasibility of a solution and not by its optimality. Therefore, it is reasonable to measure the former for all the solutions provided by the method, regardless of them being optimal or not. Quite interestingly, we observe that, although the empirical protection level for $\Gamma = 1$ is not very high (being equal to, on average, 82%), it already reaches a value of, on average, 99.6% already for $\Gamma = 2$.

Similar observations can be drawn for both of our heuristics, as reported in Table 3.10 and in Table 3.12. For $\Gamma = 0$ (the case with average data), the empirical protection level is, as expected, very small (3% on average for the two-phase heuristic and 10% for the adaptive one). For $\Gamma = 1$, it is still not very high for both methods, being close to 71% on average for both. Differently, for $\Gamma = 2$ and $\Gamma = 3$, we obtain solutions with very few violations and a higher empirical protection level equal to, respectively, 99% and 100%

on average (again for both heuristics).

In particular, the results for the cases $\Gamma = 2$ and $\Gamma = 3$ are of interest. As mentioned before, their empirical protection level is comparable to that obtained by the deterministic approach with worst-case data but their objective value is better. In this sense, the robust model directly yields an improvement over the deterministic one.

For each instance, the Γ-robust problem allows to find a range of solutions in between the extreme cases given by the deterministic problem. This is especially important for the application, where the most suitable solution can be selected, e.g., a solution which is more protected than the one obtained with average data but less conservative than the one obtained with worst-case data. In principle, even more different solutions can be obtained, for example, by varying the Γ for the different constraints, by, in the heuristics, considering $\Gamma > 0$ in the second phase, or by considering fractional values of Γ. This way, the Γ-robust problem offers a degree of freedom which can hardly be matched by the deterministic problem, emphasizing the importance of taking data uncertainty into account and modeling it accordingly.

Results on larger instances

To better assess how our algorithms scale on instances of larger size, we report on a set of experiments carried out on the eight physical networks taken from the Internet Topology Zoo [83], see Table 3.3. Due to their size (larger than those used in the previous experiments), we consider up to $|R| = 50$ VN requests. As the exact approach struggles with the time limit, already for the smaller instances, and the adaptive heuristic requires more than 1000 seconds in a number of cases, we only analyze the two-phase heuristic.

We experiment with the parameter setting $z_L = |V^0|, z_M = 2,\ z_H = 1$ which we have found to perform best in the previous experiments, with $\Gamma = 1$ in phase one and $\Gamma = 0$ in phase two. The results are reported in Table 3.14. Note that, in the table, the computing time accounts for the total time spent in the two phases, neglecting the preprocessing time invested to compute the all pairs shortest paths which are needed for the distance-bounding constraints in phase one.

The table shows that our two-phase algorithm can solve reasonably well VNE instances with large physical networks (COGENTCO contains $|V^0| = 199$ nodes) with up to $|R| = 40$ simultaneous VN requests. The method starts to fail for $|R| = 45$ as the solver is unable to find a nonzero solution to the phase one subproblem within the time limit.

The table also illustrates an interesting phenomenon.It shows that, although both subproblems get harder, as one would expect, when $|R|$ increases, the phase one subproblem gets substantially easier when the size of the physical network $|V^0|$ grows. This is, most likely, a feature of the underlying multi-knapsack structure, due to which the introduction of more physical nodes only makes node capacity a more abundant resource, without complicating the structure of the problem too much. Interestingly, the situation is reversed for the phase two subproblem, which gets harder for physical networks of larger size. This is, possibly, a consequence of the network topologies still playing a large role in it, so that, having a physical network of increased size does not directly

translate into a problem which is easier to solve.

These two opposite behaviors are, quite interestingly, somewhat complementary in that, by increasing the value of $|V^0|$, while the phase one subproblem gets easier, the phase two subproblem gets harder. Overall, we end up with a situation where, if one of the two subproblems is not solved to optimality, then the other one (in most of the cases) is, thus obtaining an, overall, still effective algorithm.

All in all, the two-phase heuristic is, among all here presented approaches, the best scaling algorithm, allowing to obtain solutions even for larger networks. However, in the case that even larger instances are desired to be solved, the heuristic has to be improved, respectively fine-tuned. That is, as it relies on MILP, the solution process of the subproblems could be improved, e.g., via cutting planes, or the subproblems them-self could be tackled via custom heuristics.

Recommendations

In this subsection, we have investigated the Γ-robust VNE problem. We have observed that an exact solution approach is promising but fails at the larger problem instances as, in many cases, no non-zero solutions can be found. The exact approach was complemented by two heuristic ones. We have evaluated a two-phase heuristic, which relies heavily on a predetermined parameter setting and an adaptive one which, at a larger time investment, can do so without. Both heuristics have been performing very well in terms of solution quality and runtime.

As a consequence of the results that we observed in our experiments, we advise to resort to the exact MILP formulation for the nominal case of VNE only with no more than $|R| = 20$ requests and to the exact Γ-robust formulation only for $\Gamma = 1$ and with up to $|R| = 10$ requests. For all the other cases, we suggest to employ the two-phase heuristic (which, among the two proposed algorithms, is definitely the faster one) with parameters $z_L = |V^0|, z_M = 2$, and $z_H = 1$. In case substantial differences in solution quality can be observed by experimenting with other parameter values (such differences could be substantial, compare Table 3.9) and a parameter tuning cannot be carried out in a preprocessing step, we advise the adoption of the computationally more demanding but, without a good guess on a suitable parameter choice, more stable, adaptive heuristic.

As to the choice of Γ, when aiming for a very high, i.e., larger than 95%, protection level, we advise, based on our experiments, to select $\Gamma = 2$ for the node capacity constraints, while (possibly) letting $\Gamma = 0$ for the link capacity ones.

3.6 Conclusion and outlook

We conclude this chapter by a brief recapitulation of our results and a discussion of open questions and further research directions.

In this chapter, we have extensively studied the VNE problem. We started by pointing

Table 3.14: Results on 8 Internet Topology Zoo instances of increasing size $|V^0|$ for $\Gamma = 1$. Entries are rounded to the nearest integer.

	$\|V^0\|$	$\|R\|$	8	9	10	12	14	16	18	20	24	28	32	35	40	45	50	**Avg**
Objective Fct.	20	Fatman	328	302	331	315	341	450	417	429	370	413	523	468	434	441	391	**397**
	30	Digex	405	409	439	422	422	424	487	476	450	445	422	235	284	305	0	**375**
	40	Cernet	396	477	477	664	679	820	704	692	724	811	724	911	682	0	0	**584**
	50	Bellsouth	366	379	478	459	497	544	582	734	571	763	827	789	864	758	0	**574**
	72	Intellifiber	385	428	442	441	538	493	496	579	583	620	524	576	599	0	290	**466**
	83	RedBestel	280	379	349	380	407	416	443	458	505	543	565	522	540	0	509	**420**
	112	Deltacom	343	408	472	541	687	547	619	632	806	803	695	617	833	996	0	**600**
	199	Cogentco	365	459	496	534	631	566	684	740	702	778	793	779	837	0	0	**558**
		Avg	**359**	**405**	**436**	**470**	**525**	**533**	**554**	**593**	**589**	**647**	**634**	**612**	**634**	**313**	**149**	**497**
O.Gap (Ph.I)	20	Fatman	0	0	0	0	4	3	10	15	23	50	44	48	78	78	78	**29**
	30	Digex	0	0	0	0	0	0	3	6	19	64	184	821	639	472	∞	**158**
	40	Cernet	0	0	0	0	0	0	0	12	17	38	53	50	68	∞	∞	**18**
	50	Bellsouth	0	0	0	0	0	0	0	0	0	0	5	6	8	71	∞	**6**
	72	Intellifiber	0	0	0	0	0	0	0	0	0	0	0	8	18	∞	684	**51**
	83	RedBestel	0	0	0	0	0	0	0	0	0	0	0	0	0	∞	78	**6**
	112	Deltacom	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	**0**
	199	Cogentco	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
		Avg	**0**	**0**	**0**	**0**	**1**	**0**	**2**	**4**	**7**	**19**	**36**	**117**	**101**	**124**	**168**	**33**
O.Gap (Ph.II)	20	Fatman	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
	30	Digex	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
	40	Cernet	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
	50	Bellsouth	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	**0**
	72	Intellifiber	0	0	0	0	0	0	0	0	0	0	8	0	0	0	0	**1**
	83	RedBestel	0	0	0	0	0	0	0	0	0	6	0	11	2	0	0	**1**
	112	Deltacom	0	0	0	0	0	0	0	0	0	8	3	0	10	5	∞	**2**
	199	Cogentco	0	0	0	0	0	5	0	0	10	8	7	11	11	∞	∞	**4**
		Avg	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**1**	**3**	**2**	**3**	**3**	**1**	**0**	**1**
Comp. Time (s)	20	Fatman	0	1	10	300	300	301	301	300	301	301	301	301	301	2	2	**201**
	30	Digex	2	6	3	9	11	30	328	338	317	330	307	300	300	2	0	**152**
	40	Cernet	2	2	2	2	6	9	72	315	318	327	321	326	327	299	299	**175**
	50	Bellsouth	1	1	1	3	5	4	7	9	140	256	373	331	372	315	299	**141**
	72	Intellifiber	8	9	10	28	21	44	107	66	124	265	600	458	404	0	3	**143**
	83	RedBestel	6	6	20	12	33	56	86	114	236	333	181	600	600	0	202	**166**
	112	Deltacom	8	15	18	42	78	118	115	96	163	309	312	295	359	553	600	**205**
	199	Cogentco	11	84	51	78	160	304	84	164	307	304	310	311	321	342	313	**210**
		Avg	**5**	**16**	**14**	**59**	**77**	**108**	**137**	**175**	**238**	**303**	**338**	**365**	**373**	**189**	**215**	**174**
Prot. Level	20	Fatman	99	100	98	98	95	92	98	96	91	97	94	96	94	90	97	**96**
	30	Digex	100	100	100	100	99	98	93	97	93	92	100	100	100	100	100	**98**
	40	Cernet	97	100	100	100	100	97	92	90	96	99	94	90	92	100	100	**96**
	50	Bellsouth	100	99	100	99	97	100	98	96	95	88	91	95	92	98	100	**97**
	72	Intellifiber	100	99	99	99	99	99	99	93	98	96	84	93	98	100	100	**97**
	83	RedBestel	100	100	99	98	99	99	100	97	97	95	93	97	85	100	99	**97**
	112	Deltacom	98	99	99	99	98	99	99	98	97	98	97	95	95	95	93	**97**
	199	Cogentco	100	99	100	98	100	100	100	100	99	99	99	98	95	99	97	**99**
		Avg	**99**	**100**	**99**	**99**	**98**	**98**	**97**	**96**	**96**	**96**	**94**	**96**	**94**	**98**	**98**	**97**

out the economic importance of VNE, especially on the basis of large scale telecommunication systems as, e.g., the Internet. We have then formalized the VNE problem and have presented an MILP formulation for the problem. The MILP formulation is extendable in the sense that it can be easily adapted to many different variations of the problem as, for example, it can be extended to include rent-at-bulk aspects. We have observed that the VNE problem can naturally be split into two phases, which directly yields a heuristic approache to the problem.

In the following, we have focused on the computational complexity of VNE. We have shown that the general VNE problem is strongly $\mathcal{NP}$-hard and inapproximable. We have also analyzed the complexity of the two subproblems of VNE and we have discussed the hardness of the problem which arises when a dimension of the input is fixed. Furthermore, we have considered dynamic programming approaches and have focused on the special case where all virtual networks are stars.

In the following, we have focused on the VNE problem under data uncertainty. Based on a chance-constrained formulation where node and traffic demands of the virtual networks are assumed to be random variables, we have proposed an exact, Γ-robust Mixed-Integer Linear Programming (MILP) formulation which allows to find solutions with large profits that are guaranteed to be feasible with a high probability. Based on this formulation, we have also introduced two MILP-based Γ-robust heuristics: a two-phase heuristic based on a given parameter choice and an adaptive one.

In the last section, we have discussed computational experiments on the VNE problem, considering both, the deterministic VNE problem and the VNE problem with data uncertainty. The experiments indicate that, in both cases, the exact approaches become computationally challenging for instances with an increasing number of virtual network requests, while both heuristics provide high quality solutions even for larger problems. Following, we have discussed under which conditions which of the heuristics, respectively the exact approach, is preferable, especially taking the "protection" of a solution in the robust setting into account.

We are convinced that the VNE problem will remain relevant in the foreseeable future. As the popularity of telecommunication services will rise even further, and more and more new technologies surface, the potential benefit of virtualization schemes can only increase as well. Therefore, we point out some further research directions.

Since the VNE problem is very application driven, research on the "practical" side of the problem is very relevant. While this includes the investigation and the improvement of (further) heuristic approaches on the one hand, enhancements on exact, e.g., MILP, approaches are certainly beneficial on the other hand, as well. In this context, we refer to Subsection 3.5.3 where we have analyzed the scalability of our two-phase heuristic. There, the heuristic started to fail since no primal solution could be found in the first subproblem. Note that, since our heuristics rely on splitting the VNE problem into different phases and solving each phase by an MILP, benefits for the exact MILP formulation directly carry over to the heuristic approach and vice versa. Such enhancements can, for instance, come from a better understanding of the underlying polyhedron, that

is, from the knowledge of strong cutting planes, etc..

As a first step, we briefly discuss two examples of such cutting planes. The first one concerns the node-mapping part of VNE and relates to the knapsack like structure of this part. For $r \in R$ and $i \in V^0$ let $\mathcal{C}_i^r \subseteq V^r$ denote a subset of items, respectively nodes from a single VN, forming a *cover* for the physical node i, i.e.,

$$\sum_{v \in \mathcal{C}_i^r} \omega_v^r > b_i. \tag{3.34}$$

Clearly, the (standard) *cover-inequality*

$$\sum_{v \in \mathcal{C}_i^r} x_{iv}^r \leq |\mathcal{C}_i^r| - 1 \tag{3.35}$$

is valid for VNE. However, this inequality implicitly assumes that $y^r = 1$. Hence, we can down-lift y^r into this constraint and obtain

$$\sum_{v \in \mathcal{C}_i^r} x_{iv}^r - (|\mathcal{C}_i^r| - 1)\, y^r \leq 0. \tag{3.36}$$

Clearly, the same concept can be extended to include multiple physical nodes and to consider items from different VNs.

The second example concerns the link-mapping part of the VNE problem and relates to the network design like structure of this part, compare Subsection 1.3.2. Given a (node) cut $S^0 \subseteq V^0$, all the traffic on the arcs of this cut must not exceed its capacity, i.e., the traffic induced by the node mapping must not exceed

$$B(S^0) := \sum_{\substack{ij \in A: \\ i \in S^0, \\ j \in V^0 \setminus S^0}} k_{ij}. \tag{3.37}$$

The corresponding *Cutset Inequality* writes:

$$\sum_{r \in R} \sum_{v,w \in V^r} d_{vw}^r \left(\sum_{i \in S^0} x_{vi}^r - \sum_{i \in S^0} x_{wi}^r \right) \leq B(S^0). \tag{3.38}$$

For given (r, v, w), if v and w are mapped into S^0, the traffic value d_{vw}^r does not appear on the cut. If v is embedded in S^0 and w is not, the traffic value is accounted for. If neither v nor w is placed in S^0 the traffic also disappears. If w is mapped to S^0 but v is not, the traffic value which is accounted for gets a negative sign, which makes the inequality weak but still valid. The inequality is valid, even if not all possible pairs (r, v, w) are accounted for.

To derive a strong inequality, one may searche for an appropriate selection of (r, v, w), where all differences yield a (maximum) positive contribution to the left hand side.

Note that in principle, the inequality is a knapsack constraint, such that, e.g., cover inequalities, can be derived from it.

For another approach regarding cutting planes, respectively the underlying polyhedron of the VNE problem, we refer to the master thesis of Rosendahl [112]. In this work, the author approached the VNE problem via a Bender's Decomposition, decoupling the virtual node and link embedding. In this context, the author showed, for example, how inequalities similar to the well known cutset inequalities, see above and compare Section 2.3, can be derived. As a next step, these findings should be extended and evaluated computationally.

Another interesting direction for future research is the development of alternative, exact algorithms for VNE. While, in this work, we have observed that many special cases of the VNE problem are theoretically hard, there may be other special cases in which custom algorithms may be beneficial in comparison to, e.g., mixed integer linear programming and, or heuristic approaches. In this context, we also mention approximation algorithms as a future research topic. As we have seen, the general problem is inapproximable, however, there may be special cases for which efficient approximation- or even exact algorithms can be found. This is, for example, the case for the virtual cluster embedding, as mentioned by Rost et al. [113], for which efficient algorithms are known.

We conclude by commenting on our dataset. For our experiments, we relied on extended test instances from the SNDLIB [100] and the Internet Topology Zoo [83]. In particular, we took the there specified networks as substrate networks and generated the virtual network requests at random as no such real-life data is available. Naturally, real-life data would greatly benefit the relevance of our simulations and boost the interest of both parties, researchers and practitioners into the topic. In particular, depending on the structure of such data, it could also be used to derive algorithms, especially tailored to certain classes of applications, which would nicely complement the general approach presented in this work. Thus, as a final remark, we devise the creation or exhibition of such data, hopefully in close collaboration with the industry, as one of the primary future research goals. As a first step in this direction, the datasets used in this work are made available for the public. They can be download from the website [39]. We hope that this data can be extended and adapted where required and, in general, that is helpful in future research.

Concluding remarks and outlook

In this thesis, we have studied two optimization problems, the Network Design Problem with Compression (NDPC) and the Virtual Network Embedding Problem (VNE). In both cases, our interest into the topic was motivated by the importance of these problems within the telecommunication industry. We have presented a broad range of results, in general obtained by applying methods and concepts from the field of mathematical, respectively combinatorial optimization to these problems. In our research, and especially in the VINO project [122] (see also the Introduction), we have aimed to provide new insights into these problems, both from a theoretical point of view and with respect to practical applications.

For this purpose, we have devised Mixed Integer Linear Programming formulations for both problems, based on which further research was conducted. In particular, for the NDPC problem, we have concentrated on the underlying polyhedron of this formulation with the goal to enhance the solution process via mixed inter linear programming. Subsequently, we have focused on the computational complexity of the problems. For NDPC, we have investigated the Compressor Placement Problem, showing that the added (de-) compression functionality makes, in some sense, the NDPC problem even more difficult than the Network Design Problem. For the VNE problem, we have considered special cases of the problem setting, showing that VNE stays $\mathcal{NP}$-hard even when input dimensions of the problem are fixed. Furthermore, we have discussed how data uncertainty can be accounted for within our models, deriving Γ-robust problem formulations for both problems. In this context, we have focused on the concept of two-source uncertainty for the NDPC problem, while for VNE, we have concentrated on heuristic approaches to the Γ-robust problem.

Both chapters, respectively our research on both problems, are concluded by a report on computational experiments. Within these, a practical evaluation of our results is given, indicating how our results can translate into benefits for the application.

While problem specific conclusions can be found at the end of each chapter, here, we give some general, final remarks. On the whole, it is to say that the field of mathematical optimization and in particular the area of combinatorial optimization provides tools very well fitted to approach the NDPC as well as the VNE problem. That is, the field allows to aggregate real-world decision making into structured mathematical abstractions, clearly defining relevant decisions, their interactions, and finally, their benefit with respect to an "optimal" decision. From a scientific point of view, many different

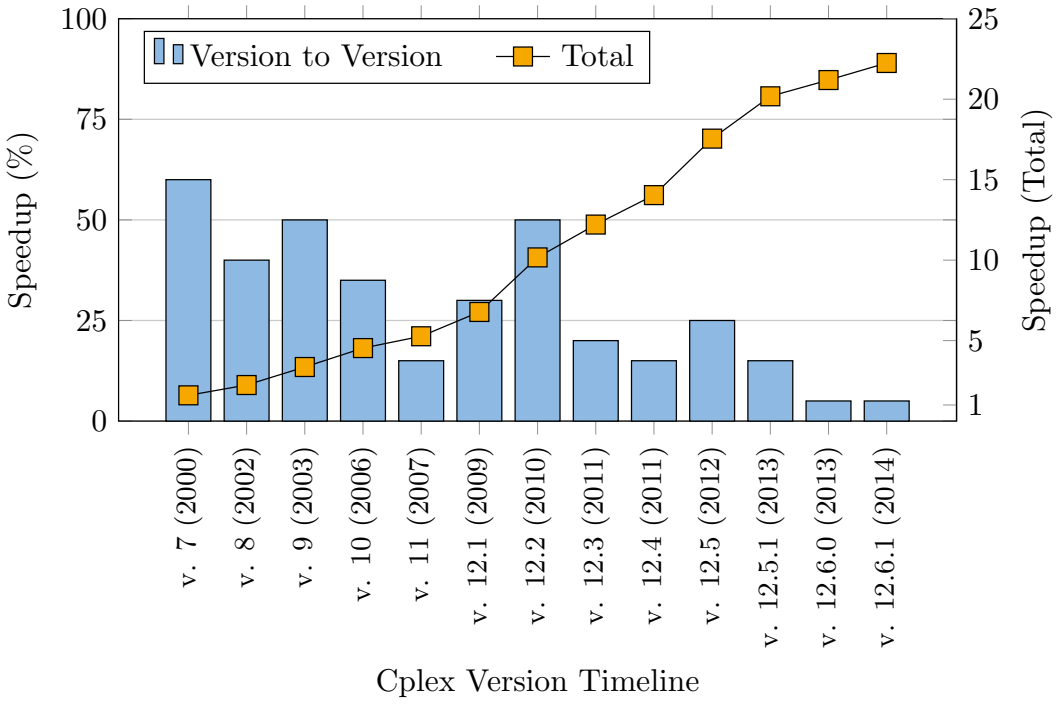


Figure 4.1: CPLEX software speedup: from version to version (blue) and in total (orange). Data is taken from the CPLEX web-page [45].

approaches to such models, respectively problems, are possible. In this thesis, we have encountered both, rather theoretical results, as, for example, results in polyhedral or complexity theory as well as rather practical results as, for example, the considerations on data uncertainty and on heuristic approaches. We believe that this indicates the "richness" of the field. Naturally, many of our results can, in the end, be translated into "better" algorithms to tackle these problems.

In general, such algorithms can be categorized into two groups, i.e., into exact and heuristic problem approaches. While the benefit of an exact problem approach is apparent, we have observed that our exact approaches do only scale well up to a certain problem size. On the contrary, (fast) heuristic approaches can be devised for practically any problem size. In this context, further potential research should strengthen both approaches: i.e., it should improve both the exact and the heuristic solution algorithms. As is typical for this field, such progress can be achieved in two ways.

On the one hand, improvements can be obtained by focusing on the problems directly, as was, for example, done in this thesis. On the other hand, progress can be made by developing general, problem independent concepts and frameworks. An example for such "black-box" development is the continuous and (almost) problem independent improvement in MILP software, see Figure 4.1 where the speedup of CPLEX [45] over different software versions is visualized.

However, progress in the mathematical foundation is only one factor in future devel-

opments as progress in the telecommunication industry will in turn affect the required mathematical models and also the underlying theory and software. This way, there will always be room for improvement and hence, need for future research such that the field of mathematical optimization in general and the NDPC, respectively the VNE problem, in particular will remain attractive for both practitioners and scientists.

We conclude this thesis by a short remark on problem data. Realistic problem data for NDPC, respectively for VNE, is hard to obtain, non the last as such data is usually not published in the fear of providing business competitors critical insights into a companies market position. Still, the relevance of research on these problems can largely benefit from such data, especially since the mathematical optimization community works worldwide which makes an efficient exchange of information difficult. Sadly, for NDPC and for VNE, there is no commonly available data pool. Usually, since both problems are generalizations of the NDP problem, the corresponding databases for NDP, e.g., the SNDLIB or the Internet Topology Zoo, are suitably extended to obtain instances for NDPC or VNE. For the future, we hope that this data can be generated and that the research community can agree upon a common dataset, such that a common foundation for research on these problems can be devised. As a first step in this direction, the datasets used in this work are made available for the public. They can be download from the website [39]. We hope that this data can be extended and adapted where required and, in general, that is helpful in future research.

List of figures

Page

Page

List of tables

List of algorithms

Problem glossary

3-PART 3-Partition Problem. 156

CPP Compressor Placement Problem. 73

EDPP Edge Disjoint Path Problem. 144

EQUICUT Minimum Cut into Equal Sized Subsets. 151

HSP Hitting Set Problem. 75

KP Knapsack Problem. 79, 151

MCMP Maximum Clique Minor Problem. 149

MKP Multiple Knapsack Problem. 144, 150

MKP-G Multiple Knapsack Problem with Grouped Items. 141

NDP Network Design Problem. 9

NDPC Network Design with Compression Problem. 40

PART Partition Problem. 154

STAB Stable Set Problem. 146

UMCF-AC Unsplittable Multi-Commodity Flow Problem with Admission Control. 142

VNE Virtual Network Embeddign Problem. 137

VNES Virtual Network Embeddign Problem of a (single) Star. 154

VNES-U Virtual Network Embeddign Problem of a (single) Uniform Star. 157

Bibliography

[1] Y. K. Agarwal. k-partition-based facets of the network design problem. *Networks*, 47(3):123–139, 2006. (Cited on pages 46, 51, 57, and 125.)

[2] Y. K. Agarwal. Polyhedral structure of the 4-node network design problem. *Networks*, 54(3):139–149, 2009. (Cited on pages 46, 51, 57, and 125.)

[3] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows: theory, algorithms, and applications*. Prentice hall, Inc., Upper Saddle River, NJ, USA, 1993. (Cited on page 5.)

[4] A. Altın, E. Amaldi, P. Belotti, and M. Pınar. Provisioning virtual private networks under traffic uncertainty. *Networks*, 49(1):100–115, 2007. (Cited on pages 16 and 135.)

[5] E. Amaldi, S. Coniglio, A. M. C. A. Koster, and M. Tieves. On the computational complexity of the virtual network embedding problem. *Electronic Notes in Discrete Mathematics: Proc. of the International Network Optimization Conference (INOC)*, 52:213–220, 2016. (Cited on pages 2, 128, and 136.)

[6] AMPL, Version 20130906, R. Fourer, D. Gay, and B. W. Kernighan. The ampl book, 2002. URL: `ampl.com`. (Cited on pages 98 and 169.)

[7] A. Anand, A. Gupta, A. Akella, S. Seshan, and S. Shenker. Packet caches on routers: the implications of universal redundant traffic elimination. *ACM SIGCOMM Computer Communication Review*, 38(4):219–230, 2008. (Cited on page 38.)

[8] A. Anand, C. Muthukrishnan, A. Akella, and R. Ramjee. Redundancy in network traffic: findings and implications. *ACM SIGMETRICS Performance Evaluation Review*, 37(1):37–48, 2009. (Cited on page 38.)

[9] D. Andersen. Theoretical approaches to node assignment. Technical report, Computer Science Department, 2002. (Cited on page 136.)

[10] T. Anderson, L. Peterson, S. Shenker, and J. Turner. Overcoming the internet impasse through virtualization. *Computer*, 38(4):34–41, 2005. (Cited on page 134.)

[11] B. Awerbuch and Y. Azar. Buy-at-bulk network design. *Proc. of the IEEE Annual Symposium on Foundations of Computer Science*, pages 542–547, 1997. (Cited on page 10.)

[12] H. Ballani, P. Costa, T. Karagiannis, and A. Rowstron. Towards predictable datacenter networks. *ACM SIGCOMM Computer Communication Review*, 41(4): 242–253, 2011. (Cited on page 154.)

[13] F. Barahona and A. R. Mahjoub. On the cut polytope. *Mathematical programming*, 36(2):157–173, 1986. (Cited on page 71.)

[14] A. Ben-Tal and A. Nemirovski. Robust convex optimization. *Mathematics of operations research*, 23(4):769–805, 1998. (Cited on page 16.)

[15] A. Ben-Tal and A. Nemirovski. Robust solutions of uncertain linear programs. *Operations research letters*, 25(1):1–13, 1999. (Cited on page 16.)

[16] A. Ben-Tal and A. Nemirovski. Robust solutions of linear programming problems contaminated with uncertain data. *Mathematical programming*, 88(3):411–424, 2000. (Cited on page 16.)

[17] D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Mathematical programming*, 98(1-3):49–71, 2003. (Cited on pages 16, 21, 38, and 161.)

[18] D. Bertsimas and M. Sim. The price of robustness. *Operations research*, 52(1): 35–53, 2004. (Cited on pages 16, 21, 22, 38, and 161.)

[19] A. P. Bianzino, C. Chaudet, D. Rossi, and J.-L. Rougier. A survey of green networking research. *IEEE Communications Surveys & Tutorials*, 14(1):3–20, 2012. (Cited on page 37.)

[20] Bluecoat, Vendor. Wan optimization controller. URL: `www.bluecoat.com`. (Cited on page 39.)

[21] R. Bolla, R. Bruschi, F. Davoli, and A. Ranieri. Performance constrained power consumption optimization in distributed network equipment. *Proc. of the IEEE International Conference on Communications (ICC)*, pages 1–6, 2009. (Cited on page 37.)

[22] R. Bolla, R. Bruschi, F. Davoli, and F. Cucchietti. Energy efficiency in the future internet: a survey of existing approaches and trends in energy-aware fixed network infrastructures. *IEEE Communications Surveys & Tutorials*, 13(2):223–244, 2011. (Cited on page 37.)

[23] J. F. Botero, X. Hesselbach, M. Duelli, D. Schlosser, A. Fischer, and H. De Meer. Energy efficient virtual network embedding. *IEEE Communications Letters*, 16 (5):756–759, 2012. (Cited on page 135.)

[24] R. Braden, T. Faber, and M. Handley. From protocol stack to protocol heap: role-based architecture. *ACM SIGCOMM Computer Communication Review*, 33 (1):17–22, 2003. (Cited on page 134.)

[25] M. Brunner, H. Abramowicz, N. Niebert, and L. M. Correia. 4ward: A european perspective towards the future internet. *IEICE Transactions on Communications*, 93(3):442–445, 2010. (Cited on page 136.)

[26] C. Büsing and F. D'Andreagiovanni. A new theoretical framework for robust optimization under multi-band uncertainty. *Operations Research Proceedings: Proc. of the International Conference of the German Operations Research Society (GOR)*, pages 115–121, 2012. (Cited on page 16.)

[27] C. Büsing and F. D'Andreagiovanni. New results about multi-band uncertainty in robust optimization. *Proc. of the International Symposium on Experimental Algorithms (SEA)*, pages 63–74, 2012. (Cited on page 16.)

[28] J. Carapinha and J. Jiménez. Network virtualization: a view from the bottom. *Proc. of the ACM Workshop on Virtualized Infrastructure Systems and Architectures (VISA)*, pages 73–80, 2009. (Cited on page 136.)

[29] J. Chabarek, J. Sommers, P. Barford, C. Estan, D. Tsiang, and S. Wright. Power awareness in network design and routing. *Proc. of the IEEE International Conference on Computer Communications (INFOCOM)*, 2008. (Cited on page 38.)

[30] C. Chekuri and S. Khanna. A polynomial time approximation scheme for the multiple knapsack problem. *SIAM Journal on Computing*, 35(3):713–728, 2005. (Cited on page 151.)

[31] C. Chekuri, S. Khanna, and F. B. Shepherd. Edge-disjoint paths in planar graphs with constant congestion. *SIAM Journal on Computing*, 39(1):281–301, 2009. (Cited on page 145.)

[32] Y. Chen, J. Li, T. Wo, C. Hu, and W. Liu. Resilient virtual network service provision in network virtualization environments. *Proc. of the IEEE International Conference on Parallel and Distributed Systems (ICPADS)*, pages 51–58, 2010. (Cited on page 135.)

[33] X. Cheng, S. Su, Z. Zhang, H. Wang, F. Yang, Y. Luo, and J. Wang. Virtual network embedding through topology-aware node ranking. *ACM SIGCOMM Computer Communication Review*, 41(2):38–47, 2011. (Cited on page 135.)

[34] M. Chowdhury, M. R. Rahman, and R. Boutaba. Vineyard: Virtual network embedding algorithms with coordinated node and link mapping. *IEEE/ACM Transactions on Networking*, 20(1):206–219, 2012. (Cited on page 135.)

[35] N. M. K. Chowdhury and R. Boutaba. A survey of network virtualization. *Computer Networks*, 54(5):862–876, 2010. (Cited on pages 129, 135, and 136.)

[36] N. M. K. Chowdhury, M. R. Rahman, and R. Boutaba. Virtual network embedding with coordinated node and link mapping. *Proc. of the IEEE International Conference on Computer Communications (INFOCOM)*, pages 783–791, 2009. (Cited on pages 135 and 141.)

[37] Cisco Systems Inc. Forecast and methodology, 2014-2019 white paper. Technical report, Technical Report, Cisco Visual Networking Index, 2015. URL: `www.cisco.com/c/en/us/solutions/collateral/service-provider/ip-ngn-ip-next-generation-network/white_paper_c11-481360.html`. (Cited on pages 30 and 128.)

[38] D. Clark, R. Braden, A. Falk, and V. Pingali. Fara: Reorganizing the addressing architecture. *ACM SIGCOMM Computer Communication Review*, 33(4):313–321, 2003. (Cited on page 134.)

[39] S. Coniglio, A. M. C. A. Koster, and M. Tieves. Datasets used in this work. URL: `https://www.math2.rwth-aachen.de/en/forschung/projekte/data`. (Cited on pages 98, 126, 169, 195, and 199.)

[40] S. Coniglio, B. Grimm, A. M. C. A. Koster, M. Tieves, and A. Werner. Optimal offline virtual network embedding with rent-at-bulk aspects. *arXiv preprint arXiv:1501.07887*, 2015. (Cited on pages 128, 135, and 139.)

[41] S. Coniglio, A. M. C. A. Koster, and M. Tieves. Virtual network embedding under uncertainty: exact and heuristic approaches. *Proc. of the IEEE International Conference on the Design of Reliable Communication Networks (DRCN)*, pages 1–8, 2015. (Cited on pages 2, 16, 128, 135, and 139.)

[42] S. Coniglio, A. M. C. A. Koster, and M. Tieves. Data uncertainty in virtual network embedding: Robust optimization and protection levels. *Journal of Network and Systems Management*, 24(3):681–710, 2016. (Cited on pages 16, 128, 135, and 139.)

[43] D. Coudert, A. M. C. A. Koster, T. K. Phan, and M. Tieves. Robust redundancy elimination for energy-aware routing. *Proc. of the IEEE International Conference on Green Computing and Communications (GreenCom), on the Internet of Things and on Cyber, Physical and Social Computing (iThings/CPSCom)*, pages 179–186, 2013. (Cited on pages 2, 16, 30, 38, 41, and 96.)

[44] D. Coudert, A. Kodjo, and T. K. Phan. Robust energy-aware routing with redundancy elimination. *Computers & Operations Research*, 64:71–85, 2015. (Cited on page 38.)

[45] Cplex. IBM ILOG Cplex, version 12.4 and 12.6, 2015. URL: `www-01.ibm.com/software/commerce/optimization/cplex-optimizer/index.html`. (Cited on pages 98, 169, and 198.)

[46] J. D'Ambrosia. 100 gigabit ethernet and beyond [commentary]. *IEEE Communications Magazine*, 48(3):S6–S13, 2010. (Cited on page 37.)

[47] L. El Ghaoui and H. Lebret. Robust solutions to least-squares problems with uncertain data. *SIAM Journal on Matrix Analysis and Applications*, 18(4):1035–1064, 1997. (Cited on page 16.)

[48] L. El Ghaoui, F. Oustry, and H. Lebret. Robust solutions to uncertain semidefinite programs. *SIAM Journal on Optimization*, 9(1):33–52, 1998. (Cited on page 16.)

[49] D. Eppstein. Finding large clique minors is hard. *Journal of Graph Algorithms and Applications*, 13(2):197–204, 2009. (Cited on page 149.)

[50] I. Fajjari, N. A. Saadi, G. Pujolle, and H. Zimmermann. VNE-AC: Virtual network embedding algorithm based on ant colony metaheuristic. *Proc. of the IEEE International Conference on Communications (ICC)*, pages 1–6, 2011. (Cited on page 135.)

[51] N. Feamster, H. Balakrishnan, and J. Rexford. Some foundational problems in interdomain routing. *Proc. of the ACM SIGCOMM Workshop on Hot Topics in Networks (HotNets)*, pages 41–46, 2004. (Cited on page 134.)

[52] N. Feamster, L. Gao, and J. Rexford. How to lease the internet in your spare time. *ACM SIGCOMM Computer Communication Review*, 37(1):61–64, 2007. (Cited on page 134.)

[53] A. Fischer, J. F. Botero, M. T. Beck, H. De Meer, and X. Hesselbach. Virtual network embedding: A survey. *IEEE Communications Surveys & Tutorials*, 15 (4):1888–1906, 2013. (Cited on pages 129, 131, 133, 135, and 136.)

[54] M. Fischetti and M. Monaci. Cutting plane versus compact formulations for uncertain (integer) linear programs. *Mathematical Programming Computation*, 4(3): 239–273, 2012. (Cited on page 23.)

[55] W. Fisher, M. Suchara, and J. Rexford. Greening backbone networks: reducing energy consumption by shutting off cables in bundled links. *Proc. of the ACM SIGCOMM Workshop on Green Networking*, pages 29–34, 2010. (Cited on page 38.)

[56] O. E. Flippo, A. W. J. Kolen, A. M. C. A. Koster, and R. L. M. J. Van de Leensel. A dynamic programming algorithm for the local access telecommunication network expansion problem. *Proc. of the European Journal of Operational Research*, 127 (1):189–202, 2000. (Cited on page 79.)

[57] L. R. Ford and D. R. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8(3):399–404, 1956. (Cited on page 6.)

[58] C. Forster, I. Dickie, G. Maile, H. Smith, and M. Crisp. Understanding the environmental impact of communication systems. *Ofcom final report*, 2009. (Cited on page 37.)

[59] M. R. Garey, D. S. Johnson, and L. Stockmeyer. Some simplified np-complete graph problems. *Theoretical computer science*, 1(3):237–267, 1976. (Cited on pages 146, 151, and 156.)

[60] E. Gawrilow and M. Joswig. polymake: a framework for analyzing convex polytopes. In G. Kalai and G. M. Ziegler, editors, *Polytopes — Combinatorics and Computation*, pages 43–74. Birkhäuser, 2000. URL: `polymake.org`. (Cited on pages 70 and 71.)

[61] S. Geng et al. The complexity of the 0/1 multi-knapsack problem. *Journal of Computer Science and Technology*, 1(1):46–50, 1986. (Cited on pages 144 and 150.)

[62] F. Giroire, D. Mazauric, J. Moulierac, and B. Onfroy. Minimizing routing energy consumption: from theoretical to practical results. *Proc. of the IEEE International Conference on Green Computing and Communications (GreenCom) and on Cyber, Physical and Social Computing (CPSCom)*, pages 252–259, 2010. (Cited on page 38.)

[63] F. Giroire, J. Moulierac, T. K. Phan, and F. Roudaut. Minimization of network power consumption with redundancy elimination. *Computer communications*, 59: 98–105, 2015. (Cited on pages 38 and 42.)

[64] M. X. Goemans, A. V. Goldberg, S. A. Plotkin, D. B. Shmoys, E. Tardos, and D. P. Williamson. Improved approximation algorithms for network design problems. *Proc. of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 94:223–232, 1994. (Cited on page 125.)

[65] Greentouch. Greentouch green meter research study: Reducing the net energy consumption in communication networks by up to 90% by 2020 (white paper), 2013. URL: `s3-us-west-2.amazonaws.com/belllabs-microsite-greentouch/uploads/documents/GreenTouch_Green_Meter_Research_Study_26_June_2013.pdf`. (Cited on pages 31 and 129.)

[66] A. Gupta, A. Kumar, and T. Roughgarden. Simpler and better approximation algorithms for network design. *Proc. of the ACM Symposium on Theory of Computing (STOC)*, pages 365–372, 2003. (Cited on page 125.)

[67] M. Gupta and S. Singh. Greening of the internet. *Proc. of the ACM SIGCOMM Conference on Applications, technologies, architectures, and protocols for computer communications*, pages 19–26, 2003. (Cited on page 37.)

[68] M. F. Habib, M. Tornatore, and B. Mukherjee. Fault-tolerant virtual network mapping to provide content connectivity in optical networks. *Proc. of the National*

Fiber Optic Engineers Conference (NFOEC), page OTh3E.4, 2013. (Cited on page 135.)

[69] M. Handley. Why the internet only just works. *BT Technology Journal*, 24(3): 119–129, 2006. (Cited on page 134.)

[70] J. Hartley. Youtube, digital literacy and the growth of knowledge. 2008. (Cited on pages 31 and 129.)

[71] J. Hastad. Clique is hard to approximate within $n^{1-\epsilon}$. *Proc. of the IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 627–636, 1996. (Cited on page 148.)

[72] I. Heller and C. Tompkins. An extension of a theorem of dantzig's. *Linear inequalities and related systems*, 38:247–254, 1956. (Cited on page 26.)

[73] U. Hoelzle. Openflow@ google. *Open Networking Summit*, 17, 2012. (Cited on page 136.)

[74] I. Houidi, W. Louati, W. B. Ameur, and D. Zeghlache. Virtual network provisioning across multiple substrate networks. *Computer Networks*, 55(4):1011–1023, 2011. (Cited on pages 135 and 141.)

[75] I. Houidi, W. Louati, and D. Zeghlache. Exact multi-objective virtual network embedding in cloud environments. *The Computer Journal*, 58(3):403–415, 2015. (Cited on pages 135 and 141.)

[76] L. Hütten. Gültige Ungleichungen für Netzwerkdesign mit Komprimierung und festem Routing. Master's thesis, RWTH Aachen University, Germany, 2016. (Cited on page 125.)

[77] F. Idzikowski. Power consumption of network elements in ip over wdm networks. *TU Berlin, TKN Group, Tech. Rep. TKN-09-006*, 2009. (Cited on page 38.)

[78] J. Inführ and G. R. Raidl. Introducing the virtual network mapping problem with delay, routing and location constraints. *Network Optimization: Proc. of the International Network Optimization Conference (INOC)*, pages 105–117, 2011. (Cited on page 135.)

[79] A. Jarray and A. Karmouch. Decomposition approaches for virtual network embedding with one-shot node and link mapping. *IEEE/ACM Transactions on Networking*, 23(3):1012–1025, 2015. (Cited on page 135.)

[80] D. S. Johnson and K. Niemi. On knapsacks, partitions, and a new dynamic programming technique for trees. *Mathematics of Operations Research*, 8(1):1–14, 1983. (Cited on page 79.)

[81] D. S. Johnson, J. K. Lenstra, and A. Kan. The complexity of the network design problem. *Networks*, 8(4):279–285, 1978. (Cited on page 10.)

[82] R. M. Karp. Reducibility among combinatorial problems. *Complexity of computer computations*, pages 85–103, 1972. (Cited on pages 13, 75, and 77.)

[83] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan. The internet topology zoo. *IEEE Journal on Selected Areas in Communications*, 29(9):1765–1775, 2011. (Cited on pages 126, 168, 190, and 195.)

[84] A. M. C. A. Koster and M. Tieves. Network design with compression: complexity and algorithms. *Proc. of the INFORMS Computing Society Conference (ICS)*, pages 74–87, 2015. (Cited on pages 2, 30, and 38.)

[85] A. M. C. A. Koster, M. Kutschka, and C. Raack. Robust network design: Formulations, valid inequalities, and computations. *Networks*, 61(2):128–149, 2013. (Cited on pages 16, 98, 99, and 161.)

[86] A. M. C. A. Koster, T. K. Phan, and M. Tieves. Extended cutset inequalities for the network power consumption problem. *Electronic Notes in Discrete Mathematics: Proc. of the International Network Optimization Conference (INOC)*, 41: 69–76, 2013. (Cited on pages 2, 30, 38, 41, 42, and 72.)

[87] A. Kumar, R. Rastogi, A. Silberschatz, and B. Yener. Algorithms for provisioning virtual private networks in the hose model. *IEEE/ACM Transactions on Networking*, 10(4):565–578, 2002. (Cited on page 135.)

[88] W. Liu, Y. Xiang, S. Ma, and X. Tang. Completing virtual network embedding all in one mathematical programming. *Proc. of the IEEE International Conference on Electronics, Communications and Control (ICECC)*, pages 183–185, 2011. (Cited on page 135.)

[89] C. Lubritto, A. Petraglia, C. Vetromile, F. Caterina, A. D'Onofrio, M. Logorelli, G. Marsico, and S. Curcuruto. Telecommunication power systems: energy saving, renewable sources and environmental monitoring. *Proc. of the IEEE International Telecommunications Energy Conference (INTELEC)*, pages 1–4, 2008. (Cited on pages 31 and 36.)

[90] T. L. Magnanti and R. T. Wong. Network design and transportation planning: Models and algorithms. *Transportation science*, 18(1):1–55, 1984. (Cited on page 11.)

[91] T. L. Magnanti, P. Mirchandani, and R. Vachani. The convex hull of two core capacitated network design problems. *Mathematical Programming*, 60(1-3):233–250, 1993. (Cited on pages 46, 51, and 57.)

[92] T. L. Magnanti, P. Mirchandani, and R. Vachani. Modeling and solving the two-facility capacitated network loading problem. *Operations Research*, 43(1):142–157, 1995. (Cited on pages 11, 13, 46, 51, 55, 59, and 63.)

[93] O. Malik. Who is the world's biggest broadband company? find out, 2010. URL: `https://gigaom.com/2010/07/28/top-ten-broadband-providers/`. (Cited on page 134.)

[94] M. A. Marsan, L. Chiaraviglio, D. Ciullo, and M. Meo. Optimal energy savings in cellular access networks. *Proc. of the IEEE International Conference on Communications (ICC)*, pages 1–5, 2009. (Cited on page 37.)

[95] S. Mattia and M. Poss. Efficient approaches for the robust network loading problem. *to appear*. (Cited on page 125.)

[96] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. OpenFlow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74, 2008. (Cited on page 136.)

[97] M. Middendorf and F. Pfeiffer. On the complexity of the disjoint paths problem. *Combinatorica*, 13(1):97–107, 1993. (Cited on pages 144 and 145.)

[98] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization*. Wiley-Interscience, New York, NY, USA, 1988. (Cited on pages 14, 53, and 55.)

[99] OpenFlow. Open networking foundation. URL: `www.opennetworking.org`. (Cited on page 136.)

[100] S. Orlowski, R. Wessäly, M. Pióro, and A. Tomaszewski. Sndlib 1.0—survivable network design library. *Networks*, 55(3):276–286, 2010. URL: `sndlib.zib.de`. (Cited on pages 11, 31, 32, 98, 125, 168, and 195.)

[101] A. Pages, J. Perello, S. Spadaro, and G. Junyent. Strategies for virtual optical network allocation. *IEEE Communications Letters*, 16(2):268–271, 2012. (Cited on page 135.)

[102] T. K. Phan. *Design and management of networks with low power consumption*. PhD thesis, Université Nice Sophia Antipolis, 2014. (Cited on pages 35, 38, and 39.)

[103] M. Pickavet, W. Vereecken, S. Demeyer, P. Audenaert, B. Vermeulen, C. Develder, D. Colle, B. Dhoedt, and P. Demeester. Worldwide energy needs for ict: The rise of power-aware networking. *Proc. of the IEEE International Symposium on Advanced Networks and Telecommunication Systems (ANTS)*, pages 1–3, 2008. (Cited on page 37.)

[104] M. Pióro and D. Medhi. *Routing, flow, and capacity design in communication and computer networks*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004. (Cited on pages 11 and 38.)

[105] S. Poljak. A note on stable sets and coloring of graphs. *Commentationes Mathematicae Universitatis Carolinae*, 15:307–309, 1974. (Cited on page 146.)

[106] C. Raack, A. M. C. A. Koster, S. Orlowski, and R. Wessäly. On cut-based inequalities for capacitated network design polyhedra. *Networks*, 57(2):141–156, 2011. (Cited on pages 46, 51, 71, and 109.)

[107] S. Raghavan, B. Golden, and E. Wasil. *Telecommunications Modeling, Policy, and Technology (Operations Research/Computer Science Interfaces Series)*, volume 44. Springer Publishing Company, Incorporated, 2008. (Cited on page 109.)

[108] M. Resende and P. M. Pardalos. *Handbook of optimization in telecommunications*. Springer Science & Business Media, 2008. (Cited on page 38.)

[109] T. Richardson, Q. Stafford-Fraser, K. R. Wood, and A. Hopper. Virtual network computing. *IEEE Internet Computing*, 2(1):33–38, 1998. (Cited on page 134.)

[110] Riverbed, Vendor. Wan optimization controller. URL: `www.riverbed.com/products/wan-optimization`. (Cited on page 39.)

[111] E. Rosen and Y. Rekhter. Bgpmpls ip virtual private networks (vpns). 2006. (Cited on page 136.)

[112] J. Rosendahl. Benders decomposition for the virtual network embedding problem. Master's thesis, RWTH Aachen University, Germany, 2016. (Cited on page 195.)

[113] M. Rost, C. Fuerst, and S. Schmid. Beyond the stars: Revisiting virtual cluster embeddings. *ACM SIGCOMM Computer Communication Review*, 45(3):12–18, 2015. (Cited on pages 136, 154, 157, and 195.)

[114] A. Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer-Verlag Berlin Heidelberg, 2002. (Cited on page 5.)

[115] D. Schwerdel, D. Günther, R. Henjes, B. Reuther, and P. Müller. German-lab experimental facility. *Future Internet-FIS: Proc. of the Future Internet Symposium (FIS)*, pages 1–10, 2010. (Cited on page 136.)

[116] V. Sekar, Y. Xie, D. Maltz, M. Reiter, and H. Zhang. Toward a framework for internet forensic analysis. *Proc. of the ACM SIGCOMM Workshop on Hot Topics in Networks (HotNets)*, 2004. (Cited on page 134.)

[117] A. L. Soyster. Technical note—convex programming with set-inclusive constraints and applications to inexact linear programming. *Operations research*, 21(5):1154–1157, 1973. (Cited on page 16.)

[118] T. Trinh, H. Esaki, and C. Aswakul. Quality of service using careful overbooking for optimal virtual network resource allocation. *Proc. of the IEEE International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pages 296–299, 2011. (Cited on page 135.)

[119] J. S. Turner and D. E. Taylor. Diversifying the internet. *Proc. of the IEEE International Conference on Global Telecommunications Conference (GLOBECOM)*, 2: 6–13, 2005. (Cited on page 134.)

[120] S. J. Vaughan-Nichols. We love ipv6, we love ipv6 not. *Enterprise IT Planet*, 2004. (Cited on page 134.)

[121] W. Vereecken, L. Deboosere, D. Colle, B. Vermeulen, M. Pickavet, B. Dhoedt, and P. Demeester. Energy efficiency in telecommunication networks. *Proc. of the European Conference on Networks and Optical Communications (NOC)*, pages 44–51, 2008. (Cited on page 37.)

[122] VINO,Virtual Network Optimization. Optimierung virtueller und dynamischer kommunikationsnetze. URL: `http://vino.mathematik.uni-kassel.de`. BMBF grant 05M13PAA, joint project 05M2013 by the German Federal Ministry of Education and Research. (Cited on pages 2 and 197.)

[123] Y. Yemini and S. Da Silva. Towards programmable networks. *Proc. of the IFIP/IEEE International Workshop on Distributed Systems: Operations and Management (DSOM)*, pages 1–11, 1996. (Cited on page 134.)

[124] H. Yu, V. Anand, C. Qiao, and G. Sun. Cost efficient design of survivable virtual infrastructure to recover from facility node failures. *Proc. of the IEEE International Conference on Communications (ICC)*, pages 1–6, 2011. (Cited on page 135.)

[125] M. Yu, Y. Yi, J. Rexford, and M. Chiang. Rethinking virtual network embedding: substrate support for path splitting and migration. *ACM SIGCOMM Computer Communication Review*, 38(2):17–29, 2008. (Cited on pages 133, 135, and 136.)

[126] D. Yuan. Optimization models and methods for communication network design and routing. *Linkoping Studies in Science and Technology - Dissertations*, 2001. (Cited on page 38.)

[127] S. Zeadally, S. U. Khan, and N. Chilamkurti. Energy-efficient networking: past, present, and future. *Journal of Supercomputing*, 62(3):1093–1118, 2012. (Cited on page 37.)

[128] Z. Zhang, S. Su, J. Zhang, K. Shuang, and P. Xu. Energy aware virtual network embedding with dynamic demands: Online and offline. *Computer Networks*, 93: 448–459, 2015. (Cited on page 135.)

[129] Y. Zhu and M. H. Ammar. Algorithms for assigning substrate network resources to virtual network components. *Proc. of the IEEE International Conference on Computer Communications (INFOCOM)*, 12, 2006. (Cited on pages 135 and 141.)