DISS. ETH NO. 27975

Efficient Methods for Congruency-Constrained Optimization

A thesis submitted to attain the degree of

DOCTOR OF SCIENCES of ETH ZURICH

(Dr. sc. ETH Zurich)

presented by

MARTIN NÄGELE

MSc ETH Mathematics, ETH Zurich

born on March 23, 1993
citizen of Austria

accepted on the recommendation of

Prof. Dr. Rico Zenklusen, examiner
Dr. Neil Olver, co-examiner

2021

Acknowledgment

First and foremost, I am deeply grateful to my advisor Prof. Dr. Rico Zenklusen for his guidance and support over the course of my time as a PhD student in his group. Working with him was inspiring in so many ways, and I profited enormously from his profound knowledge and experience that he was always happy to share. Rico definitely has a great share in bringing me to the point where I am now. Thank you very much for all of that, Rico!

I am also very happy that I spent the last few years in a very enjoyable and welcoming atmosphere within our group at the Institute of Operations Research. Thanks go out to Adam, Alfonso, Andrea, Annette, Christian, Christoph, Christos, David, Federica, Georg, Haris, Ingo, Ivan, Jörg, Joe, Kevin, Laura, Leon, Miriam, Richard, Sabrina, Simon, Stephan, Stephen, Stefan, and Vera. I am delighted that some of those friendships grew far beyond the academic world and will definitely persist despite us probably taking very different paths in our future careers.

Finally, I am deeply indebted to my wife Pia. She was on my side celebrating successes, she put up with me doing night shifts when deadlines approached and things got tight, and she always knew when I needed support. Thank you so much for all your contributions—words can barely describe how grateful I am for all you did. I love you, Pia! ♡

Abstract

In this thesis, we study discrete combinatorial optimization problems with congruency constraints and present new techniques for dealing with such constraint types.

Strong motivation for studying congruency constraints comes from a long-standing open question in Integer Programming whether integer programs with constraint matrices with bounded subdeterminants are efficiently solvable. An important special case thereof are congruency-constrained integer programs $\min\{c^\top x \colon Tx \leq b, \gamma^\top x \equiv r \pmod m, x \in \mathbb{Z}^n\}$ with a totally unimodular constraint matrix T. Such problems have been shown to be polynomial-time solvable for $m = 2$, which led to an efficient algorithm for integer programs with bimodular constraint matrices, i.e., full-rank matrices whose $n \times n$ subdeterminants are bounded by two in absolute value. Whereas these advances heavily relied on existing results on well-known combinatorial minimum cut and circulation problems with parity constraints, new approaches are needed beyond the bimodular case, i.e., $m > 2$. We make first progress in this direction through several new techniques. In particular, we show how to efficiently decide feasibility of congruency-constrained integer programs with a totally unimodular constraint matrix for $m = 3$. Furthermore, for general m, our techniques also allow for identifying flat directions of infeasible problems, and deducing bounds on the proximity between solutions of the problem and its relaxation.

In the second part of this thesis, we study a generalization of the aforementioned parity-constrained minimum cut problems (and therefore also a generalization of other well-known variants of minimum cut problems such as global minimum cuts and minimum s-t cuts), namely congruency-constrained minimum cuts, where we consider cuts whose number of vertices is congruent to r modulo m, for some integers r and m. We develop a new contraction technique inspired by Karger's celebrated contraction algorithm for minimum cuts, which, together with further insights, leads to a polynomial time randomized approximation scheme for congruency-constrained minimum cuts for any constant modulus m. Instead of contracting edges of the original graph, we use splitting-off techniques to create an auxiliary graph on a smaller vertex set, which is used for performing random edge contractions. This way, a well-structured distribution of candidate pairs of vertices to be contracted is obtained, where the involved pairs are generally not connected by an edge. As a byproduct, our technique reveals new structural insights into near-minimum odd cuts, and, more generally, near-minimum congruency-constrained cuts.

Finally, we present results on finding a spanning tree subject to constraints on the edges in a family of cuts forming a laminar family of small width. Especially in the context of the Traveling Salesman Problem (TSP), new techniques for finding spanning trees with well-defined properties, and in particular, trees satisfying parity constraints on a chain of cuts, have been crucial in pushing a the analysis of a certain class of approximation algorithms. Our main contribution is a new dynamic programming approach where the value of a table entry does not only depend on the values of previous table entries, as it is usually the case, but also on a specific representative solution saved together with each table entry. This allows for handling a broad range of constraint types, including a relaxation of congruency constraints, or upper and lower bounds on the number of edges in each cut. Concretely, we present a quasi-polynomial time algorithm for the Minimum Chain-Constrained Spanning Tree Problem with an essentially best possible guarantee. Furthermore, we show how parity constraints as used in the context of (path) TSP and a generalization thereof can be handled, and discuss implications in this context.

Zusammenfassung

In dieser Arbeit betrachten wir diskrete kombinatorische Optimierungsprobleme mit einer Kongruenzbedingung und präsentieren neue Techniken, um solche Bedingungen zu behandeln.

Motivation für das Betrachten von Kongruenzbedingungen stammt von einer seit langem offenen Frage im Gebiet der Ganzzahloptimierung, nämlich ob ganzzahlige lineare Optimierungsprobleme, bei denen die Matrix des Ungleichungssystems beschränkte Subdeterminanten hat, effizient lösbar sind. Ein wichtiger Spezialfall dieser Problemklasse sind ganzzahlige lineare Optimierungsprobleme mit einer Kongruenzbedingung von der Form $\min\{c^\top x \colon Tx \leq b, \gamma^\top x \equiv r \pmod m, x \in \mathbb{Z}^n\}$ mit einer vollständig unimodularen Matrix T. Für solche Probleme wurde gezeigt, dass sie in Polynomialzeit lösbar sind falls $m = 2$ gilt. Dies führte zu effizienten Algorithmen für ganzzahlige lineare Optimierungsprobleme, bei denen die Matrix des Ungleichungssystems bimodular ist, d.h. dass die Matrix vollen Rang hat und alle quadratischen Untermatrizen der Größe $n \times n$ im Absolutbetrag höchstens 2 sind. Während diese Fortschritte auf bekannten Resultaten über minimale Schnitte und Zirkulationen mit Paritätsbedingungen beruhen, sind für Resultate, die über den bimodularen Fall hinausgehen, d.h. $m > 3$ betrachten, neue Herangehensweisen nötig. Wir erreichen erste Schritte in diese Richtung, indem wir mehrere neue Techniken einführen. Konkret zeigen wir, wie wir effizient bestimmen können, ob ein ganzzahliges lineares Optimierungsproblem mit einer vollständig unimodularen Matrix und einer Kongruenzbedingung mit Modul $m = 3$ eine Lösung hat oder nicht. Desweiteren erlauben unsere Techniken für allgemeine m flache Richtungen von Problemen die keine Lösung haben zu bestimmen, und wir können Schranken auf den Abstand von Lösungen eines ganzzahligen linearen Optimierungsproblems mit einer Kongruenzbedingung und einer Lösung der Relaxierung des Problems zeigen.

Im zweiten Teil dieser Arbeit behandeln wir eine Verallgemeinerung des zuvor erwähnten Problems, minimale Schnitte mit Paritätsbedingungen zu finden; insbesondere also auch eine Verallgemeinerung von anderen bekannten Varianten von Schnitt-Problemen wie globale minimale Schnitte und minimale s-t Schnitte: Wir studieren minimale Schnitte mit einer Kongruenzbedingung, das heißt Schnitte die eine Anzahl von Knoten enthalten, die kongruent zu r modulo m ist für bestimmte gegebene ganze Zahlen r und m. Wir entwickeln einen neuen Kontraktionsalgorithmus, der durch einen bekannten Kontraktionsalgorithmus von Karger zum finden von minimalen Schnitten inspiriert ist. Zusammen mit weiteren Erkenntnissen erhalten wir für jeden beliebigen Modul m ein Polynomialzeit-Approximationsschema zum finden von minimalen Schnitten, die eine Kongruenzbedingung mit Modul m erfüllen. Anstatt – wie im Algorithmus von Karger – Kanten des gegebenen Graphen zu kontrahieren, verwenden wir Splitting-Off-Techniken um einen Hilfsgraphen zu konstruieren, in dem wir sodann zufällige Kanten zur Kontraktion wählen. Auf diesem Weg erhalten wir eine wohlstrukturierte Vertilung über Knotenpaare, die wir im ursprünglichen Graph kontrahieren können, auch wenn sie dort nicht durch eine Kante verbunden sind. Als zusätzliche Resultate enthüllt unsere Technik neue strukturelle Einblicke in minimale Schnitte, die eine Kongruenzbedingung erfüllen, und insbesondere auch Einblicke in die Struktur von nur fast minimalen solchen Schnitten.

Zu guter Letzt präsentieren wir Resultate zu Spannbäumen, die Bedingungen an die Kanten in einer Familie von Schnitten, die eine laminare Familie von kleiner Breite bilden, erfüllen. Speziell im Kontext des Problems des Handlungsreisenden (TSP) haben Techniken zum Finden von Spannbäumen mit wohldefinierten Eigenschaften an großer Bedeutung gewonnen. Idealerweise sollten die Spannbäume Paritätsbedingungen an die Anzahl der Kanten in einer Kette von Schnitten erfüllen, um die Approximationsgarantien einer Klasse von Algorithmen zu verbessern. Unser Hauptbeitrag ist ein neuere Ansatz der dynamischen Programmierung, bei dem ein Tabelleneintrag nicht wie üblich nur von den Werten der vorhergehenden Tabelleneinträge abhängt, sondern auch von einer konkreten Lösung des Problems, die der Tabelleneintrag repräsentiert. Diese Änderung erlaubt es uns, eine breite Palette an Bedingungen behandeln zu können, insbesondere eine Relaxierung einer Kongruenzbedingung oder untere und obere Schranken auf die Anzahl von Kanten, die einen Schnitt kreuzen. Konkret präsentieren wir einen Algorithmus mit quasipolynomieller Laufzeit und de facto bestmöglicher Approximationsgarantie für das Problem der minimalen Schnitte mit Schranken auf die Anzahl von Kanten, die Schnitte kreuzen, welche eine Kette bilden. Außerdem zeigen wir, wie die Paritätsbedingungen, die im Zusammenhang mit TSP und seiner Pfad-Version eine Rolle spielen, behandelt werden können, und wir diskutieren Implikationen in diesem Zusammenhang.

This thesis includes material from the following publications:

1. Martin Nägele, Richard Santiago, and Rico Zenklusen. Congruency-constrained TU problems beyond the bimodular case. *To appear in SODA 2022*. arXiv:2109.03148[math.OC] (2021).
2. Martin Nägele and Rico Zenklusen. A new contraction technique with applications to congruency-constrained cuts. *Proceedings of the 2019 Conference on Integer Programming and Combinatorial Optimization (IPCO): 327–340 (2019)*. https://doi.org/gxkc
3. Martin Nägele and Rico Zenklusen. A new contraction technique with applications to congruency-constrained cuts. *Mathematical Programming 183(1): 455–481 (2020)*. https://doi.org/gxj6
4. Martin Nägele and Rico Zenklusen. A new dynamic programming approach for spanning trees with chain constraints and beyond. *Proceedings of the 2019 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA): 1550–1569, (2019)*. https://doi.org/gxj7

Results presented in this thesis have been part of projects supported by the Swiss National Science Foundation grant 200021_184622 and the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 817750).

Contents

Chapter 1

Introduction

The problems studied in this thesis lie in the research area of Discrete Optimization. In this area, problems can generally be described as finding one element with certain properties among a finite number of options—typically one that maximizes or minimizes some objective function—or deciding whether an element with certain properties exists at all. In most interesting problems, the number of potential solutions is huge and they are not given explicitly, so that generating and checking all of them is not a desirable option. Metaphorically, this endeavour is often referred to as "finding the needle in the haystack", thereby picturing how hopeless such an approach would be in practice. Classical problems that fall into the category of discrete optimization problems comprise all kinds of network problems like finding shortest paths from one vertex to another in a graph, vehicle routing problems in which we have to design routes of cars to deliver certain goods, scheduling problems to efficiently use resources or allocate people to tasks, clustering problems of grouping similar objects, or integer linear programming problems that ask for integral solutions of linear inequality systems, just to name a few. Already from this very limited set of examples, one can clearly see that many of these problems are closely linked to or used for modelling real-world problems, and applications cover a vast variety of tasks.

To judge the quality of algorithms designed to solve such problems, we measure their running time as a function of the input size of an instance of the problem. We say that an algorithm is *efficient* its running time can be bounded by a polynomial in the input size. Many classical problems admit deterministic efficient algorithms, for example shortest path problems, matching problems, or flow problems. Still, there are also large classes of problems for which efficient algorithms are not known to date. Somewhat surprisingly, even though it is widely believed to be true, it is not known whether there exist problems that are intrinsically hard and do not admit such efficient algorithms. In the context of decision problems (i.e., questions that can be answered "yes" or "no"), this question is known as the P vs. NP problem: The complexity class NP contains all decision problems for which a solution to a "yes"-instance can be verified in polynomial time, i.e., a proof that the answer is "yes" can efficiently be checked. All problems mentioned earlier (or, to be precise, their decision versions) fall into this category, but from an algorithms perspective, being in NP doesn't buy much: The question of how a solution to a "yes"-instance can be generated is still left open. Problems where this step can be performed deterministically and efficiently, i.e., in polynomial time, belong to the subclass P of NP. Using this terminology, the big open question is whether $\mathrm{P} = \mathrm{NP}$, i.e., whether all problems in NP have efficient algorithms, or whether there exist problems that are in NP but not in P. While the common belief is that $\mathrm{P} \neq \mathrm{NP}$, no attempts to prove either of the two outcomes of the question were successful so far. This P vs. NP problem even appears on the list of Millennium Prize Problems published by the Clay Mathematics Institute in May 2000, and a reward of one million US dollars is since waiting as an award for a correct answer [Cla].

Interestingly, the problem class NP contains a subclass of problems, the so-called NP-complete problems, which can be seen as "the most difficult" problems in NP in the following sense: If there was a polynomial-

time algorithm for one of those problems, then there would be one for all problems in NP, i.e., we would obtain $\mathrm{P} = \mathrm{NP}$. Consequently, based on the common belief that $\mathrm{P} \neq \mathrm{NP}$, we do not expect polynomial-time algorithms for NP-complete problems. Still, this is not the end of the study of such problems. In case of optimization problems, one way out is to look for efficient *approximation algorithms*, i.e., efficient algorithms might not always find optimal solutions, but provide solutions that are guaranteed to be close to optimal, for example with objective values off by at most a constant factor. Another option is to try to identify problem parameters such that efficient algorithms exist once these parameters are fixed. Similarly, there might also be constraints in the problems such that, once we allow some controlled violation of these constraints, corresponding solutions can be found efficiently.

In this thesis, we study three problem families that can each be cast into one of the previously mentioned categories: Congruency-constrained integer programming (Chapter 2), congruency-constrained minimum cuts (Chapter 3), and chain-constrained spanning trees (Chapter 4). While the problems and their backgrounds are introduced independently and in detail in the corresponding chapters, we want to highlight some connections here, in particular the appearance of *congruency constraints* in all of the studied problems. Let us start by stating a general integer programming problem of the form

$$\begin{aligned} \min \quad & c^\top x & & \\ & Ax & \leq \quad & b \\ & x & \in \quad & \mathbb{Z}^n \end{aligned} \tag{1.1}$$

for a matrix $A \in \mathbb{Z}^{m \times n}$, a right-hand side vector $b \in \mathbb{Z}^m$, and an objective vector $c \in \mathbb{Z}^n$. Integer programming (or, more precisely, its decision version, where the question is whether a solution $x \in \mathbb{Z}^n$ with at most some given objective value exists) is one of the aforementioned NP-hard problems. Consequently, as mentioned earlier, we do not expect efficient algorithms for solving such problems. The crux of such integer programming problems really lies in the integrality constraint $x \in \mathbb{Z}^n$: Optimizing over the continuous domain $x \in \mathbb{R}^n$, so-called linear programming, can be done efficiently.

For integer programming, interesting problem subclasses that may or may not be polynomial-time solvable can be obtained by constraining the matrix A. One well-studied subclass is obtained by requiring that A is *totally unimodular*, i.e., requiring that all determinants of square submatrices are in $\{-1, 0, 1\}$. Totally unimodular matrices are very structured, and it is well-known that so are the feasible regions of integer programs with totally unimodular constraint matrices, which can be exploited to reduce solving such problems to solving linear programs. Generalizing unimodularity, for $\Delta \in \mathbb{Z}_{>1}$, we consider so-called Δ-modular matrices A: Matrices of full column rank n, and with all determinants of $n \times n$ submatrices bounded by Δ in absolute value. Nurtured by the insights on (totally) unimodular constraint matrices mentioned above, and a positive result for $\Delta = 2$ (integer programs with *bimodular* constraint matrices) by Artmann, Weismantel, and Zenklusen [AWZ17], it is conjectured that integer programming with Δ-modular constraint matrices admits polynomial time algorithms if Δ is bounded by a constant. In Chapter 2, we approach this conjecture by studying an important subclass of such bounded subdeterminant integer programs, namely *congruency-constrained TU problems*: We look at integer programming problems of the form given in (1.1), assume that A is totally unimodular, and add a *congruency-constraint* of the form

$$\gamma^\top x \equiv r \pmod{m}$$

for some $\gamma \in \mathbb{Z}^n$, $m \in \mathbb{Z}_{>0}$, and $r \in \mathbb{Z}$. By rewriting such a congruency constraint in the form $\gamma^\top x + my = r$ with an auxiliary integer variable $y \in \mathbb{Z}$, one can readily see that congruency-constrained TU problems are a special case of m-modular problems. Studying congruency constraints in this context is motivated by the aforementioned result of Artmann, Weismantel, and Zenklusen [AWZ17], where parity-constrained problems (i.e., congruency-constrained problems with modulus $m = 2$) play a central role. We present an approach that, among others, allows to decide feasibility of congruency-constrained TU problems with $m = 3$ in strongly polynomial time, thereby going beyond the previously studied parity-constraints and the corresponding bimodular constraint matrices.

Chapter 3 studies a further special case of congruency-constrained TU problems that comes in a graph theoretical setting: Congurency-constrained minimum cut problems. Here, given an undirected graph $G = (V, E)$, we want to find a minimum non-empty cut $C \subsetneq V$ among all cuts C that satisfy the congruency-constraint $|C| \equiv r \pmod{m}$ on the cardinality of C, and such that $|\delta^+(C)|$ is minimized. This problem is still NP-hard for general m, so we consider the setup where m is a constant. Here, polynomial-time algorithms are known in the case where m is a prime power [NSZ19], but the complexity status is open for general constant m. We present a quasi-polynomial randomized approximation scheme for general constant m, i.e., an algorithm parametrized by $\varepsilon > 0$ that returns with high probability a feasible solution whose value is within a factor of $(1 + \varepsilon)$ of the actual optimal value, and runs in quasi-polynomial time for any constant ε. Despite being neither a polynomial-time nor an exact algorithm, this result strengthens our belief that congruency-constrained minimum cut problems should be solvable in deterministic polynomial time, and motivates further research in that direction. Moreover, from a techniques point of view, it shows that the classical and very simple randomized approach to minimum cut problems by Karger can be enhanced to also work in a setting with congruency constraints.

Finally, Chapter 4 presents algorithms for constrained spanning tree problems, where we impose constraints on a family of cuts that form a chain or a laminar family of small width. Our approach can handle cut constraints of different types: For each cut in the family, we can for example impose upper and/or lower bounds on the number of edges crossing the cut. Other variants include requiring either a large number of edges, say at least τ many, or less than τ edges such that the number of edges satisfies a parity constraint. Spanning trees satisfying such a type of parity constraints are of interest in the analysis of the classical Christofides-Serdyukov algorithm for the famous Travelling Salesman Problem [Chr76] applied to its path version. There, we are given n sites, metric distances, and two distinguished and different sites s and t among the n given ones, and the goal is to find a shortest path from s to t that contains all n sites. The classical approach starts with some short spanning tree, and then adds extra edges to guarantee that there is a suitable path in the union of the spanning tree edges and the extra edges. Parity constraints naturally come up in this setting because a (multi)set of edges contains a spanning s-t path if and only if the degrees are even at all vertices except for s and t, and odd at s and t. Loosely speaking, the total length of the necessary extra edges depends on the choice of the initial spanning tree, and it turns out that strong bounds can be achieved if the tree has an odd number of edges in certain cuts that form a chain. With the above-mentioned type of constraints (i.e., having an odd number of edges or at least τ many), we achieve a polynomial-time $(1.5 + 1/\tau)$-approximation algorithm for Path TSP, thereby almost matching the classical Christofides-Serdyukov analysis that gives a 1.5-approximation for TSP, and the one by Zenklusen that gives the same factor for Path TSP [Zen19]. Since very recently and through much more involved techniques, it is known that for some $\delta > 10^{-36}$, this problem can even be approximated within a factor of $1.5 - \delta$ [KKG21].

Until not too long ago, congruency constraints typically appeared in the area of Combinatorial Optimization only in the form of parity constraints, with examples including the minimum odd cut problem, packing odd cycles in graphs, or submodular function minimization over odd or even cardinality sets. Only the study of Δ-modular integer linear programs and the connection to congruency-constrained TU optimization triggered going to congruency constraints with more general modulus m. This includes submodular minimization with congruency constraints [NSZ19] and much of the work presented in this thesis. For that reason, there are no well-established techniques for dealing with congruency constraints yet. One contribution of the presented work is to showcase approaches that help with such constraint types. Concretely, the results in this thesis present several cases where guessing well-chosen parts or properties of a solution leads to simpler problems, and in some cases allows for finding solutions to congruency-constrained problems through solving problems without congruency constraints. Another paradigm that appears repeatedly is the so-called divide-and-conquer approach: Carefully splitting a problem into two or more independent and smaller subproblems of the same or a very similar type, solving these problems recursively, and finally combining solutions of the subproblems to obtain a solution of the initial problem. Last but not least, we also exploit the power of linear relaxations of the problems that we study. Opposed to formulations with integer variables, compact linear formulations can

be solved efficiently, and even in cases where we obtain fractional solutions, we may be able to either round the fractional solution to an integral one, or deduce properties of the instance that we may want to exploit.

In the interest of easier access to the results presented in this thesis, all three of the subsequent chapters can be read independently, with problem settings and their backgrounds being introduced without assuming knowledge from other chapters. Moreover, we again remark that the content of Chapter 2 is also available as a preprint [NSZ21]; Chapter 3 appeared in [NZ20], with a short version presented in [NZ19a]; and Chapter 4 is an extended version of [NZ19b].

Chapter 2

Congruency-constrained TU problems beyond the bimodular case

2.1 Introduction

Integer linear programs (ILPs) $\min\{c^\top x \colon Ax \leq b, x \in \mathbb{Z}^n\}$ for $A \in \mathbb{Z}^{k\times n}$, $b \in \mathbb{Z}^k$, and $c \in \mathbb{Z}^n$ are one of the most basic yet powerful discrete optimization problems. They are well-known to be NP-hard, and extensive research is dedicated to identify efficiently solvable subclasses. One of the best known such classes is when the constraint matrix A is required to be *totally unimodular* (TU), i.e., all square submatrices of A have a determinant in $\{-1, 0, 1\}$. The class of totally unimodular ILPs still comprises a large number of interesting and heavily studied problems, as for example network flow and cut problems, bipartite matching problems, and many others.

Intriguingly, it is still badly understood what kind of generalizations of this classical result on ILPs with totally unimodular constraint matrices are possible to obtain larger classes of efficiently solvable ILPs. In particular, there is a long-standing open question on whether ILPs are efficiently solvable if their constraint matrix is Δ-modular for constant Δ. Here, a matrix $A \in \mathbb{Z}^{k\times n}$ is Δ-modular for $\Delta \in \mathbb{Z}_{>0}$ if it has full column rank n, and all $n \times n$ submatrices have determinants bounded by Δ in absolute value.[1] Besides TU constraint matrices, progress has only been achieved for the bimodular case $\Delta = 2$, for which an efficient algorithm was presented by Artmann, Weismantel, and Zenklusen [AWZ17]. A relevant special case of such problems are Congruency-Constrained TU Problems.[2]

Congruency-Constrained TU Optimization (CCTU): Let $T \in \{-1, 0, 1\}^{k\times n}$ be TU, $b \in \mathbb{Z}^k$, $c \in \mathbb{Z}^n$, $m \in \mathbb{Z}_{>0}$, $\gamma \in \mathbb{Z}^n$, and $r \in \mathbb{Z}$. The task is to show infeasibility, unboundedness, or find a minimizer of

$$\min\left\{c^\top x \;\middle|\; Tx \leq b,\ \gamma^\top x \equiv r \pmod m,\ x \in \mathbb{Z}^n\right\} \quad .$$

Even for $m = 2$, CCTU problems capture classical combinatorial optimization problems like the min odd cut problem. Moreover, there are reasons to believe that insights on CCTU problems may be key to make further progress on the open question of bounded subdeterminant ILPs. For $\Delta = 2$, a result of Veselov and Chirkov [VC09] implies that bimodular ILPs reduce to CCTU problems with $m = 2$, i.e., with parity constraints (see [AWZ17]). The result in [VC09] does not extend to $\Delta > 2$, and it remains open whether another reduction to CCTU problems may exist. Questions closely related to CCTU have also appeared in

[1] One may also consider *totally* Δ*-modular* matrices A, where *all* square subdeterminants of A are bounded by Δ in absolute value. The notion of Δ-modularity is more general in the sense that totally Δ-modular ILPs can be reduced to Δ-modular ILPs.

[2] A CCTU problem with modulus m can be written as an m-modular ILP by transforming the congruency constraint into a linear equality constraint as follows. First append the row $\gamma^\top$ to the matrix T and then append a column with zeros everywhere except for the last entry (the one corresponding to the newly added row), which is set to m. Finally, the right-hand side of the newly added constraint is set to r, the target residue.

recent progress of Fiorini, Joret, Weltge, and Yuditsky [FJWY21], who obtained an efficient algorithm for totally Δ-modular ILPs with a constraint matrix having at most two non-zeros in each row. This algorithm needs to compute certain circulations with parity constraints, which can be interpreted as CCTU problems with a bounded number of additional constraints.

Moreover, we highlight that for prime numbers m, CCTU problems with modulus m are equivalent to ILPs with a constraint matrix A that has full column rank and all of whose $n \times n$ subdeterminants are within $\{0, \pm m\}$, in the sense that any of the two problems can be efficiently transformed to the other one.[3]

Typically, we consider a modulus m in CCTU problems that is constant, since CCTU with arbitrary non-constant modulus m is NP-hard (one can, for example, model the minimum bisection problem).

2.1.1 Our results

We present the first progress towards solving CCTU problems beyond the parity-constrained case by approaching the corresponding feasibility problem.

Congruency-Constrained TU Feasibility (CCTUF): Let $T \in \{-1, 0, 1\}^{k \times n}$ be a totally unimodular matrix, let $b \in \mathbb{Z}^k$, $m \in \mathbb{Z}_{>0}$, $\gamma \in \mathbb{Z}^n$, and $r \in \mathbb{Z}$. The task is to show infeasibility or find a solution of the system

$$Tx \leq b,\ \gamma^\top x \equiv r \pmod m,\ x \in \mathbb{Z}^n \ .$$

Our main result is the following.

Theorem 2.1. *There is a strongly polynomial-time randomized algorithm for CCTUF problems with $m = 3$.*

As we show in Section 2.A, being able to solve feasibility problems is also enough to detect unboundedness of CCTU problems.[4] One of the key ideas in the proof of Theorem 2.1 is to reduce a CCTUF problem to a hierarchy of slightly relaxed congruency-constrained problems with totally unimodular constraint matrices that we call R-CCTUF problems, and which we define as follows.

***R*-Congruency-Constrained TU Feasibility (*R*-CCTUF):** Let $T \in \{-1, 0, 1\}^{k \times n}$ be a totally unimodular matrix and let $b \in \mathbb{Z}^k$. Additionally, let $m \in \mathbb{Z}_{>0}$, $\gamma \in \mathbb{Z}^n$, and $R \subseteq \{0, \dots, m-1\}$. The task is to show infeasibility or find a feasible solution of the system

$$Tx \leq b,\ \gamma^\top x \in R \pmod m,\ x \in \mathbb{Z}^n \ .$$

Here, the constraint $\gamma^\top x \in R \pmod m$ is satisfied if and only if there exists an $r \in R$ such that $\gamma^\top x \equiv r \pmod m$. We call R the set of *target residues*. Clearly, every CCTUF problem is an R-CCTUF problem with $R = \{r\}$. Intuitively, the larger the set R of target residues is, the easier the corresponding problem gets—in the extreme case of $|R| = m$, the congruency constraint is trivially fulfilled by any solution, and simply finding a solution of the TU problem without congruency constraint is enough. Additionally, R-CCTUF problems can always be reduced to several problems of the same type with a smaller set of target residues. In particular, any R-CCTUF problem can be reduced to $|R|$ many CCTUF problems, namely one for each $r \in R$. Our new progress for R-CCTUF problems is going two steps into the hierarchy if the modulus m is a prime number, i.e., we can solve feasibility problems with $|R| \geq m - 2$.

Theorem 2.2. *There is a strongly polynomial-time randomized algorithm for R-CCTUF problems with constant prime modulus m and $|R| \geq m - 2$.*

[3] The reduction mentioned in Footnote 2 from a CCTU problem to a Δ-modular ILP shows one direction. The other one follows by an analogous reduction to the one used in the bimodular case [AWZ17].

[4] Analogous to linear and integer programming, we call a CCTU *unbounded* if it is possible to achieve arbitrarily good objective values. Hence, having an unbounded feasible region does not imply unboundedness of the problem.

Observing that for $m = 3$, an R-CCTUF problem with $|R| = m - 2$ is in fact a CCTUF problem, Theorem 2.1 immediately follows from Theorem 2.2. Our proof of Theorem 2.2 is inspired by methods developed in [AWZ17] for bimodular integer programs, but goes significantly beyond the strategy and techniques employed there. In particular, we also decompose R-CCTUF problems into smaller ones following Seymour's decomposition of TU matrices, but we need methods that allow for progressing in the hierarchy of R-CCTUF problems introduced above. This step requires us to have prime modulus due to an application of the Cauchy-Davenport Inequality. The decomposition approach deterministically reduces general R-CCTUF problems to problems with so-called *base-block* constraint matrices. While parity-constraints are fairly common in Combinatorial Optimization and known techniques could be leveraged in [AWZ17] to solve parity-constrained base block problems, we present new approaches for $m > 2$. In particular, we create new links to recent advances on congruency-constrained submodular optimization and exact weight flow problems. The only known algorithm for exact weight flow problems is randomized, which is why we obtain a randomized algorithm as stated in Theorem 2.2.[5]

Interestingly, focusing on the case of $|R| = m - 1$ only, our techniques lead to a substantially simpler approach for R-CCTUF problems that does not need to rely on decomposition methods and can therefore avoid randomization and the prime modulus requirement, resulting in the following theorem.

Theorem 2.3. *There is a strongly polynomial-time algorithm to solve R-CCTUF problems with $|R| = m - 1$.*

For $m = 2$, Theorem 2.3 states that feasibility of parity-constrained TU problems can be decided efficiently. This is a special case of bimodular IP feasibility, which has been known to admit polynomial time algorithms since the work of Veselov and Chirkov [VC09]. Let us also remark that for general m, the congruency constraint in R-CCTUF problems with $|R| = m-1$ can be rewritten in the form $\gamma^\top x \not\equiv r \pmod m$ for some residue r. Such constraint types and generalizations thereof have been studied in different settings already, in particular in the context of minimizing submodular functions (see Goemans and Ramakrishnan [GR95], and Grötschel, Lovász, and Schrijver [GLS93]).

Our approach for Theorem 2.3 is derived from interesting structural properties of R-CCTUF problems that are likely to be of independent interest, and two of which we want to highlight here. One is concerned with *flat directions* of the underlying polyhedron, i.e., vectors $d \in \mathbb{Z}^n \setminus \{0\}$ for which the *width* $\max\{d^\top x \colon x \in \mathbb{Z}^n, Tx \leq b\} - \min\{d^\top x \colon x \in \mathbb{Z}^n, Tx \leq b\}$ is small. Prior to our work, results of this type have only been known for very restricted cases. In particular, it is proved in Artmann's PhD thesis [Art20, Theorem 3.4] that for CCTUF problems restricted to modulus $m = 3$ and to base block constraint matrices, it holds that if the problem is infeasible, then a row of the constraint matrix is a flat direction of width 1. Our techniques show, through an arguably much simpler approach, that analogous results hold for arbitrary moduli m and CCTUF problems without any further restriction on the constraint matrix. Moreover, our result also generalizes to R-CCTUF problems, providing the following bound on the width, which can easily be seen to be tight.

Theorem 2.4. *For every R-CCTUF problem, either there is a constraint matrix row that is a flat direction of the underlying polyhedron of width at most $m - |R| - 1$, or a feasible solution of the R-CCTUF problem can be found in strongly polynomial time.*

Finally, our techniques also lead to proximity results. We call the problem obtained from CCTU, CCTUF, or R-CCTUF problems after dropping the congruency constraint the *relaxation* of the respective problem. Note that this relaxation is not a linear relaxation in the usual sense as we still require integral solutions, but is nonetheless closely related to it due to the totally unimodular constraint matrices. Prior knowledge of proximity results in this context have been very limited. In particular, it was known [Art20, Lemma 3.3] that given a feasible CCTU problem with $m = 3$, then for any vertex $y \in \mathbb{Z}^n$ of the underlying polyhedron $\{x \in \mathbb{R}^n \colon Tx \leq b\}$, there exists a feasible solution x of the CCTU problem such that $\|y - x\|_\infty \leq 2$. While the method used in [Art20] is specific for the $m = 3$ case, our techniques lead to the following more general

[5] In this context, we consider a randomized algorithm to be one that always correctly detects infeasibility of a problem, and finds a solution of a feasible problem with high probability $1 - 1/n$, where n is the number of variables.

result for arbitrary modulus m and, again, the more general congruency-constraint type. Here, R-CCTU denotes the optimization versions of R-CCTUF problems, analogous to the relation between CCTU and CCTUF problems. In other words, an R-CCTU problem is a CCTU problem where the congruency-constraint $\gamma^\top x \equiv r \pmod m$ for a single residue r is replaced by $\gamma^\top x \in R \pmod m$ for a set R of residues.

Theorem 2.5. *Consider a feasible R-CCTU problem with modulus m.*

(i) For any x_0 feasible for the relaxation, there is an x feasible for the problem with $\|x - x_0\|_\infty \leq m - |R|$.
(ii) For any x_0 optimal for the relaxation, there is an x optimal for the problem with $\|x - x_0\|_\infty \leq m - |R|$.

Moreover, in (i) and (ii), given x_0 and any feasible or optimal solution of the R-CCTU problem, respectively, a solution x with the stated properties can be found in strongly polynomial time.

2.1.2 Related work

The maximum absolute value Δ of a subdeterminant of the constraint matrix is a parameter that has received significant attention in integer programming recently. The closely related problem class of congruency-constrained combinatorial optimization problems has been investigated already in the early 80's for the parity-constrained case, and several further advances have been made since. We briefly recap prior work linked to these areas.

A problem that can be cast as a bounded subdeterminant integer program, has gained substantial interest recently [BFMR14; CFHJW20; CFHW21], and was resolved in [FJWY21], is the stable set problem in graphs G with bounded odd cycle packing number $\mathrm{ocp}(G)$, i.e., graphs for which the maximum number of disjoint odd cycles is bounded. The incidence matrix of such a graph has maximum subdeterminant $2^{\mathrm{ocp}(G)}$ (see, e.g., [GKS95]). Several further interesting results link the parameter Δ to properties of integer programs, their relaxations, and underlying polyhedra (see, e.g., [BDEHN14; EV17; LPSX20; LPSX21; PSW21; Tar86] and references therein). Furthermore, there has been interesting recent progress on the problem of approximating the largest subdeterminant of a matrix (see Di Summa, Eisenbrand, Faenza, and Moldenhauer [DEFM15], and Nikolov [Nik15]). Also, IPs with more constrained subdeterminant structures that admit efficient algorithms for integer programming were considered [VC09; AEGOVW16; GSW21].

One of the most classical congruency-constrained combinatorial optimization problems is the minimum odd cut problem, which asks to find a minimum cut among all cuts with an odd number of vertices. Padberg and Rao [PR82] presented a first efficient method for the minimum odd cut problem. Subsequently, Barahona and Conforti [BC87] showed that efficient minimization is also possible over all cuts with an even number of vertices. Later works by Grötschel, Lovász, and Schrijver [GLS84], and by Goemans and Ramakrishnan [GR95] generalized these results to the minimization of submodular functions. More precisely, the approach of [GR95] allows for minimizing over so-called triple families, which includes the case of cuts $C \subseteq V$ of cardinality *not* congruent to r modulo m, for any integers r and m. Nägele, Sudakov, and Zenklusen [NSZ19] showed that a submodular function can also be efficiently minimized over sets of cardinality $r \pmod m$, for any integer m that is a constant prime power. For the special case of minimum cuts, Nägele and Zenklusen [NZ20] presented a randomized PTAS for finding a minimum cut among all cuts containing $r \pmod m$ many vertices, for any constant m. The latter result is also presented in Chapter 3 of this thesis.

2.1.3 Organization of the chapter

In Section 2.2, we present the key ideas and techniques that lead to our new results. In particular, Section 2.2.1 presents a decomposition lemma, a crucial ingredient that is central to all our results, and we showcase its strength by readily deducing from it our flatness and proximity results (Theorems 2.4 and 2.5). Subsequently, Section 2.2.2 gives an overview of our approach to CCTUF problems and the proof of Theorem 2.2.

A proof of the decomposition lemma as well as more applications thereof (in particular, Theorem 2.3), are given in Section 2.3, while Sections 2.4 and 2.5 fill in details and present the missing proofs from Section 2.2.2.

2.2 Overview of our approach

2.2.1 Decomposition, flat directions, and proximity

One technique that we employ repeatedly is a careful decomposition of vectors into well-structured ones. In particular, we often apply such decomposition to solutions of CCTUF or R-CCTUF problems, to obtain a structured sum of other vectors. A key role in this decomposition is taken by *elementary vectors*, which we define as follows.

Definition 2.6. *Let $T \in \mathbb{Z}^{k\times n}$ be a totally unimodular matrix.*

(i) A vector $d \in \mathbb{Z}^n$ is TU-appendable *to T if the matrix $\left(\begin{smallmatrix} T \\ d^\top \end{smallmatrix}\right)$ is totally unimodular.*

(ii) A vector $x \in \mathbb{Z}^n$ is elementary *w.r.t. T if $d^\top x \in \{-1, 0, 1\}$ for all d that are TU-appendable to T.*

Concretely, we obtain the following decomposition lemma.

Lemma 2.7 (Decomposition lemma)**.** *Let $T \in \{-1, 0, 1\}^{k\times n}$ be a totally unimodular matrix, let $b \in \mathbb{Z}^k$, and let $x_0, y \in \mathbb{Z}^n$ be two solutions of the system $Tx \leq b$. Then, we can determine in strongly polynomial time $y^1, \ldots, y^n \in \mathbb{Z}^n$ and $\lambda_1, \ldots, \lambda_n \in \mathbb{Z}_{\geq 0}$ such that $y - x_0 = \sum_{i=1}^n \lambda_i y^i$ with the following properties:*

(i) $y^1, \ldots, y^n$ are elementary with respect to T.

(ii) For $\mu_1, \ldots, \mu_n \in \mathbb{Z}_{\geq 0}$ with $\mu_i \leq \lambda_i$ for all $i \in [n]$, the vector $\tilde{y} \coloneqq x_0 + \sum_{i=1}^n \mu_i y^i$ satisfies $T\tilde{y} \leq b$.

In words, the above decomposition lemma allows for efficiently writing a solution y to the relaxation of a CCTUF (or, more generally, also R-CCTUF) as a sum of another solution x_0 and a combination of elementary vectors y^i that can moreover be freely combined to obtain other solutions to the relaxation. A formal proof of this decomposition lemma is given in Section 2.3.3.

One of our applications of the decomposition lemma is to bound the search space in which we need to look for solutions of R-CCTUF problems. Note that given a solution x_0 of the relaxation of an R-CCTUF problem and any feasible R-CCTUF solution y, i.e., one that also satisfies the congruency constraint, as well as a TU-appendable row $d^\top$, Lemma 2.7 allows for efficiently decomposing $y - x_0$ into a sum of the form $\sum_{i=1}^n \lambda_i y^i$ with $\sum_{i=1}^n \lambda_i \geq |d^\top(y - x_0)|$. Hence, if $|d^\top(y - x_0)|$ is large, the sum $\sum_{i=1}^n \lambda_i y^i$ has many terms, and due to point (ii), there are many options to build new solutions $x_0 + \sum_{i=1}^n \mu_i y^i$ of the relaxation of the R-CCTUF problem by removing an arbitrary subset of the terms (i.e., choosing $\mu_i \in \{0, \ldots, \lambda_i\}$). Thus, in order to obtain a new feasible solution for the R-CCTUF problem, we have to make sure that $\gamma^\top x_0 + \sum_{i=1}^n \mu_i \gamma^\top y^i \in R \pmod m$, i.e., that we hit a feasible residue again. The following lemma shows that there always exists such a choice with $\sum_{i=1}^n \mu_i \leq m - |R|$.

Lemma 2.8. *Let $m \in \mathbb{Z}_{>0}$, $R \subseteq \{0, \ldots, m-1\}$, and $r_1, \ldots, r_\ell \in \mathbb{Z}$ with $\sum_{i\in[\ell]} r_i \in R \pmod m$. If there is no interval $I = \{i_1, \ldots, i_2\}$ with $i_1, i_2 \in [\ell]$ and $i_1 < i_2$ such that $\sum_{i\in[\ell]\setminus I} r_i \in R$, then $\ell \leq m - |R|$.*

Proof. Assume for the sake of deriving a contradiction that there is no interval $I \subseteq \{1, \ldots, \ell\}$ such that $\sum_{i\in[\ell]\setminus I} r_i \in R$, but $\ell \geq m - |R| + 1$. Consider the ℓ integers $s_0 = 0$, $s_1 = r_1, \ldots, s_{\ell-1} = r_1 + \ldots + r_{\ell-1}$. Observe that $s_j \notin R \pmod m$ for all $j \in [\ell - 1]$; for otherwise, there is an interval $I = \{j + 1, \ldots, \ell\}$ for some $j \in [\ell - 1]$ such that $\sum_{i\in[\ell]\setminus I} r_i = s_j \in R \pmod m$, contradicting the assumption. Thus, $s_j \in \{0, \ldots, m - 1\} \setminus R \pmod m$ for $j \in [\ell - 1]$. Hence, because $\ell \geq m - |R| + 1$, we have by the pigeonhole principle that there exist distinct $j_1, j_2 \in [\ell - 1]$ such that $s_{j_1} \equiv s_{j_2} \pmod m$. Thus, $I = \{j_1 + 1, \ldots, j_2\}$ is an interval with $\sum_{i\in[\ell]\setminus I} r_i = \sum_{i\in[\ell]} r_i - (s_{j_2} - s_{j_1}) \equiv \sum_{i\in[\ell]} r_i \in R \pmod m$, again contradicting the assumption and hence completing the proof. □

Indeed, Lemma 2.8 shows that as long as the sum

$$\underbrace{\gamma^\top y^1 + \ldots + \gamma^\top y^1}_{\lambda_1 \text{ many terms}} + \ldots + \underbrace{\gamma^\top y^n + \ldots + \gamma^\top y^n}_{\lambda_n \text{ many terms}} \in R - \gamma^\top x_0 \pmod m$$

has at least $m-|R|+1$ many terms, there is a subset of consecutive terms that can be removed while keeping the total residue inside the set $R-\gamma^\top x_0$. Iterating the procedure eventually leaves us with terms corresponding to a solution of the form $\tilde{y} \coloneqq x_0 + \sum_{i=1}^{n} \mu_i y^i$ with $\sum_{i=1}^{n} \mu_i \leq m-|R|$. Observe that this solution $\tilde{y}$ is close to the solution x_0 of the relaxation of the initial problem in the sense that $|d^\top(\tilde{y}-x_0)| \leq m-|R|$, which can be used as a bound for the search space when looking for feasible solutions. Beyond that, the idea described above is also at the heart of our flatness and proximity results (Theorems 2.4 and 2.5).

One caveat in the above construction is that a direct realization of the approach suggested by Lemma 2.8 may have a worst-case running time polynomial in m, which is not polynomial in the input size of the R-CCTUF problem when m is part of the input. Interestingly, given a sum $\sum r_i$ that lies in $R \pmod m$ for residues $r_i \in \mathbb{Z}$ and a set $R \subseteq \{0,\dots,m-1\}$, it is generally NP-hard to find a smallest possible number of terms r_i that also sum to a residue in R modulo m, as can be seen by a reduction from the Subset Sum problem, for example. Nonetheless, we are able to get the following constructive result by exploiting that the sum $\sum_{i=1}^{n} \lambda_i y^i$ contains no more than n distinct vectors y^i, and the fact that we do not need to find a shortest partial sum with residue in $R-\gamma^\top x_0$ but only one with at most $m-|R|$ terms. Its formal proof is postponed to Section 2.3.3.

Lemma 2.9. *Consider an R-CCTUF problem with modulus m, constraint matrix T, a feasible solution y, and a solution x_0 of its relaxation. We can obtain in strongly polynomial time a feasible solution $\tilde{y}$ such that*

(i) for any $d \in \mathbb{Z}^n$ that is TU-appendable to T, we have $d^\top(\tilde{y}-x_0) \leq m-|R|$, and
(ii) for any $c \in \mathbb{Z}^n$ such that x_0 minimizes $c^\top x$ over the relaxation of the R-CCTUF problem, $c^\top \tilde{y} \leq c^\top y$.

Note that point (ii) adds an additional property on the relation of the costs of the three vectors x_0, y, and $\tilde{y}$ that is useful in optimization settings. To showcase two concrete applications of Lemma 2.9 in this overview, we show how Lemma 2.9 readily implies our flatness and proximity results, i.e., Theorems 2.4 and 2.5. We start by showing Theorem 2.4, which is a consequence of the following statement.

Lemma 2.10. *Consider an R-CCTUF problem, and let $d^\top x \leq \beta$ be one of its constraints. Either*

(i) d is a flat direction of width at most $m-|R|-1$ for the underlying polyhedron, or
(ii) the problem is feasible if and only if the R-CCTUF problem without the constraint $d^\top x \leq \beta$ is.

In case (ii), a solution of the initial problem can be obtained in strongly polynomial time from any solution of the initial problem without the constraint $d^\top x \leq \beta$.

Proof. Assume that d is a direction of width at least $m-|R|$, and let x_0 be feasible for the relaxation of the R-CCTUF problem such that $d^\top x_0 \leq \beta - m + |R|$. It is enough to show that we can in strongly polynomial time obtain a feasible solution of the initial problem, assuming that we are given a feasible solution y of the problem without the constraint $d^\top x \leq \beta$. Applying Lemma 2.9 in this setting, we get that given y, we can in strongly polynomial time obtain another feasible solution $\tilde{y}$ such that $d^\top \tilde{y} \leq d^\top x_0 + m - |R| \leq \beta$, i.e., a solution that also satisfies the constraint $d^\top x \leq \beta$. This proves the desired statement. □

Proof of Theorem 2.4. Consider an R-CCTUF problem and one of its constraints $d^\top x \leq \beta$. Using a result of Tardos [Tar86], we can in strongly polynomial time determine whether this constraint identifies a direction of width at most $m-|R|-1$ of the underlying polyhedron (namely, by optimizing the objectives $d^\top x$ and $-d^\top x$ over the polyhedron). If not, by Lemma 2.10, the constraint can be dropped without changing the feasibility status. Iterating over all constraints, we either find a flat direction, or we end up with a problem without inequality constraints that is trivially feasible, thus implying that the initial problem was feasible as well. In that case, a solution of the initial problem can be constructed within the desired running time from a solution of the final problem through Lemma 2.10. □

Let us remark that the width $m-|R|-1$ of flat directions in infeasible problems is best possible for any size of R, as can be seen from the infeasible problems given by $\{x \in \mathbb{Z} \colon 0 \leq x \leq m-\ell-1, x \in R_\ell \pmod m\}$ with $R_\ell = \{m-\ell,\dots,m-1\}$ for $\ell \in \{1,\dots,m-1\}$.

Finally, we also show how Lemma 2.9 implies Theorem 2.5. More precisely, we prove the following generalization, from which Theorem 2.5 follows immediately.

Theorem 2.11. *Consider a feasible R-CCTU problem with modulus m and constraint matrix T.*

(i) For any feasible solution x_0 of the relaxation, there is a feasible solution x of the R-CCTU problem such that for every vector d that is TU-appendable to T, we have $d^\top(x - x_0) \leq m - |R|$.

(ii) For any optimal solution x_0 of the relaxation, there is an optimal solution x of the R-CCTU problem such that for every vector d that is TU-appendable to T, we have $d^\top(x - x_0) \leq m - |R|$.

Moreover, in (i) and (ii), given x_0 and any feasible or optimal solution of the R-CCTU problem, respectively, a solution x with the stated properties can be found in strongly polynomial time.

Proof. For part (i), apply Lemma 2.9 to the given problem with feasible solutions y and x_0 of the problem and its relaxation, respectively, to obtain a feasible solution $\tilde{y}$. Property (i) in Lemma 2.9 states that $d^\top(\tilde{y} - x_0) \leq m - |R|$ for any $d \in \mathbb{Z}^n$ that is TU-appendable to the constraint matrix. Moreover, if y is given, we can also obtain $\tilde{y}$ in strongly polynomial time by Lemma 2.9, hence $\tilde{y}$ has the properties of the solution x claimed by Theorem 2.11.

To also deduce part (ii), we proceed identically, but take x_0 to be an optimal solution of the relaxation with respect to the minimization objective $c^\top x$, and y an optimal solution to the problem. In that case, on top of what we derived before, $\tilde{y}$ satisfies $c^\top \tilde{y} \leq c^\top y$ by property (ii) in Lemma 2.9. Thus, because y is optimal, this must be an equality and $\tilde{y}$ is optimal, as well. □

Proof of Theorem 2.5. Note that for every $i \in [n]$, the unit vector e_i and its negative $-e_i$ are TU-appendable to every totally unimodular matrix. Thus, the solutions guaranteed by Theorem 2.11 satisfy

$$\|x - x_0\|_\infty = \max_{i \in [n]} \max\{e_i^\top(x - x_0), -e_i^\top(x - x_0)\} \leq m - |R| \ . \qquad \square$$

We postpone further applications of the decomposition lemma to Section 2.3, and continue with an overview of our approach to deal with R-CCTUF problems. The above discussion aimed at exemplifying how the decomposition lemma can be employed, and should help to better understand further implications, including settings that we state in the following overview of how to deal with R-CCTUF problems.

2.2.2 Overview of our approach to R-CCTUF problems and Theorem 2.2

When approaching R-CCTUF problems of the form

$$Tx \leq b, \ \ \gamma^\top x \in R \pmod m, \ \ x \in \mathbb{Z}^n$$

with constant prime modulus m, we follow the general idea of decomposing the problem into smaller ones by applying Seymour's TU decomposition to the constraint matrix T. Exploiting Seymour's decomposition to approach problems that involve TU matrices is a standard approach that has been successfully used in a variety of contexts (see for example [DK14; AWZ17; AF21]). In particular, this includes the solution to parity-constrained TU problems presented in [AWZ17]. However, going to congruency-constraints with modulus 3 or larger creates substantial extra hurdles beyond prior techniques. For completeness and clear references, we repeat Seymour's TU decomposition framework here, which breaks a TU matrix into smaller ones using so-called 1-, 2-, and 3-sums, and pivoting operations, which are defined as follows.

Definition 2.12 (1-, 2-, and 3-sums)**.** *Let $A \in \mathbb{Z}^{k_A \times n_A}$, $B \in \mathbb{Z}^{k_B \times n_B}$, $e \in \mathbb{Z}^{k_A}$, $f \in \mathbb{Z}^{n_B}$, $g \in \mathbb{Z}^{k_B}$, $h \in \mathbb{Z}^{n_A}$.*

(i) The 1-sum *of A and B is $A \oplus_1 B := \left(\begin{smallmatrix} A & 0 \\ 0 & B \end{smallmatrix}\right)$.*

(ii) The 2-sum *of $\begin{pmatrix} A & e \end{pmatrix}$ and $\begin{pmatrix} f^\top \\ B \end{pmatrix}$ is $\begin{pmatrix} A & e \end{pmatrix} \oplus_2 \begin{pmatrix} f^\top \\ B \end{pmatrix} := \begin{pmatrix} A & ef^\top \\ 0 & B \end{pmatrix}$.*

(iii) The 3-sum *of $\left(\begin{smallmatrix} A & e & e \\ h^\top & 0 & 1 \end{smallmatrix}\right)$ and $\left(\begin{smallmatrix} 0 & 1 & f^\top \\ g & g & B \end{smallmatrix}\right)$ is $\left(\begin{smallmatrix} A & e & e \\ h^\top & 0 & 1 \end{smallmatrix}\right) \oplus_3 \left(\begin{smallmatrix} 0 & 1 & f^\top \\ g & g & B \end{smallmatrix}\right) := \left(\begin{smallmatrix} A & ef^\top \\ gh^\top & B \end{smallmatrix}\right)$.*

Definition 2.13 (Pivoting). *Let $C \in \mathbb{Z}^{k\times n}$, $p \in \mathbb{Z}^n$, $q \in \mathbb{Z}^k$, and $\varepsilon \in \{-1, 1\}$. The matrix obtained from pivoting on ε in $T \coloneqq \begin{pmatrix} \varepsilon & p^\top \\ q & C \end{pmatrix}$, i.e., pivoting on the element T_{11} of T, is $\operatorname{pivot}_{11}(T) \coloneqq \begin{pmatrix} -\varepsilon & \varepsilon p^\top \\ \varepsilon q & C - \varepsilon q p^\top \end{pmatrix}$. More generally, $\operatorname{pivot}_{ij}(T)$ for indices i and j such that $T_{ij} \in \{-1, 1\}$ is obtained from T by first permuting rows and columns such that the element T_{ij} is permuted to the first row and first column, then performing the above pivoting operation on the permuted matrix, and finally reversing the row and column permutations.*

It is well-known that a 1-, 2-, and 3-sum is totally unimodular if and only if the two summands it is obtained from are, and a pivoted matrix is totally unimodular if and only if the original matrix is. Seymour's TU decomposition theorem states that a TU matrix is either very structured, or it can be decomposed using 1-, 2-, and 3-sums, or pivoting steps. We use the following variation of the decomposition theorem, which provides some extra guarantees on the dimensions of the matrices appearing in the decomposition. It readily follows from classical statements of Seymour's decomposition for TU matrices (see Section 2.5.1 for details).

Theorem 2.14 (Seymour's TU decomposition). *Let $T \in \mathbb{Z}^{k\times n}$ be a totally unimodular matrix. Then, one of the following cases holds.*

(i) *T or $T^\top$ is a network matrix.*

(ii) *T is, possibly after iteratively applying the operations of*
- *deleting a row or column with at most one non-zero entry,*
- *deleting a row or column that appears twice or whose negation also appears in the matrix, and*
- *changing the sign of a row or column,*

equal to one of

$$\begin{pmatrix} 1 & -1 & 0 & 0 & -1 \\ -1 & 1 & -1 & 0 & 0 \\ 0 & -1 & 1 & -1 & 0 \\ 0 & 0 & -1 & 1 & -1 \\ -1 & 0 & 0 & -1 & 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

(iii) *T can, possibly after row and column permutations, be decomposed into a 1-, 2-, or 3-sum of totally unimodular matrices with $n_A, n_B \geq 2$.*

(iv) *T can, after pivoting once and possibly performing row and column permutations, be decomposed into a 3-sum of totally unimodular matrices with $n_A, n_B \geq 2$.*

Additionally, we can in time $\operatorname{poly}(n)$ decide which of the cases holds and determine the involved matrices.

Cases (i) and (ii) are the cases where T is a so-called *base block* matrix. We exploit the structure of those matrices to reduce CCTUF problems with such a constraint matrix T to certain combinatorial optimization problems with congruency constraints. In particular, if T is a network matrix, the corresponding problem can be interpreted as a congruency-constrained circulation problem. Here, we exploit a connection to exact weight matching problems [CGM92] that results in an efficient randomized procedure. For T being the transpose of a network matrix, we present a reduction to a congruency-constrained submodular minimization problem, which can be solved (whenever m is a prime power) by a recent algorithm by Nägele, Sudakov, and Zenklusen [NSZ19]. We expand on these connections in Section 2.4, thereby obtaining the following statement on the corresponding feasibility problems.

Theorem 2.15. *Let T be a TU matrix for which case (i) or (ii) in Theorem 2.14 holds. There is a strongly polynomial time randomized algorithm to solve CCTUF problems with constraint matrix T and constant prime power modulus.*

In the cases where the constraint matrix T admits a decomposition as a 1-, 2-, or 3-sum, i.e., case (iii) of Theorem 2.14, we can write $T = \begin{pmatrix} A & ef^\top \\ gh^\top & B \end{pmatrix}$. If T is a 2-sum, g and h will be zero vectors; if T is a 1-sum, also e and f will be zero vectors. This matrix decomposition splits the variables x, the right-hand sides b, and the residue vector γ into two parts accordingly. The R-CCTUF problem can then be rewritten as the

problem of finding a feasible solution of the system

$$\begin{aligned} \begin{pmatrix} A & ef^\top \\ gh^\top & B \end{pmatrix} \cdot \begin{pmatrix} x_A \\ x_B \end{pmatrix} &\leq \begin{pmatrix} b_A \\ b_B \end{pmatrix} \\ \gamma_A^\top x_A + \gamma_B^\top x_B &\in R \pmod m \\ x_A \in \mathbb{Z}^{n_A},\ x_B &\in \mathbb{Z}^{n_B} \ . \end{aligned} \tag{2.1}$$

For any fixed values of $\alpha \coloneqq f^\top x_B$ and $\beta \coloneqq h^\top x_A$, the above problem can be split into the two almost independent CCTUF problems

$$\begin{aligned} Ax_A &\leq b_A - \alpha e \\ h^\top x_A &= \beta \\ \gamma_A^\top x_A &\equiv r_A \pmod m \\ x_A &\in \mathbb{Z}^{n_A} \end{aligned} \quad \text{and} \quad \begin{aligned} Bx_B &\leq b_B - \beta g \\ f^\top x_B &= \alpha \\ \gamma_B^\top x_B &\equiv r_B \pmod m \\ x_B &\in \mathbb{Z}^{n_B} \end{aligned} \ , \tag{2.2}$$

where we would like to find solutions x_A and x_B for residues r_A and r_B such that $r_A + r_B \in R \pmod m$. Hence, this desired relation between the target residues r_A and r_B is the only dependence between the two problems once α and β are fixed. We refer to the problem on the left as the *A-problem* and the problem on the right as the *B-problem*.

A solution of the initial R-CCTUF problem can only exist for pairs $(\alpha, \beta) \in \mathbb{Z}^2$ for which both the A- and the B-problem are feasible. We denote this set by $\Pi \subseteq \mathbb{Z}^2$. In Section 2.5, we will see that Π is a polyhedron that can be obtained by essentially projecting feasible solutions of the relaxation of our R-CCTUF problem down to the (α, β)-space. This will allow us to deduce structural properties of Π. For now, we aim at narrowing down the values of $(\alpha, \beta) \in \Pi$ that we have to consider to find a feasible solution. To this end, we use the following Lemma.

Lemma 2.16. *Consider an R-CCTUF problem of the form given in (2.1). We can in strongly polynomial time obtain $\ell_i, u_i \in \mathbb{Z}$ with $u_i - \ell_i \leq m - |R|$ for $i \in \{0, 1, 2\}$ such that if the R-CCTUF problem has a solution, then it has one with $\ell_0 \leq \alpha + \beta \leq u_0$, $\ell_1 \leq \alpha \leq u_1$, and $\ell_2 \leq \beta \leq u_2$, where $\alpha = f^\top x_B$ and $\beta = h^\top x_A$.*

Note that α, β, and $\alpha + \beta$ are scalar products of a solution of (2.1) with suitably chosen row vectors. We show in Section 2.3.2 that those rows are all TU-appendable to the constraint matrix, thus enabling the application of techniques from the previous section to prove existence of solutions with those scalar products bounded to the desired range. Here, as a consequence of Lemma 2.16, we can restrict our attention to $O(m^2)$ many pairs (α, β) in the *narrowed* set

$$\Pi_{\text{narrowed}} \coloneqq \Pi \cap \left\{(\alpha, \beta) \in \mathbb{Z}^2 \colon \ell_0 \leq \alpha + \beta \leq u_0, \ell_1 \leq \alpha \leq u_1, \ell_2 \leq \beta \leq u_2\right\} \ .$$

We will later see that properties of Π imply that we can choose ℓ_i, u_i such that we even have

$$\Pi_{\text{narrowed}} = \left\{(\alpha, \beta) \in \mathbb{Z}^2 \colon \ell_0 \leq \alpha + \beta \leq u_0, \ell_1 \leq \alpha \leq u_1, \ell_2 \leq \beta \leq u_2\right\} \ .$$

One natural attempt at this point would be to explicitly try all $O(m^4)$ remaining combinations of r_A, r_B, and $(\alpha, \beta) \in \Pi_{\text{narrowed}}$, and recurse on the corresponding (now independent) A- and B-problems in (2.2). If we could guarantee that both problems had about the same number of variables in each such step (more precisely, at least a constant fraction of the original variables), this would lead to a polynomial time procedure at least for constant moduli m: The number of variables would go down by roughly a factor of two in every step; hence we would fall back to cases (i) or (ii) of Theorem 2.14 after $O(\log n)$ many iterations at the latest, each increasing the number of subproblems by a factor of $O(m^4)$, giving a total running time bound of

$m^{O(\log n)}$.[6] Unfortunately, the guarantees of Theorem 2.14 are much weaker: We can only guarantee that both A and B have at least two columns, and if their sizes happen to be imbalanced in most decomposition steps, the above argument fails.

Still, we can always solve the relaxations of both problems for all $(\alpha, \beta) \in \Pi_{\text{narrowed}}$. Without loss of generality, let us assume that the B-problem is the smaller among the A- and the B-problem (with respect to the number of columns in its constraint matrix, i.e., the number of variables). Because B has at most half the number of columns compared to T, it turns out that we can afford (in terms of running time) to recursively call Theorem 2.2 on an R-CCTUF version of the B-problem (i.e., the B-problem in (2.2) with the congruency constraint replaced by $\gamma_B^\top x_B \in R_B \pmod m$, for sets R_B of the same size as the set R in the original problem). Concretely, for any fixed $(\alpha, \beta) \in \Pi_{\text{narrowed}}$, at most $m - |R| + 1$ such recursive calls suffice to determine up to $m - |R| + 1$ different feasible residues of the B-problem (or fewer, if there are less than that many). We elaborate on why this is enough in what follows.

Let $\pi \colon \Pi_{\text{narrowed}} \to 2^{\{0,\dots,m-1\}}$ be the function assigning to any $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ the set $\pi(\alpha, \beta) \subseteq \{0, \dots, m-1\}$ of residues $r_B \in \{0, 1, \dots, m-1\}$ for which the B-problem is feasible. We call π a *narrowed pattern* associated to the problem given in (2.1). Note that this pattern depends on the 3-sum decomposition and the choice of ℓ_i and u_i in Lemma 2.16, and hence may not be unique. Also, we remark that a narrowed pattern can be seen as a restriction (to the narrowed domain Π_{narrowed}) of a *global* pattern that maps any $(\alpha, \beta) \in \Pi$ to the corresponding set of feasible residues of the B-problem.

We can easily obtain a feasible solution for (2.1) if, among the solutions of the B-problem that we compute, we find a solution x_B that fulfills $\gamma_A^\top x_A + \gamma_B^\top x_B \in R \pmod m$, where x_A is the computed solution to the relaxation of the A-problem. Indeed, in this case, the concatenation of the two solutions x_A and x_B is feasible for the relaxation of (2.1). In particular, if $|\pi(\alpha, \beta)| \geq m - |R| + 1$ for some $(\alpha, \beta) \in \Pi_{\text{narrowed}}$, we are guaranteed that there is such a feasible combination. As explained above, through recursive calls to our procedure on the B-problem, we can decide whether we are in this case, and if so also compute $m - |R| + 1$ different feasible residues (and corresponding solutions). Concretely, if we start from a problem with $|R| = m - 2$, whenever we find a pair (α, β) of scalar products in Π_{narrowed} with $|\pi(\alpha, \beta)| \geq 3$, we can find a feasible solution. If $|\pi(\alpha, \beta)| \leq 2$ for all $(\alpha, \beta) \in \Pi_{\text{narrowed}}$, we study the pattern π more closely.

One interesting special case is when $|\pi(\alpha, \beta)| = 1$ for all $(\alpha, \beta) \in \Pi_{\text{narrowed}}$, i.e., each of the B-problems is feasible for precisely one residue r_B. It turns out that in this case, π is *linear* in the following sense.

Definition 2.17. *Let $\Pi \subseteq \mathbb{Z}^2$, and let $\pi \colon \Pi \to 2^{\{0,\dots,m-1\}}$ for some $m \in \mathbb{Z}_{>0}$. We say that π is* linear *if $|\pi(\alpha, \beta)| = 1$ for all $(\alpha, \beta) \in \Pi$, and there exist $r_0, r_1, r_2 \in \mathbb{Z}$ such that the mapping $r \colon \Pi \to \mathbb{Z}$ fulfilling $\pi(\alpha, \beta) = \{r(\alpha, \beta)\}$ satisfies $r(\alpha, \beta) \equiv r_0 + r_1\alpha + r_2\beta \pmod m$ for all $(\alpha, \beta) \in \Pi$.*

Linearity of π and the shape of the domain Π_{narrowed} makes it possible to encode the feasibility structure of the B-problem in only two variables y_1 and y_2 that represent the scalar products α and β, which allows for replacing x_B with those new variables.

Theorem 2.18. *Consider an R-CCTUF problem of the form given in (2.1) and let π an associated narrowed pattern. If π is linear, then (2.1) can be reduced to the R-CCTUF problem*

$$\begin{array}{rcrcrcrcl}
 & & Ax_A & + & ey_1 & & & \leq & b_A \\
 & & h^\top x_A & & & - & y_2 & = & 0 \\
\ell_0 & \leq & & & y_1 & + & y_2 & \leq & u_0 \\
\ell_1 & \leq & & & y_1 & & & \leq & u_1 \\
\ell_2 & \leq & & & & & y_2 & \leq & u_2 \\
 & & \gamma_A^\top x_A & + & r_1 y_1 & + & r_2 y_2 & \in & r_0 + R \pmod m \\
 & & x_A & & & & & \in & \mathbb{Z}^{n_A} \\
 & & & & y_1 & , & y_2 & \in & \mathbb{Z}
\end{array} \tag{2.3}$$

[6] More generally, this enumerative approach is efficient whenever the depth of Seymour's decomposition is at most logarithmic in the input size.

for suitable $\ell_0, u_0, \ell_1, u_1, \ell_2, u_2 \in \mathbb{Z}$ *with* $u_i - \ell_i \leq m - |R|$ *and* $r_0, r_1, r_2 \in \{0, 1, \ldots, m-1\}$ *that can be determined in strongly polynomial time. That is, the initial* R*-CCTUF problem is feasible if and only if* (2.3) *is, and a solution of one problem can be transformed into one for the other in strongly polynomial time.*

Hence, when π is linear, we aim at applying Theorem 2.18 and continuing our procedure with the R-CCTUF problem (2.3). To make progress, we aim at obtaining a smaller problem, which, as before, we measure in terms of the number of variables. Note that the number of variables of (2.3) is the number of columns of A plus 2, which is the same as the number of columns of the original problem plus 2 minus the number of columns of B. However, recall that by Theorem 2.14, we are only guaranteed that the matrix B has at least two columns—which, in the extreme case, is not enough to reduce the number of columns through Theorem 2.18. Nevertheless, the equality constraint in (2.3) allows for eliminating a variable while keeping the TU structure of the constraint matrix, thus guaranteeing that we can make progress. The following theorem formalizes this result.

Theorem 2.19. *Let* $\begin{pmatrix} A & a_1 \\ a_2^\top & \alpha \end{pmatrix}$ *be a TU matrix with* $\alpha \neq 0$*. Then, the matrix* $A - \alpha a_1 a_2^\top$ *is TU, and the two systems*

$$\begin{cases} Ax + a_1 y \leq b \\ a_2^\top x + \alpha y = \beta \end{cases} \quad \text{and} \quad \begin{cases} \left(A - \alpha a_1 a_2^\top\right) x \leq b - \alpha\beta a_1 \\ y = \alpha\beta - \alpha a_2^\top x \end{cases}$$

are equivalent.

Combining Theorems 2.18 and 2.19, we can thus make progress in case of a linear narrowed pattern π. For non-linear narrowed patterns, like the one exemplified in Fig. 2.1, there are pairs (α, β) for which there is more than one residue available, i.e., $|\pi(\alpha, \beta)|$, which is an additional flexibility we can exploit as follows.

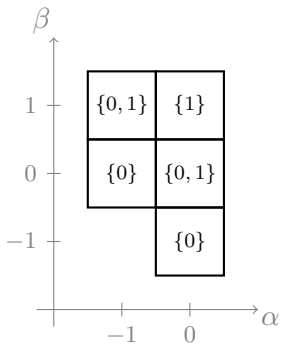


Figure 2.1: A non-linear pattern π with support defined by $-1 \leq \alpha \leq 0$, $-1 \leq \beta \leq 1$, and $-1 \leq \alpha + \beta \leq 1$.

Concretely, consider a pair $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ of scalar products with $\pi(\alpha, \beta) = \{r_B^1, \ldots, r_B^\ell\}$ for some $\ell > 1$ and pairwise different $r_B^i \in \{0, \ldots, m-1\}$, and let $x_B^1, \ldots, x_B^\ell$ be solutions of the relaxation of the B-problem with residues $\gamma_B^\top x_B^i = r_B^i$. Observe that we can combine any feasible solution x_A of the corresponding A-problem with any of the solutions x_B^i to obtain feasible solutions (x_A, x_B^i) of the relaxation of the initial R-CCTUF problem. Thus, there is a solution with scalar products (α, β) if and only if the following variation of the A-problem is feasible, where $R' \coloneqq R - \pi(\alpha, \beta) = \{(r - r_B \bmod m) \colon r \in R, r_B \in \pi(\alpha, \beta)\}$:

$$\begin{aligned} Ax_A &\leq b_A - \alpha e \\ h^\top x_A &= \beta \\ \gamma_A^\top x_A &\in R' \pmod m \\ x_A &\in \mathbb{Z}^{n_A} \ . \end{aligned} \tag{2.4}$$

We will create a subproblem of the above form for each pair $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha, \beta)| \geq 2$, and recurse on these problems. Doing so for all such scalar product pairs $(\alpha, \beta) \in \Pi_{\text{narrowed}}$, we create $O(m^2)$ many R-CCTUF problems to recurse on, each having at most $n - 2$ many variables. A key observation that allows for bounding the number of times we construct a problem of type (2.4) and recurse on it is that problem (2.4) is simpler than the problem we started with, because the set of target residues R' strictly increased in size compared to R, whenever m is a prime number. This is a consequence of the Cauchy-Davenport Inequality stated below.

Lemma 2.20 (Cauchy-Davenport Inequality)**.** *Let* m *be a prime number and* $R_1, R_2 \subseteq \{0, \ldots, m-1\}$*. Then*

$$|\{(r_1 + r_2 \bmod m) \colon r_1 \in R_1, r_2 \in R_2\}| \geq \min\{m, |R_1| + |R_2| - 1\} \ .$$

Consequently, after at most $m - |R|$ reduction steps, the target residues comprise all possible residues and the corresponding problem gets trivial. Therefore, the total number of subproblems that are spawned can be

bounded by $O(m^{2(m-|R|)})$. It thus remains to discuss scalar products $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha, \beta)| = 1$ that are not covered by the previous arguments. Fig. 2.1 shows an example of a narrowed pattern that contains three scalar product pairs (α, β) with $|\pi(\alpha, \beta)| = 1$ together with two pairs with $|\pi(\alpha, \beta)| = 2$. Again, explicitly solving the corresponding A-problems is not an option because we lack the necessary progress either in terms of the number of variables or the number of target residues.

Also, it is not possible to apply Theorem 2.18 only to the pairs $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha, \beta)| = 1$, because Theorem 2.18 crucially relies on the shape of the full domain of π, which can be described by inequalities of the form $\ell_0 \leq \alpha + \beta \leq u_0$, $\ell_1 \leq \alpha \leq u_1$, and $\ell_2 \leq \beta \leq u_2$. Therefore, we focus in this case on identifying a well-chosen linear *sub-pattern* $\widetilde{\pi}$ of π, i.e., a mapping $\widetilde{\pi}$ with the properties that (i) its domain is a subset of the domain of π and can be described by inequalities of the above type, (ii) $\widetilde{\pi}(\alpha, \beta) = \{r_{\alpha,\beta}\}$ for some $r_{\alpha,\beta} \in \pi(\alpha, \beta)$, and (iii) $\tilde{\pi}$ is linear according to Definition 2.17. Loosely speaking, a sub-pattern covers some of the available residues in the B-problem, and it is structured enough so that we can apply a variation of Theorem 2.18 to cover these options through a smaller problem. If $|R| \geq m - 2$, it turns out that one such sub-pattern is enough in the following sense.

Lemma 2.21. *Let $\pi \colon \Pi_{\text{narrowed}} \to 2^{\{0,\dots,m-1\}}$ be a narrowed pattern associated to a feasible R-CCTUF problem of the form in (2.1) with prime modulus m and $|R| \geq m - 2$. Then, we can in strongly polynomial time determine a linear sub-pattern $\widetilde{\pi}$ of π such that one of the following holds:*

- *(i) There are $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha, \beta)| = 1$ so that for any x_A solving the relaxation of the A-problem with respect to (α, β), there is an x_B solving the relaxation of the B-problem such that the combination (x_A, x_B) is feasible for the R-CCTUF problem.*
- *(ii) There is a feasible solution for some pair $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha, \beta)| \geq 2$.*
- *(iii) There is a feasible solution (x_A, x_B) for some pair $(\alpha, \beta) \in \operatorname{dom}(\widetilde{\pi})$ such that $\widetilde{\pi}(\alpha, \beta) = \{\gamma_B^\top x_B\}$.*

Thus, to check feasibility for an R-CCTUF problem of the form (2.1), we can first compute, for each pair $(\alpha, \beta) \in \Pi_{\text{narrowed}}$, a solution x_A to the relaxation of the A-problem with respect to scalar products (α, β) and check whether there is a solution x_B to the B-problem that, combined with x_A, leads to a feasible solution to the initial problem. If this is the case, we are done. Otherwise, we know that (i) of Lemma 2.21 does not hold, and therefore either (ii) or (iii) must hold. Moreover, as previously explained, we call our procedure recursively for pairs $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha, \beta)| \geq 2$, spawning independent and simpler (because we increase the size of the allowed target residues R) subproblems for the A-problem. Hence, if (ii) of Lemma 2.21 applies, then one of these simpler subproblems will lead to a feasible solution to the original problem. Finally, we apply (a slight extension of) Theorem 2.18 using the linear sub-pattern $\widetilde{\pi}$ and Theorem 2.19, thereby reducing to problems with fewer variables. By Lemma 2.21, we know that if there is a feasible solution, we will find one in the described procedure. Altogether, we get to the following theorem.

Theorem 2.22. *Consider an R-CCTUF problem with prime modulus m, n variables, $\ell \in \{m-1, m-2\}$ many target residues, and a constraint matrix T falling into case (iii) of Theorem 2.14. Let $p = \min\{n_A, n_B\}$ be the number of columns of the smaller matrix in the decomposition. After solving less than $3(m-\ell+1)^2$ many R-CCTUF problems with p variables, modulus m, and at most ℓ target residues, we can in strongly polynomial time determine either a solution of the problem, or a family $\mathcal{F}$ of at most*

- *one R-CCTUF problem with at most $n - p + 1$ variables, modulus m, and ℓ target residues, and*
- *$(m-\ell+1)^2$ R-CCTUF problems with $n - p$ variables, modulus m, and at least $\ell + 1$ target residues*

such that the initial R-CCTUF problem is feasible iff at least one problem in $\mathcal{F}$ is feasible. Also, a feasible solution to any problem in $\mathcal{F}$ can in strongly polynomial time be transformed to one of the initial problem.

Finally, it remains to cover the case where the constraint matrix T falls into case (iv) of Theorem 2.14, i.e., only after pivoting, a decomposition step is possible. It turns out that such R-CCTUF problems can be rewritten as a problem of the same type with the pivoted constraint matrix and one extra constraint that is a variable bound, thus subsequently allowing for a decomposition step without interfering with the progress that

was made before (the number of variables and the number of target residues stay the same in the described transformation). The following theorem formalizes this.

Theorem 2.23. *Consider an R-CCTUF problem with constraint matrix T for which case (iv) of Theorem 2.14 applies, i.e., $\mathrm{pivot}_{ij}(T)$ admits a decomposition as a 3-sum according to Theorem 2.14. Then we can in strongly polynomial time determine an R-CCTUF problem of the form*

$$\overline{T}y \leq \overline{b},\ \ y_j \leq \delta,\ \ \overline{\gamma}^\top y \in R \pmod m,\ \ y \in \mathbb{Z}^n\ , \tag{2.5}$$

where $\overline{T}$ is, up to changing the sign in some rows and columns, the matrix $\mathrm{pivot}_{ij}(T)$, and solutions of the initial problem can be transformed to solutions of (2.5) in strongly polynomial time, and vice versa.

Leveraging Theorems 2.14, 2.15, 2.22 and 2.23, we can conclude our main result, Theorem 2.2.

Proof of Theorem 2.2. Consider an R-CCTUF problem with modulus m and $\ell \geq m-2$ target residues. If $\ell = m$, a solution can be found in strongly polynomial time by solving the relaxation of the problem using the framework of Tardos [Tar86]. Else, we apply Theorem 2.14 to the constraint matrix T. If case (i) or (ii) of Theorem 2.14 applies, Theorem 2.15 guarantees that we can efficiently solve the corresponding problem. If case (iv) applies, we can reduce the problem to one where case (iii) applies through Theorem 2.23. Finally, if case (iii) of Theorem 2.14 applies, we apply Theorem 2.22 to reduce the problem to several smaller problems on which we recursively call our procedure. Through these recursive calls, the initial R-CCTUF problem is reduced to several simpler R-CCTUF problems, where each of them has either m many target residues or its constraint matrix is a base block matrix.

We first bound the number of such simpler R-CCTUF problems that we obtain. Let $f(n,\ell)$ be the smallest upper bound on the number of such problems that we obtain through our reductions when starting from an instance with n variables and ℓ target residues. We claim that

$$f(n,\ell) \leq m^{2(m-\ell)} \cdot n^{m-\ell+3\log_2 m+2}\ .$$

Indeed, note that $f(n,\ell) = 1$ for $n \leq 3$ and all $\ell \leq m$, and $f(n,m) = 1$ for all n, and assume that the inequality holds for all instances of up to $n-1$ variables. By Theorem 2.22 and this assumption, we get for some $p \in \{2,\dots,\lfloor n/2 \rfloor\}$:

$$\begin{aligned} f(n,\ell) &\leq 3m^2 f(p,\ell) + f(n-p+1,\ell) + m^2 f(n-p,\ell+1) \\ &\leq m^{2(m-\ell)} n^{m-\ell+3\log_2 m+2} \Bigg(\underbrace{\left(\frac{p}{n}\right)^2 + \left(\frac{n-p+1}{n}\right)^2 + \frac{n-p}{n^2}}_{\leq 1} \Bigg) \leq m^{2(m-\ell)} n^{m-\ell+3\log_2 m+2}\ , \end{aligned}$$

as desired.

Now observe that each of the at most $f(n,\ell)$ many subproblems can either be solved directly in strongly polynomial time as stated earlier (if it is a problem with m target residues), or we can apply the strongly polynomial randomized algorithm provided by Theorem 2.15 to each of them $\log_n(nf(n,\ell)) = O(1)$ many times to correctly solve each problem with error probability at most $1/nf(n,\ell)$. Thus, by a union bound, we can solve all these problems (and thus the initial problem) correctly with probability $1 - 1/n$. To finish the proof, it remains to observe that the time for solving the discussed problems clearly dominates the time needed for transformations and solution propagation. □

2.3 Proof and further implications of the decomposition lemma

For the sake of presentation, we postpone the proof of the decomposition lemma (Lemma 2.7) and Lemma 2.9 to the end of this section and start by showing additional implications, namely Theorem 2.3 and Lemma 2.16.

2.3.1 An alternative approach to R-CCTUF problems with $|R| = m - 1$: Proving Theorem 2.3

In this section, we prove that R-CCTUF problems with $|R| = m - 1$ can be solved deterministically and in strongly polynomial time, as stated by Theorem 2.3. This result is closely linked to our flatness statement, Theorem 2.4, which already guarantees that if none of the constraint matrix rows of the R-CCTUF problem is a flat direction of the underlying polyhedron with width $m - |R| - 1$, then the problem can be solved efficiently. For $|R| = m - 1$, the width in this statement is 0, i.e., the corresponding constraint is a tight constraint for the full underlying polyhedron. Using Theorem 2.19, we can see that in this case of non-full-dimensional underlying polyhedra, we can easily project to a lower-dimensional space.

Lemma 2.24. *Consider an R-CCTUF problem in $n \geq 2$ variables with a constraint that is tight for all points in the underlying polyhedron. We can in strongly polynomial time determine an R-CCTUF problem in $n-1$ variables such that solutions of the first problem can be transformed to solutions of the second problem in strongly polynomial time, and vice versa.*

Proof. After permuting variables and constraints, we may assume that the inequality system in the given R-CCTUF problem has the form

$$\begin{pmatrix} \overline{T} & a_1 \\ a_2^\top & \alpha \end{pmatrix} \begin{pmatrix} \overline{x} \\ x_n \end{pmatrix} \leq \begin{pmatrix} \overline{b} \\ b_n \end{pmatrix}, \quad \text{where} \quad T = \begin{pmatrix} \overline{T} & a_1 \\ a_2^\top & \alpha \end{pmatrix}, \quad x = \begin{pmatrix} \overline{x} \\ x_n \end{pmatrix}, \quad \text{and} \quad b = \begin{pmatrix} \overline{b} \\ b_n \end{pmatrix},$$

such that $a_2^\top \overline{x} + \alpha x_n = b_n$ is a constraint that is tight for any solution to the relaxation of the R-CCTUF problem and $\alpha \neq 0$. By Theorem 2.19, $(\overline{y}, y_n)$ is a solution of the above system if and only if $\overline{y}$ solves the TU system $\left(\overline{T} - \alpha a_1 a_2^\top\right) \overline{x} \leq \overline{b}$, and $y_n = \alpha b_n - \alpha a_2^\top \overline{y}$. Therefore, the original R-CCTUF problem can be reduced in strongly polynomial time to the following R-CCTUF problem with only $n-1$ variables:

$$\overline{T}\overline{x} \leq \overline{b}, \;\; (\overline{\gamma} - \alpha\gamma_n a_2)^\top \overline{x} \not\equiv r - \alpha\gamma_n b_n \pmod{m}, \;\; \overline{x} \in \mathbb{Z}^{n-1} \;. \qquad \square$$

Although not exploited here, we remark that the above reduction of non-full-dimensional problems also applies to the optimization version of the considered problem. Now, combining Lemma 2.24 and Theorem 2.4, we immediately obtain a proof of Theorem 2.3.

Proof of Theorem 2.3. First of all, we observe that using a result of Tardos [Tar86], we can solve linear programs over the underlying polyhedron of a given R-CCTUF problem in strongly polynomial time, and hence, we can also detect in strongly polynomial time whether there is a tight constraint. If there is no tight constraint, then the problem can be solved by Theorem 2.4. Otherwise, the problem is trivial when $n = 1$, and if $n \geq 2$, we can repeatedly apply Lemma 2.24 until we obtain a problem with $n = 1$, or one that does not have tight constraints. Note that the number of variables reduces by 1 in each application of Lemma 2.24, hence there are less than n iterations. We conclude the proof by observing that solutions of a problem without tight constraints that stem from Lemma 2.24 can be transformed back to solutions of the initial problem in strongly polynomial time by the very same lemma. $\square$

2.3.2 Bounded scalar products

The goal of this subsection is to deduce Lemma 2.16, which we use to restrict the search space for solutions of R-CCTUF problems. It turns out that this lemma is an implication of a more general result that we expand on below.

Lemma 2.25. *Consider a feasible R-CCTUF problem with constraint matrix T and modulus m, and let d be TU-appendable to T. We can determine in strongly polynomial time $\ell, u \in \mathbb{Z}$ with $u - \ell \leq m - |R|$ such that the R-CCTUF problem has a feasible solution x_0 if and only if it has one with $\ell \leq d^\top x_0 \leq u$.*

Proof. Let $Tx \leq b$ be the inequality system in the R-CCTUF problem. To start with, we can in strongly polynomial time determine $\eta_{\max} \coloneqq \max\{d^\top x \colon Tx \leq b,\ x \in \mathbb{R}^n\}$ and $\eta_{\min} \coloneqq \min\{d^\top x \colon Tx \leq b,\ x \in \mathbb{R}^n\}$. If $\eta_{\max} - \eta_{\min} \leq m - |R|$, we can choose $u = \eta_{\max}$ and $\ell = \eta_{\min}$, and there is nothing to show. Otherwise, we claim that the statement holds for any choice of $\ell, u \in \{\eta_{\min}, \dots, \eta_{\max}\}$ with $u - \ell \leq m - |R|$. To see this, consider any such choice of ℓ and u and consider the given R-CCTUF problem with the constraints $\ell \leq d^\top x \leq u$ added to the inequality system. Because by construction, d is a flat direction of width exactly $m - |R|$ for that problem, applying twice Lemma 2.10 (once for each of the two constraints that we added) gives that the problem with the constraints added is feasible if and only if the original one is. □

Note that if we are given vectors $d_1, \dots, d_p$ that are all simultaneously TU-appendable to the constraint matrix of the problem, we can apply Lemma 2.25 iteratively with the TU-appendable vectors d_i, adding the obtained constraints $\ell_i \leq d_i^\top x \leq u_i$ to the system in each step. This immediately implies the following corollary.

Corollary 2.26. *Consider a feasible R-CCTUF problem with constraint matrix T and modulus m, and let $d_1, \dots, d_p$ be simultaneously TU-appendable to T. We can determine in strongly polynomial time $\ell_i, u_i \in \mathbb{Z}$ with $u_i - \ell_i \leq m - |R|$ for $i \in [p]$ such that the R-CCTUF problem has a feasible solution x_0 if and only if it has one with $\ell_i \leq d^\top x_0 \leq u_i$ for all $i \in [p]$.*

Now Lemma 2.16 follows immediately from Corollary 2.26 after observing the following.

Observation 2.27. *Consider a matrix T that is a 3-sum of the form $T = \left(\begin{smallmatrix} A & ef^\top \\ gh^\top & B \end{smallmatrix}\right)$. Then, the rows $\begin{pmatrix} 0 & f^\top \end{pmatrix}$, $\begin{pmatrix} h^\top & 0 \end{pmatrix}$, and $\begin{pmatrix} h^\top & f^\top \end{pmatrix}$ are simultaneously TU-appendable to T.*

Proof. Observe that

$$\begin{pmatrix} A & ef^\top \\ 0 & f^\top \\ h^\top & f^\top \\ h^\top & 0 \\ gh^\top & B \end{pmatrix} = \begin{pmatrix} A & e & e \\ 0 & 1 & 1 \\ h^\top & 0 & 1 \end{pmatrix} \oplus_3 \begin{pmatrix} 0 & 1 & f^\top \\ 1 & 1 & f^\top \\ 1 & 1 & 0 \\ g & g & B \end{pmatrix}. \tag{2.6}$$

Recall that because the totally unimodular matrix T decomposes into a 3-sum of the two matrices $\left(\begin{smallmatrix} A & e & e \\ h^\top & 0 & 1 \end{smallmatrix}\right)$ and $\left(\begin{smallmatrix} 0 & 1 & f^\top \\ g & g & B \end{smallmatrix}\right)$, we know that these matrices are totally unimodular, as well. It can be easily seen that this implies total unimodularity of the two summands in (2.6), and hence also of the 3-sum of the two matrices. □

Proof of Lemma 2.16. By Corollary 2.26 above, it is enough to show that the vectors $\begin{pmatrix} 0 & f^\top \end{pmatrix}$, $\begin{pmatrix} h^\top & 0 \end{pmatrix}$, and $\begin{pmatrix} h^\top & f^\top \end{pmatrix}$ are simultaneously TU-appendable to T. The latter is true, as seen in Observation 2.27 above. □

Finally, we note that the assumption of simultaneous TU-appendability in Corollary 2.26 is necessary to obtain ranges of width $m - |R|$ for each scalar product. More generally, if we want to obtain bounds simultaneously for all TU-appendable vectors, our general proximity result, Theorem 2.11, only implies ranges of width $2(m - |R|) + 1$.

2.3.3 Proof of the decomposition lemma (Lemma 2.7) and Lemma 2.9

In order to prove Lemma 2.7 we first show a key property of pointed polyhedral cones defined by TU matrices (which we also call *TU cones*), from which will later derive Lemma 2.7. To this end, we recall that, for a polyhedral cone $C \coloneqq \{x \in \mathbb{R}^n \colon Ax \leq 0\}$ (where $A \in \mathbb{Q}^{k \times n}$), an *extremal ray* of C is a non-zero vector $r \in C$ that lies on a 1-dimensional face of C. Moreover, we use the following notion of *elementary extremal ray*.

Definition 2.28 (Elementary extremal ray)**.** *An extremal ray r of a polyhedral cone $C \subseteq \mathbb{Z}^n$ is* elementary *if $r \in \mathbb{Z}^n$ and the greatest common divisor of the coordinates of r is one.*

Hence, for every rational cone C and every extremal ray r of the cone, there is some unique $\lambda > 0$ such that $\lambda \cdot r$ is an elementary extremal ray of C.

Lemma 2.29 below shows that any point in a pointed cone C that is defined by a TU matrix can be integrally decomposed into few elementary extremal rays in strongly polynomial time. We highlight that the crucial part of Lemma 2.29 is that the coefficients λ_i can be chosen to be integral. Note that, despite the cone being defined by a TU matrix, the elementary extremal rays in Lemma 2.29 have to be well-chosen because the set of elementary extremal rays of C does not form a totally unimodular matrix.[7] Hence, even if a set of n elementary extremal rays of C spans y, it may be that the decomposition of y into a conic combination of these elementary extremal rays requires non-integral coefficients. (This is arguably the case to be expected without choosing the rays carefully.)

Lemma 2.29. *Let $T \in \{-1, 0, 1\}^{k \times n}$ be a totally unimodular matrix such that the cone $C := \{x \in \mathbb{R}^n \colon Tx \leq 0\}$ is pointed, and let $y \in C \cap \mathbb{Z}^n$. Then one can determine in strongly polynomial time elementary extremal rays $y^1, \dots, y^n \in \mathbb{Z}^n$ of C and coefficients $\lambda_1, \dots, \lambda_n \in \mathbb{Z}_{\geq 0}$ such that $y = \sum_{i=1}^n \lambda_i y^i$.*

Proof. We prove the statement by determining successively pairs (λ_i, y^i) of the desired decomposition of y. We start by explaining how we compute λ_1 and y^1, and then highlight how to iterate the procedure to obtain the full decomposition of y. To obtain a first coefficient λ_1 and vector y^1 of the desired decomposition of y, we define an auxiliary polytope P_1 by

$$P_1 := C \cap C_1 \quad , \quad \text{where} \quad C_1 := \{x \in \mathbb{R}^n \colon -Tx \leq -Ty\} \quad .$$

Hence,

$$P_1 := \left\{ x \in \mathbb{R}^n \colon \begin{pmatrix} T \\ -T \end{pmatrix} x \leq \begin{pmatrix} 0 \\ -Ty \end{pmatrix} \right\} \quad .$$

Note that C_1 can be interpreted as a reversed version of C with apex at y. Also note that P_1 is a polytope because C is pointed. Indeed, if P_1 were unbounded, there would need to be a non-zero vector $r \in \mathbb{R}^n$ with $Tr \leq 0$ and $-Tr \leq 0$, which implies $Tr = 0$ and contradicts that C is pointed. Moreover, as highlighted above, observe that P_1 can be described by the constraint matrix $\binom{T}{-T}$, which is TU.

Let $T^=$ be the set of constraints of C that are tight at y. Hence, $T^= y = 0$. Similarly, let $T^<$ denote the remaining constraints of C, which are the ones not tight at y. Hence, $T^< y < 0$. In addition, without loss of generality, we may assume that the rows in $T^<$ are linearly independent from those of $T^=$; for otherwise they are redundant and we can drop them. Let y^1 be any extremal ray of

$$Q_1 := \{x \in \mathbb{R}^n \colon T^= x = 0, T^< x \leq 0\} \quad .$$

Note that Q_1 is pointed because $Q_1 \subseteq C$ and C is pointed; thus, it has extremal rays. Such an extremal ray y^1 can be computed efficiently via standard techniques.[8] By rescaling y^1, we can assume without loss of generality that $y^1 \in \mathbb{Z}^n$ is an elementary extremal ray of Q_1. Let

$$\lambda_1 := \max \left\{ \lambda \in \mathbb{R}_{\geq 0} \colon -T^<(\lambda y^1) \leq -T^< y \right\} \quad ,$$

[7] Indeed, cones defined by TU matrices can have exponentially many elementary extremal rays. This follows for example by the well-known fact that the bipartite matching polytope P, which can be described by a TU matrix, has vertices $v \in \mathrm{vertices}(P)$ with exponentially many edges incident to them. Hence, the set of constraints of P that are tight at v define a TU cone (when shifted such that v becomes the origin) with exponentially many elementary extremal rays.

[8] Any vertex $u \in \mathbb{R}^n_{\geq 0}$ of the polytope $P' := Q_1 \cap \{x \in \mathbb{R}^n \colon 1^\top x \leq 1\}$, with $u \neq 0$, induces an extremal ray of Q_1. Hence, it is enough to compute an optimal vertex solution of the linear program $\max\{1^\top x \colon x \in P'\}$, which can be done in polynomial time via standard methods. Note that all numbers/coefficients involved in this linear program are small (actually they are all within $\{-1, 0, 1\}$). Hence, the runtime is thus trivially strongly polynomial in the original input size.

that is, λ_1 captures how far in the direction of the elementary extremal ray y^1 we can go, when starting from the origin, while staying within P_1. The constraints of the above optimization problem are of the form $\lambda a_i \leq b_i$ for $i \in [\ell]$, with $a_i \coloneqq -(T^{<}y^1)_i$ and $b_i \coloneqq -(T^{<}y)_i$. By definition of $T^{<}$, we have $T^{<}y < 0$, and thus $b_i > 0$ for all $i \in [\ell]$. Hence,

$$\lambda_1 = \min\left\{ \frac{b_i}{a_i} : i \in [\ell] \text{ with } a_i > 0 \right\} \ ,$$

which shows that λ_1 can be computed in strongly polynomial time by first computing a_i and b_i for $i \in [\ell]$ and then determining the minimizing ratio b_i/a_i.

Note that $\lambda_1 y^1$ must be a vertex of P_1. This follows because $\lambda_1 y^1 \in P_1$ by construction, and y^1 is an extremal ray of Q_1 (it thus lies on a face of Q_1 of dimension 1), and therefore y^1 is also an extremal ray of P_1 because Q_1 is a face of P_1.[9] Hence, $\lambda_1 y^1$ is a face of P_1 of dimension 0, i.e., $\lambda_1 y^1 \in \text{vertices}(P_1)$. Moreover, because P_1 is described by a TU system, its set of vertices must be all integral, and hence $\lambda_1 y^1 \in \mathbb{Z}^n$. Furthermore, we must also have that $\lambda_1 \in \mathbb{Z}_{\geq 0}$. If not, then we can write $\lambda_1 = p/q$ with $p, q \in \mathbb{Z}_{\geq 0}$ such that their greatest common divisor $\gcd(p, q)$ equals 1 and $q \geq 2$. As $\lambda_1 y^1 \in \mathbb{Z}^n$, we must have that q divides $p y^1_i$ for all $i \in [n]$. However, this implies that q divides y^1_i for all $i \in [n]$, which follows from $\gcd(p, q) = 1$ and a well-known basic number theory result.[10] But this contradicts with y^1 being elementary.

We now proceed inductively on the vector $y' \coloneqq y - \lambda_1 y^1$. Note that by construction we have $Ty' \leq 0$, and can thus reiterate the above-explained approach with the vector y' instead of y. Let $T_1^{=}$ be the rows of T that correspond to y'-tight constraints of $Tx \leq 0$; hence, $T_1^{=}y' = 0$. Analogously as before, let $T_1^{<}$ be the other rows, which correspond to constraints of $Tx \leq 0$ that are not y'-tight. As before, we then define

$$Q_2 \coloneqq \{x \in \mathbb{R}^n : T_1^{=}x = 0, T_1^{<}x \leq 0\} \ ,$$

compute an elementary extremal ray of Q_2 and continue as above. Note that $\dim(Q_2) < \dim(Q_1)$, because $y' \coloneqq y - \lambda_1 y^1$ was chosen such that a new constraint of $Tx \leq 0$ that was not y-tight became y'-tight. Hence, this procedure will terminate after at most $\dim(Q_1) \leq n$ many iteration. If the procedure terminates in less than n iterations, in which case we get a decomposition with fewer than n terms, we can add arbitrary extremal rays with zero coefficients to the decomposition to obtain the claimed n many terms. □

The following statement shows that elementary extremal rays of a TU cone are elementary with respect to the TU matrix defining the cone. This property links the notions of elementary extremal ray and of being elementary with respect to a TU matrix.

Lemma 2.30. *Let $T \in \{-1, 0, 1\}^{k \times n}$ be a totally unimodular matrix and $r \in \mathbb{Z}^n$ be an elementary extremal ray of $C \coloneqq \{x \in \mathbb{R}^n : Tx \leq 0\}$. Then r is elementary with respect to T.*

Proof. With the goal of deriving a contradiction, assume that there is a vector $d \in \{-1, 0, 1\}^n$ that is TU-appendable to T and such that $\eta \coloneqq d^\top r \notin \{-1, 0, 1\}$. Without loss of generality, we assume $\eta > 0$, which can be achieved by replacing d by $-d$ if necessary. We denote by

$$L \coloneqq \{\lambda r : \lambda \geq 0\}$$

the 1-dimensional face of C on which r lies.

Note that $(1/\eta) \cdot r$ lies in the following polyhedron, which is defined by a TU constraint matrix (due to TU-appendability of d):

$$Z \coloneqq \left\{x \in \mathbb{R}^n : Tx \leq 0, d^T x = 1\right\} \ .$$

[9] Here we use the basic polyhedral fact that a face of a face of a polyhedron is a face of the polyhedron.

[10] More precisely, we use that for any $a, b, c \in \mathbb{Z}$ with $\gcd(a, b) = 1$, if a divides bc then a divides c.

Hence, $(1/\eta) \cdot r$ can be written as a convex combination of integer points in Z, say

$$\frac{1}{\eta} \cdot r = \sum_{j=1}^{q} \mu_j z_j \ , \tag{2.7}$$

with $\mu_j \geq 0, z_j \in Z \cap \mathbb{Z}^n$ for $j \in [q]$, and $\sum_{j=1}^{q} \mu_j = 1$. Observe that $(1/\eta) \cdot r$ is the only point on L that is also in Z because $d^\top r \neq 0$, i.e.,

$$L \cap Z = \{(1/\eta) \cdot r\} \ .$$

As $(1/\eta) \cdot r \notin \mathbb{Z}^n$, because r is elementary and $\eta > 1$, we have

$$z_j \notin L \quad \forall j \in [q] \ .$$

However, this leads to a contradiction because it implies that the decomposition (2.7) expresses a point on the 1-dimensional face L of C as a convex combination of points in C, none of which lies on L. This is impossible because any convex combination that describes a point on a 1-face of a polyhedron needs to use terms on the same face. □

We are now ready to prove Lemma 2.7.

Proof of Lemma 2.7. Because the statement is invariant under a shift of the coordinate system, we can assume $x_0 = 0$ for convenience. (Formally, instead of considering $Tx \leq b$ and x_0, y, we consider the system $Tx \leq b - Tx_0$ and replace x_0 and y by the origin and $y - x_0$, respectively.) Moreover, we observe that we can assume that the system $Tx \leq b$ contains, for each $i \in [n]$, the constraint

$$\begin{cases} x_i \geq 0 & \text{if } y_i \geq 0 \ , \\ x_i \leq 0 & \text{if } y_i < 0 \ . \end{cases} \tag{2.8}$$

Indeed, by adding these constraints, the thus obtained system $\widetilde{T}x \leq \widetilde{b}$ is still a TU system for which both the origin and y are feasible. Moreover, a decomposition of y with respect to this new system $\widetilde{T}x \leq \widetilde{b}$ has the desired properties because a vector is TU-appendable to T if and only if it is TU-appendable to $\widetilde{T}$, which implies that a vector is elementary w.r.t. T if and only if it is elementary w.r.t. $\widetilde{T}$.[11] Hence, we assume from now on that $Tx \leq b$ contains the constraints (2.8), which implies that T has full column rank.

We now define a TU matrix $\overline{T} \in \{-1, 0, 1\}^{k \times n}$ which is obtained from T by changing the sign of some of its rows. More precisely for each row $w^\top$ of T, the matrix $\overline{T}$ contains a row

$$\begin{cases} w^\top & \text{if } w^\top y \leq 0 \ , \\ -w^\top & \text{if } w^\top y > 0 \ . \end{cases}$$

We define

$$C \coloneqq \left\{x \in \mathbb{R}^n \colon \overline{T}x \leq 0\right\} \ .$$

Note that C is pointed because $\overline{T}$ has full column rank, which follows from T having full column rank. We now apply Lemma 2.29 to the TU matrix $\overline{T}$ and point y. This leads to a decomposition of y as $y = \sum_{i=1}^{n} \lambda_i y^i$ such that, for $i \in [n]$, we have $\lambda_i \in \mathbb{Z}_{\geq 0}$ and y^i is an elementary extremal ray of C. We claim that this decomposition has the desired properties.

Note that by Lemma 2.30, each vector y^i for $i \in [n]$ is elementary with respect to $\overline{T}$. It is therefore also elementary with respect to T, because $\overline{T}$ and T have the same set of TU-appendable rows as they are the same matrices up to sign changes of some of the rows.

[11] The fact that TU-appendability to T is the same as TU-appendability to $\widetilde{T}$ is an immediate consequence of the fact that adding rows that are all-zero except for a single 1 or -1 entry to any TU matrix preserves TU-ness.

It remains to show that for any coefficients $\mu_1, \dots, \mu_n \in \mathbb{Z}_{\geq 0}$ with $\mu_i \leq \lambda_i$ for $i \in [n]$, we have that the vector

$$\widetilde{y} \coloneqq x_0 + \sum_{i=1}^{n} \mu_i y^i = \sum_{i=1}^{n} \mu_i y^i$$

satisfies $T\widetilde{y} \leq b$. To this end consider a constraint $w^\top x \leq \beta$ of the system $Tx \leq b$. We distinguish between whether $w^\top$ or $-w^\top$ is a row of $\overline{T}$. If $w^\top$ is a row of $\overline{T}$, then

$$w^T \widetilde{y} = \sum_{i=1}^{n} \mu_i w^\top y^i \leq 0 \leq \beta \ ,$$

where the first inequality follows from $w^T y^i \leq 0$ because y^i is a ray of C, and the second inequality follows from the fact that the origin is feasible for the system $Tx \leq b$, which implies that all right-hand sides are non-negative.

Consider now the case where $-w^\top$ is a row of $\overline{T}$. Then we have

$$w^\top \widetilde{y} = w^\top y - \sum_{i=1}^{n} (\lambda_i - \mu_i)\, w^\top y^i \leq w^\top y \leq \beta \ ,$$

where the first inequality follows from $\lambda_i \geq \mu_i$ together with $w^\top y^i \geq 0$, which holds because $\overline{T} y^i \leq 0$ and $\overline{T}$ contains the row $-w^\top$, and the last inequality follows from $Ty \leq b$, which contains the constraint $w^\top y \leq \beta$. Hence, $\widetilde{y}$ fulfills all constraints of the system $Tx \leq b$, as desired, which finishes the proof. □

Proof of Lemma 2.9. By applying Lemma 2.7 to the solutions y and x_0 of the system $Tx \leq b$ of the given R-CCTUF problem, we obtain in strongly polynomial time $y^1, \dots, y^n \in \mathbb{Z}^n$ and $\lambda_1, \dots, \lambda_n \in \mathbb{Z}_{\geq 0}$ such that $y = x_0 + \sum_{i=1}^n \lambda_i y^i$ and (i) $d^\top y^i \in \{-1, 0, 1\}$ for all $i \in [n]$ and all d that are TU-appendable to T, and (ii) $\tilde{y} = x_0 + \sum_{i=1}^n \mu_i y^i$ is feasible for $Tx \leq b$ for any choice of $\mu_i \in \{0, \dots, \lambda_i\}$. By these properties, in order to prove Lemma 2.9, it is enough to identify in strongly polynomial time $\mu_i \in \{0, \dots, \lambda_i\}$ with $\sum_{i=1}^n \mu_i < m - |R|$ such that $\gamma^\top \tilde{y} = \gamma^\top x_0 + \sum_{i=1}^n \mu_i \gamma^\top y^i \in R \pmod m$. Denoting $\Lambda = \sum_{i=1}^n \lambda_i$ and

$$R' = \{(r - \gamma^\top x_0 \bmod m) \colon r \in R\} \ , \quad \text{as well as} \quad \begin{aligned} r_1 = \ldots = r_{\lambda_1} &= \gamma^\top y^1 \ , \\ r_{\lambda_1+1} = \ldots = r_{\lambda_1+\lambda_2} &= \gamma^\top y^2 \ , \\ &\vdots \\ r_{\lambda_1+\ldots+\lambda_{n-1}+1} = \ldots = r_\Lambda &= \gamma^\top y^n \ , \end{aligned} \tag{2.9}$$

we can formulate this problem as follows: We start from the sum $\sum_{i \in S_0} r_i \in R' \pmod m$ with $S_0 = \{1, \dots, \Lambda\}$, and our goal is to identify a subset $S \subseteq S_0$ of size at most $m - |R| = m - |R'|$ such that $\sum_{i \in S} r_i \in R' \pmod m$, as well. By Lemma 2.8, we know that if $|S_0| > m - |R'|$, there exists an interval $I_1 = \{i_1^1, \dots, i_2^1\}$ with $i_1^1, i_2^1 \in S_0$ and $i_1^1 < i_2^1$ such that for $S_1 = S_0 \setminus I_1$, we have $\sum_{i \in S_1} r_i \in R' \pmod m$. Iterating this argument, we obtain that for $j = 1, 2, \dots$ and while $|S_{j-1}| > m - |R'|$, there exists an interval $I_j = \{i_1^j, \dots, i_2^j\}$ with $i_1^j, i_2^j \in S_0$ and $i_1^j < i_2^j$ such that $I_j \cap S_{j-1} \neq \emptyset$, and for $S_j = S_{j-1} \setminus I_j$, we have $\sum_{i \in S_j} r_j \in R' \pmod m$. For clarity, we remark that in step j, we are removing the terms with indices in $S_{j-1} \cap I_j$ from the sum. Moreover, while these indices are consecutive in the sum that we consider in step j, they may not be so in the original sum $\sum_{i=1}^n r_i$. Also, this means that an index $i \in \{1, \dots, \Lambda\}$ may well be contained in several intervals I_j.

Because $I_j \cap S_{j-1} \neq \emptyset$, the number of terms in the sum strictly decreases in every step, so the procedure terminates, which shows existence of the desired solution $\tilde{y}$, as already pointed out in Section 2.2.1. To arrive at a suitably short sum in strongly polynomial time, we split the deletion process into two phases:

Phase 1: Steps j such that $|S_{j-1}| > m-1$, i.e., the sum has more than $m-1$ terms.
Hence, the above arguments can be applied with R' replaced by the singleton set $\{(\sum_{i\in S_0} r_i \bmod m)\}$ such that the sums $\sum_{i\in S_j} r_i$ obtained in this phase all have the same residue. Equivalently, terms that sum to $0 \pmod m$ are removed in every step, i.e., $\sum_{i\in S_{j-1}\cap I_j} r_i \equiv 0 \pmod m$.

Phase 2: Steps j such that $|S_{j-1}| \leq m-1$, i.e., the sum has at most $m-1$ terms.
In this case, at most $|R|-1$ further deletion steps suffice to reduce to at most $m-|R|$ many terms.

A way to perform the steps in strongly polynomial time both in phase 1 and phase 2, as well as a strongly polynomial bound on the number of steps in phase 1 is provided by the following two claims:

(i) We can, in every step of the described procedure and in strongly polynomial time, determine an interval to delete of maximum possible size, i.e., determine I_j such that $|S_{j-1}\cap I_j|$ is maximized.
(ii) If in every step, I_j is chosen according to point (i), the procedure ends after at most n steps.

Together, (i) and (ii) immediately prove Lemma 2.9. To proof the two claims, let us start with focusing on claim (ii). First, we observe that in phase 1, choosing I_j to maximize $|S_{j-1}\cap I_j|$ implies that no two intervals will overlap, i.e., $I_j \cap I_k = \emptyset$ for all intervals I_j and I_k that we construct in this phase. To see this, assume for the sake of deriving a contradiction that I_ℓ is an interval that overlaps with some earlier intervals $I_{j_1}, \dots, I_{j_t}$ with $j_1 < \dots < j_t < \ell$, and choose the minimum ℓ with this property. In particular, we thus know that the intervals $I_{j_1}, \dots, I_{j_t}$ do not overlap with each other and with any other intervals I_j with $j < \ell$. This implies that in step j_1, $I' \coloneqq I_\ell \cup I_{j_1} \cup \dots \cup I_{j_t}$ is a candidate interval: Indeed, taking I' would remove the terms

$$\sum_{i\in S_{j_1-1}\cap I'} r_i = \sum_{i\in I'} r_i = \sum_{p=1}^{t}\sum_{i\in I_{j_p}} r_i + \sum_{i\in I_\ell \setminus \bigcup_{p=1}^{t} I_{j_p}} r_i = \sum_{p=1}^{t}\sum_{i\in S_{j_p-1}\cap I_{j_p}} r_i + \sum_{i\in S_{\ell-1}\cap I_\ell} r_i \equiv 0 \pmod m \ ,$$

where we use that I_ℓ is the first interval that overlaps with other intervals, and that because we are in phase 1, each individual sum in the last expression is $0 \pmod m$. Moreover, note that $S_{j_1-1}\cap I_{j_1} \subsetneq S_{j_1-1}\cap I'$, hence $|S_{j_1-1}\cap I_{j_1}| < |S_{j_1-1}\cap I'|$, contradicting the choice of I_{j_1} to maximize $|S_{j_1-1}\cap I_{j_1}|$. Thus, the intervals I_j obtained in phase 1 are all disjoint, hence in particular, we always have $S_{j-1}\cap I_j = I_j$, i.e., in step j, we remove precisely the terms with indices in I_j from the sum.

Next, recall the way that residues r_i were defined in (2.9): They come in n chunks of equal residues, namely with indices in $C_1 = \{1, \dots, \lambda_1\}$, $C_2 = \{\lambda_1+1, \dots, \lambda_1+\lambda_2\}$, $\dots$, $C_n = \{\lambda_1+\dots+\lambda_{n-1}, \dots, \Lambda\}$. We observe that each of those chunks C_i can contain at most 2 endpoints of intervals I_j that are constructed during phase 1. To see this, assume for the sake of deriving a contradiction that one C_ℓ contains at least three interval endpoints. We distinguish two cases:

- C_ℓ contains both endpoints of an interval $I_j = \{i_1^j, \dots, i_2^j\}$, and (at least) one endpoint of $I_k = \{i_1^k, \dots, i_2^k\}$. Intervals do not overlap, so assume without loss of generality that $i_2^j < i_1^k$ and choose k such that i_1^k is smallest possible. We claim that instead of I_j or I_k (whichever was deleted first), we could also have chosen the larger interval $I' = \{i_1^k - i_2^j + i_1^j, \dots, i_2^k\}$: Indeed,

$$\sum_{i\in I'} r_i = \sum_{i=i_1^k-i_2^j+i_1^j-1}^{i_1^k-1} r_i + \sum_{i=i_1^k}^{i_2^k} r_i = \sum_{i\in I_j} r_i + \sum_{i\in I_k} r_i \equiv 0 \pmod m \ ,$$

  where we use that $r_i = r_{i'}$ for all $i, i' \in C_\ell$, and that because we are in phase 1, each individual sum in the last expression is $0 \pmod m$. Because $|I_j|, |I_k| < |I'|$, this contradicts the choice of intervals I_j such that $|S_{j-1}\cap I_j| = |I_j|$ is maximized.
- C_ℓ does not contain both endpoints of any interval I_j. This implies that every interval that has one endpoint in C_ℓ contains at least one of the minimum or maximum indices in C_ℓ. Consequently, if C_ℓ contains at least three endpoints, one of these two indices is covered by at least two intervals, contradicting that intervals are disjoint in phase 1.

This proves that every C_i can contain at most 2 endpoints of intervals constructed in phase 1, hence there can be at most n such intervals, and phase 1 ends after at most n steps. This proves claim (ii).[12]

Finally, and to complete the proof of Lemma 2.9, we focus on claim (i) above, i.e., on how to efficiently find intervals I_j maximizing $|S_{j-1} \cap I_j|$. To this end, let us recall what the situation is: We are given $r_1, \ldots, r_\Lambda$ as defined in (2.9), and a set S of target residues (in phase 1, S will contain a single residue; in phase 2, it will be equal to R' from (2.9)) such that $\sum_{i\in[\Lambda]} r_i \in S \pmod m$, and the goal is to identify an interval $I = \{i_1, \ldots, i_2\} \subseteq [\Lambda]$ such that $\sum_{i\in[\Lambda]\setminus I} r_i \in S \pmod m$, and $|I|$ has maximum possible size. Observe that if we update the values λ_i, Λ, and r_i accordingly to reflect the remaining sum after each step of the procedure, this is the precise setup that we are faced with in each step. In what follows, we show that an optimal interval $I = \{i_1, \ldots, i_2\}$ can be identified after solving $O(n^2|S|)$ many IPs with a constant number of variables and a constant number of constraints.

To see this, let C_j and C_k (as defined earlier), with $1 \le j \le k \le n$, be such that $i_1 \in C_j$ and $i_2 \in C_k$. If $j < k$, then $i_1 = \sum_{i=1}^{j-1} \lambda_i + x$ for some $x \in \{1, \ldots, \lambda_j\}$, and $i_2 = \sum_{i=1}^{k-1} \lambda_i + y$ for some $y \in \{1, \ldots, \lambda_k\}$,

$$\sum_{i\in[\Lambda]\setminus I} r_i = \sum_{i\in[\Lambda]} r_i - \left((\lambda_j - x + 1) r_{\lambda_1+\ldots+\lambda_j+1} + \sum_{i=j+1}^{k-1} \lambda_i r_{\lambda_i+\ldots+\lambda_i+1} + y r_{\lambda_1+\ldots+\lambda_k+1} \right) ,$$

and thus

$$\sum_{i\in[\Lambda]\setminus I} r_i \in S \pmod m \quad \Longleftrightarrow \quad -x r_{\lambda_1+\ldots+\lambda_j+1} + y r_{\lambda_1+\ldots+\lambda_k+1} \in S' \pmod m ,$$

where S' is a shifted version of S. Moreover, observe that $|I| = \lambda_j - x + 1 + \sum_{i=j+1}^{k-1} \lambda_i + y$, hence $|I|$ is of maximum size if $y - x$ is maximized. Altogether, we obtain that x and y are optimal solutions of

$$\begin{array}{lrcl} \min_{s\in S'} \quad \max & y - x & & \\ & -x r_{\lambda_1+\ldots+\lambda_j+1} + y r_{\lambda_1+\ldots+\lambda_k+1} & = & zm + s \\ & x & \in & \{1, \ldots, \lambda_j\} \\ & y & \in & \{1, \ldots, \lambda_k\} \\ & z & \in & \mathbb{Z} \ . \end{array} \tag{2.10}$$

Similarly, if $j = k$, then $i_1 = \sum_{i=1}^{j-1} \lambda_i + x$ and $i_2 = \sum_{i=1}^{j-1} \lambda_i + y$ for some $x, y \in \{1, \ldots, \lambda_j\}$ with $x \le y$, and we have

$$\begin{aligned} \sum_{i\in[\Lambda]\setminus I} r_i = \sum_{i\in[\Lambda]} r_i - (y - x + 1) r_{\lambda_1+\ldots+\lambda_j+1} \in S \pmod m \\ \Longleftrightarrow \quad (y - x) r_{\lambda_1+\ldots+\lambda_j+1} \in S' \pmod m , \end{aligned}$$

where again, S' is a shifted version of S. Moreover, $|I| = y - x + 1$, hence $|I|$ is of maximum size if $y - x$ is maximized. Thus, we obtain that x and y are optimal solutions of

$$\begin{array}{lrcl} \min_{s\in S'} \quad \max & y - x & & \\ & (y - x) r_{\lambda_1+\ldots+\lambda_j+1} & = & zm + s \\ & x & \le & y \\ & x, y & \in & \{1, \ldots, \lambda_j\} \\ & z & \in & \mathbb{Z} \ . \end{array} \tag{2.11}$$

Finally, observe that to solve the problems in (2.10) and (2.11), it is enough to solve the inner maximization problem for every $s \in S'$. Given s, these maximization problems are integer programs with 3 variables and

[12] We remark that a slightly more careful analysis, in particular of endpoints in C_1 and C_n, immediately improves this bound to $n - 1$, but this is not needed for our purpose.

a constant number of constraints, and can thus be solved in time polynomial in the encoding size of the IP using Lenstra's algorithm [Len83], which is strongly polynomial in the size of the R-CCTUF problem. Moreover, for fixed j and k, it is immediate that a solution of (2.10) or (2.11) (if it exists) corresponds to an largest possible interval I with endpoints in C_j and C_k. Altogether, by going through the $O(n^2)$ many options for $j, k \in [n]$, we can in strongly polynomial time determine the optimal interval I. This proves claim (i), and thus concludes the proof of Lemma 2.9. □

2.4 Solving base block problems

In this section, we discuss how to solve CCTU problems—and thus also CCTUF and R-CCTUF problems—whose constraint matrices are base block matrices, i.e., matrices falling into case (i) or (ii) of Theorem 2.14. Note that we can always assume to start with a CCTU problem whose relaxation is feasible, which we can check in strongly polynomial time; for otherwise, the CCTU problem is clearly infeasible. Hence we assume feasibility of the relaxation throughout this section. To start with, let us recall the definition of a *network matrix*.

Definition 2.31. *A matrix T is a* network matrix *if the rows of T can be indexed by the edges of a directed spanning tree (V, U), and the columns can be indexed by the edges of a directed graph (V, A) on the same vertex set, such that for every arc $a = (v, w) \in A$ and every arc $u \in U$,*

$$T_{u,a} = \begin{cases} 1 & \text{if the unique } v\text{-}w \text{ path in } U \text{ passes through } u \text{ forwardly,} \\ 0 & \text{if the unique } v\text{-}w \text{ path in } U \text{ does not pass through } u, \\ -1 & \text{if the unique } v\text{-}w \text{ path in } U \text{ passes through } u \text{ backwardly.} \end{cases}$$

Note that here, a directed graph is called a spanning tree if it is a spanning tree when ignoring edge directions. Moreover, we remark that we allow graphs to have several parallel edges connecting the same two vertices. In particular, the graph (V, A) in the above definition may have parallel edges, which correspond to identical columns of T. An important fact for our purposes is the following.

Lemma 2.32 (see, for example, [Sch98])**.** *Given a matrix T, one can efficiently recognize whether it is a network matrix. If so, a directed graph (V, A) and a directed tree (V, U) as in Definition 2.31 can be found efficiently.*

In the subsequent three sections, we distinguish three cases, namely whether the constraint matrix T of the CCTU problem that we consider is a network matrix, the transpose of a network matrix, or a matrix stemming from the constant-size matrices given in case (ii) of Theorem 2.14. As indicated above, we show in each case that the corresponding CCTU problem can be solved efficiently under some assumptions, thus implying Theorem 2.15, which covers the corresponding feasibility problems.

In the case of network matrices and their transposes, we perform reductions to combinatorial problems. In this context, it is convenient to transform the CCTU problems into a more structured class of CCTU problems, which we call *normalized CCTU* problems and are defined as follows.

Definition 2.33 (Normalized CCTU problem)**.** *A problem of the form*

$$\min \left\{ c^\top x \mid Tx \leq b,\ \gamma^\top x \equiv r \pmod{m}, x \in \mathbb{Z}^n_{\geq 0} \right\} \tag{2.12}$$

fulfilling that the origin is an optimal solution to the relaxation of (2.12)*, is called a* normalized CCTU *problem.*

Note that the right-hand side b of a normalized CCTU problem is non-negative because the origin is feasible. As we briefly discuss in the following, it is not hard to see that one can assume to deal with normalized CCTU problems, as formalized in the following observation.

Observation 2.34. *Every CCTU problem can be reduced in strongly polynomial time to a normalized CCTU problem. Furthermore, if the constraint matrix of the first problem is a base block matrix, the constraint matrix of the latter problem is a base block matrix of the same type.*

Proof. Indeed, consider an arbitrary CCTU problem (with feasible relaxation)

$$\min \left\{ c^\top x \;\middle|\; Tx \leq b,\ \gamma^\top x \equiv r \pmod m,\ x \in \mathbb{Z}^n \right\} \quad . \tag{2.13}$$

An equivalent CCTU problem where the origin is an optimal solution to its relaxation can simply be obtained by a standard shifting argument. To this end, assume first that the relaxation has a finite optimal solution. In this case we compute such a finite optimal solution x_0, and then substitute x by $x' + x_0$ to obtain the equivalent CCTU problem

$$\min \left\{ c^\top x' \;\middle|\; Tx' \leq b',\ \gamma^\top x' \equiv r' \pmod m,\ x' \in \mathbb{Z}^n \right\} \quad ,$$

where $b' = b - Tx_0$ and $r' = r - \gamma^\top x_0$. Clearly, the origin is an optimal solution to this transformed problem. In case the relaxation is unbounded, we know by Lemma 2.72 that (2.13) is either infeasible or unbounded. Hence, it is unbounded if and only if it is feasible. Moreover, Lemma 2.72 allows for obtaining efficiently a description of a set of unbounded solutions from any solution to (2.13). Hence, in this case, the optimization problem for (2.13) is equivalent to its feasibility version, and we can therefore replace the objective c by an all-zeros objective. This brings us back to the first case where the relaxation has a finite optimum.

Furthermore, to reduce to non-negative variables we can use another standard transformation that replaces every variable $x \in \mathbb{Z}$ by the difference $x^+ - x^-$ of two non-negative variables $x^+, x^- \in \mathbb{Z}_{\geq 0}$. Notably, these substitutions change the constraint matrix, but it can be observed that base block matrices remain base block matrices of the same type.[13] Applying this reduction on top of the previous one, we maintain that the origin is an optimal solution to the relaxation, thus obtaining Observation 2.34. □

Moreover, note that by our proximity result, Theorem 2.5, we obtain that a normalized CCTU problem has an optimal solution x^* with $\|x^*\|_\infty \leq m - 1$. Due to the non-negativity of the variables in a normalized CCTU problem, we thus obtain that there is an optimal solution x^* with $x^*_i \in \{0, \dots, m-1\}$ for each entry $i \in [n]$. This is a property we repeatedly exploit in our reductions developed in the following.

2.4.1 Network matrices

In this section, we show that CCTU problems with unary encoded objectives and constraint matrices that are network matrices can be solved efficiently using a randomized algorithm.

Theorem 2.35. *There is a strongly polynomial time randomized algorithm to solve CCTU problems with unary encoded objectives, constant modulus and constraint matrices that are network matrices.*

Our approach in this case is to exploit the graph structure that comes with network matrices to interpret CCTU problems (or, more precisely, normalized CCTU problems) with network constraint matrices as minimum-cost congruency-constrained circulation problems in certain directed graphs. To get started, let us recall that a circulation f in a directed graph $G = (V, A)$ with capacities $u \colon A \to \mathbb{Z}_{\geq 0}$ is a mapping $f \colon A \to \mathbb{Z}_{\geq 0}$ such that $f(a) \leq u(a)$ for every arc $a \in A$, and $f(\delta^+(v)) = f(\delta^-(v))$ for every vertex $v \in V$. Given arc lengths $\ell \colon A \to \mathbb{Z}$, the length of a circulation f is $\ell(f) \coloneqq \sum_{a \in A} \ell(a) f(a)$. Note that here, arc lengths are allowed to be negative.

A congruency-constrained circulation problem is formally defined as follows.

[13] Indeed, if we start with a constraint matrix T, this transformation to non-negative variables will lead to constraints described by the constraint matrix $[T \ -T]$ together with non-negativity constraints. Moreover, each of the base block matrix types is closed under copying columns, changing the signs of columns, and adding rows with a single non-zero entry.

Congruency-Constrained Circulation (CCC): Let $G = (V, A)$ be a directed graph with capacities $u\colon A \to \mathbb{Z}_{\geq 0}$, arc lengths $\ell\colon A \to \mathbb{Z}$, and let $\eta\colon A \to \mathbb{Z}$, $r \in \mathbb{Z}$, and $m \in \mathbb{Z}_{>0}$. Find a minimum-length circulation $f\colon A \to \mathbb{Z}_{\geq 0}$ in the given network such that $\sum_{a \in A} \eta(a) f(a) \equiv r \pmod{m}$.

The lemma below reduces CCTU problems with constraint matrices that are network matrices to CCC problems.

Lemma 2.36. *CCTU problems with modulus m and constraint matrices that are network matrices can be reduced in strongly polynomial time to CCC problems with modulus m and capacities within $\{0, \dots, m-1\}$.*

Proof. First of all, we know by Observation 2.34 that any CCTU problem with a constraint matrix that is a network matrix can be efficiently reduced to a normalized CCTU problem with a constraint matrix of the same type. Thus, assume we are given a normalized problem of the form

$$\min \left\{ c^\top x \;\middle|\; Tx \leq b,\ \gamma^\top x \equiv r \pmod{m},\ x \in \mathbb{Z}^n_{\geq 0} \right\}$$

with a network matrix T. By Theorem 2.11, we have that there is an optimal solution x to the above problem with $|d^\top x| \leq m - 1$ for all d that are TU-appendable to T.

We now define a CCC problem to which the above CCTU problem reduces. To this end, let (V, U) be the directed tree whose edges index the rows of the network matrix T, and let (V, E) be the digraph whose edges index the columns of T, as described in Definition 2.31. Let G be the directed graph with vertex set V and edge set $A \coloneqq U \cup \overleftarrow{U} \cup \overleftarrow{E}$, where $\overleftarrow{U} \coloneqq \{(w, v)\colon (v, w) \in U\}$ and analogously $\overleftarrow{E} \coloneqq \{(w, v)\colon (v, w) \in E\}$. Moreover, for an arc $u = (v, w)$, denote by $\overleftarrow{u} = (w, v)$ the corresponding reverse arc. We define the capacities $u\colon A \to \mathbb{Z}_{\geq 0}$, lengths $\ell\colon A \to \mathbb{Z}$, and values $\eta\colon A \to \mathbb{Z}$ of the CCC problem as follows. For all $a \in A$,

$$u(a) \coloneqq \begin{cases} \min\{b_a, m-1\} & \text{if } a \in U \\ m - 1 & \text{if } a \in \overleftarrow{U} \cup \overleftarrow{E} \end{cases},$$

$$\ell(a) \coloneqq \begin{cases} c_{\overleftarrow{a}} & \text{if } a \in \overleftarrow{E} \\ 0 & \text{if } a \in U \cup \overleftarrow{U} \end{cases}, \quad \text{and}$$

$$\eta(a) \coloneqq \begin{cases} \gamma(\overleftarrow{a}) & \text{if } a \in \overleftarrow{E} \\ 0 & \text{if } a \in U \cup \overleftarrow{U} \end{cases}.$$

Moreover, the modulus and target residue of the CCC problem are the same as of the CCTU problem, i.e., m and r, respectively. This concludes the definition of the CCC problem to which we reduce.

Finally, the desired statement follows directly from the following claim, which relates solutions of the CCTU problem to feasible circulations of the above-defined CCC problem.

Claim 2.37. *There is a solution of the CCC problem of length no larger than the optimal value of the CCTU problem. Conversely, given a circulation f for the CCC problem, one can compute in strongly polynomial time a solution x of the CCTU problem with $c^\top x = \ell(f)$.*

To see the forward direction of the claim, we start with an optimal solution x to the CCTU problem. By Theorem 2.11, we can assume that $|d^\top x| \leq m - 1$ for all d that are TU-appendable to T. In particular, this implies $x \in \{0, \dots, m-1\}^E$.

We now start by defining a circulation $g\colon A \to \mathbb{Z}_{\geq 0}$ (that may violate the capacity constraints given by u) by

$$g \coloneqq \sum_{e \in E} x(e) \left(\chi^{\overleftarrow{e}} + \chi^{P_e} \right), \tag{2.14}$$

where, for every $e = (v, w) \in E$, the set $P_e \subseteq U \cup \overleftarrow{U}$ is the unique path from v to w in $U \cup \overleftarrow{U}$ that has all edges directed from v to w. Finally, the circulation $f\colon A \to \mathbb{Z}_{\geq 0}$ that corresponds to x is obtained from g by cancelling out flows on arcs in opposite directions. Formally, we set

$$f(a) \coloneqq \begin{cases} g(a) & \text{if } a \in E \ , \\ g(a) - \min\{g(a), g(\overleftarrow{a})\} & \text{if } a \in U \cup \overleftarrow{U} \ . \end{cases}$$

Hence, one can interpret f as being obtained from g by cancelling flow on 2-cycles. By the definition of the lengths ℓ, one immediately obtains $\ell(f) = c^\top x$ as desired. Moreover, because x is integral, we have that g is integral and therefore also f. Also, $\sum_{a \in A} \eta(a) f(a) = \gamma^T x \equiv r \pmod{m}$. It remains to observe that f is a circulation, i.e., each vertex has the same in-flow as out-flow with respect to f and f fulfills the capacity constraints given by u.

Note that each vertex has the same in- and out-flow with respect to g, because every term in (2.14) corresponds to sending a flow of $x(e)$ along the cycle $\overleftarrow{e} \cup P_e$. Because f is obtained from g by cancelling flow on 2-cycles, also f has the same in- and out-flow at every vertex.

It remains to verify that the capacities given by u are respected by f. The capacities of arcs $a \in \overleftarrow{E}$, which are $u(a) = m - 1$, are fulfilled by f because $x(e) \leq m - 1$. Consider now an arc $a \in U$ and denote by $C_a \subseteq V$ the unique cut in (V, U) that satisfies $\delta^+(C_a) = \{a\}$ and $\delta^-(C_a) = \emptyset$. Such a cut exists as (V, U) is a tree. Because f is a circulation, we have

$$\begin{aligned} 0 = f(\delta^+(C_a)) - f(\delta^-(C_a)) = f(a) - f(\overleftarrow{a}) + f(\delta^+(C_a) \cap \overleftarrow{E}) - f(\delta^-(C_a) \cap \overleftarrow{E}) \\ \iff \qquad f(a) - f(\overleftarrow{a}) = \sum_{e \in E\colon a \in P_e} x(e) - \sum_{e \in E\colon \overleftarrow{a} \in P_e} x(e) \ . \end{aligned} \tag{2.15}$$

Observe that the difference of the last two sums is precisely $d^\top x$, where d is the row vector of T indexed by u. Because $d^\top x \leq b_a$ is a constraint of the original normalized CCTU problem, we have $f(a) - f(\overleftarrow{a}) = d^\top x \leq b_a$. Moreover, because both $d^\top$ and $-d^\top$ are TU-appendable to the constraint matrix T, we obtain by Theorem 2.11 that $-m + 1 \leq f(a) - f(\overleftarrow{a}) \leq m - 1$. Hence,

$$-(m-1) \leq f(a) - f(\overleftarrow{a}) \leq \min\{b_a, m-1\} \ .$$

The above inequality implies $f(a) \leq \min\{b_a, m-1\} - f(\overleftarrow{a}) \leq \min\{b_a, m-1\} = u(a)$. Moreover, $f(\overleftarrow{a}) \leq m - 1 + f(a) = u(\overleftarrow{a}) + f(a)$. Note that because f has by definition a value of zero on either a or $\overleftarrow{a}$, this implies that either $f(\overleftarrow{a}) = 0 \leq m - 1 = u(\overleftarrow{a})$, if $f(\overleftarrow{a}) = 0$, or $f(\overleftarrow{a}) \leq m - 1 + f(a) = m - 1 = u(\overleftarrow{a})$, if $f(a) = 0$. In any case, we have $f(\overleftarrow{a}) \leq u(\overleftarrow{a})$. Thus, f also fulfills the capacity constraints for all arcs in $U \cup \overleftarrow{U}$.

For the backward direction of Claim 2.37, assume that we are given an integral circulation f in G respecting the capacity constraints u, and define $x(e) \coloneqq f(\overleftarrow{e})$ for all $e \in E$. Note that we thus obtain x in strongly polynomial time. Again, (2.15) holds and the right-hand side is $d^\top x$, where d is the row indexed by a in T. Non-negativity of f and the capacity constraints then imply for all $a \in A$ that

$$d^\top x = f(a) - f(\overleftarrow{a}) \leq f(a) \leq b_a \ .$$

Hence, x satisfies all constraints $Tx \leq b$ and is non-negative due to non-negativity of f. Moreover, we again have

$$\gamma^\top x = \sum_{e \in E} \gamma(e) x(e) = \sum_{e \in \overleftarrow{E}} \eta(\overleftarrow{e}) f(\overleftarrow{e}) = \sum_{a \in A} \eta(a) f(a) \equiv r \pmod{m} \ .$$

Hence, the vector x is feasible for the CCTU problem. This proves the claim, which in turn implies the statement of Lemma 2.36, as desired. □

We remark that for modulus $m = 2$, an analogous reduction to the one we used in the proof of Lemma 2.36 was already done in [AWZ17]. Our reduction is a generalization of that one. For the special case with modulus $m = 2$, the resulting CCC problems are non-trivial only if $r = 1$, i.e., when the goal is to find an odd circulation. This can easily be reduced to finding a shortest odd cycle in a suitable auxiliary graph, which can be solved via standard techniques. For general m, however, the solution structure can be significantly more complex. We observe and exploit a connection to the so-called *exact length circulation problem*, where the goal is to find a circulation whose length is equal to a given value.

Exact Length Circulation (XLC): Let $G = (V, A)$ be a digraph with capacities $u\colon A \to \mathbb{Z}_{>0}$ and arc lengths $\ell\colon A \to \mathbb{Z}$. Given $L \in \mathbb{Z}$, find a circulation f in the given network such that $\ell(f) = L$.

Exact length circulation problems can be solved using a randomized pseudopolynomial algorithm, as shown by Camerini, Galbiati, and Maffioli [CGM92]. They reduce the problem to an exact cost perfect matching problem, which can then be reduced to computing the coefficients of a well-defined polynomial. The following theorem summarizes the result of Camerini, Galbiati, and Maffioli [CGM92] for XLC.

Theorem 2.38 ([CGM92])**.** *There is a randomized algorithm to solve XLC problems in a directed graph $G = (V, E)$ with capacities $u\colon A \to \mathbb{Z}_{\geq 0}$ in time* $\mathrm{poly}(|V|, \max_{a\in A} u(a), \max_{a\in A} |\ell(a)|)$.

Thus, it remains to build the connection between CCC and XLC problems. We achieve this by integrating the contributions $\eta(a)$ of every arc towards the congruency constraint into its length, and searching for the minimum length of a suitable circulation using binary search, thereby obtaining the following lemma.

Lemma 2.39. *Every CCC problem in a graph $G = (V, A)$ with modulus m, arc lengths $\ell\colon A \to \mathbb{Z}$, and capacities $u\colon A \to \{0, 1, \ldots, m-1\}$ can be polynomially reduced to* $\mathrm{poly}(m, |V|, |A|, \max_{a\in A} |\ell(a)|)$ *many XLC problems in G with the same capacities.*

Proof. Note that in any CCC problem, we may assume without loss of generality that $\eta(a) \in \{0, \ldots, m-1\}$ by reducing the values modulo m. Now, for every arc a in a given CCC problem, define a new length function $\widetilde{\ell}(a) = \ell(a) \cdot m^2|A| + \eta(a)$. We thus have $\widetilde{\ell}(f) = \ell(f) \cdot m^2|A| + \sum_{a\in A} \eta(a)f(a)$, and because $\sum_{a\in A} \eta(a)f(a) < m^2|A|$, we can retrieve both $\ell(f)$ and $\sum_{a\in A} \eta(a)f(a)$ from $\widetilde{c}(f)$. Consequently, finding a circulation of length L with $\sum_{a\in A} \eta(a)f(a) \equiv r \pmod m$ is equivalent to solving XLC problems in G with respect to lengths $\widetilde{\ell}$ and with target length $\widetilde{L} = L \cdot m^2|A| + km + r$ for all $k \in \{0, \ldots, m|A| - 1\}$. We can find the smallest L for which there is a CCC solution of length L by binary search in $O(\log(m|A| \cdot \max_{a\in A} |\ell(a)|))$ iterations, because $|\ell(f)| = \left|\sum_{a\in A} \ell(a)f(a)\right| \leq m|A| \cdot \max_{a\in A} |\ell(a)|$. Altogether, this gives the desired result. □

Combining the above findings, we conclude this section with a proof of Theorem 2.35.

Proof of Theorem 2.35. By Lemma 2.36, a CCTU problem whose constraint matrix is a network matrix can be reduced in strongly polynomial time to a CCC problem with $u(a) \leq m - 1$ for all $a \in A$. By Lemma 2.39, this problem further reduces to $\mathrm{poly}(m, |V|, |A|, \max_{a\in A} |c(a)|)$ many XLC problems, where each of them can be solved in $\mathrm{poly}(|V|, \max_{a\in A} u(a), \max_{a\in A} |c(a)|) = \mathrm{poly}(|V|, m, \max_{a\in A} |c(a)|)$ time using a randomized algorithm. Thus, overall, we obtain that there is a randomized algorithm to solve a CCTU problem whose constraint matrix is a network matrix in time $\mathrm{poly}(m, |V|, |A|, \max_{a\in A} |c(a)|)$, i.e., a strongly polynomial algorithm if the objective c is given in unary encoding and m is a constant. □

2.4.2 Transposes of network matrices

The purpose of this section is to prove the following theorem.

Theorem 2.40. *There is a strongly polynomial time algorithm to solve CCTU problems with constant prime power modulus and constraint matrices that are transposed network matrices.*

To achieve this result, we again exploit the graph structure coming with network matrices. This time, we reduce CCTU problems (or, more precisely and equivalently, normalized CCTU problems) to certain directed cut problems of the following form.

Constrained Tree Cuts (CTC): Let $T = (V, U)$ be a directed tree, $A \subseteq V \times V$ and $b\colon A \to \mathbb{Z}_{\geq 0}$. Let $c\colon U \to \mathbb{Z}$ be arc costs, $\alpha\colon V \to \mathbb{Z}$, $r \in \mathbb{Z}$, and $m \in \mathbb{Z}_{>0}$. Find a family of sets $S_1, \dots, S_\ell \subseteq V$ minimizing the total cost $\sum_{i=1}^{\ell} c(\delta^+(S_i))$ such that

(i) $\delta^-(S_i) = \emptyset$ for all $i \in [\ell]$,
(ii) $|\{i \in [\ell]\colon v \in S_i\}| - |\{i \in [\ell]\colon w \in S_i\}| \leq b_a$ for all $a = (v, w) \in A$, and
(iii) $\sum_{i=1}^{\ell} \alpha(S_i) \equiv r \pmod m$, where $\alpha(S_i) \coloneqq \sum_{v \in S_i} \alpha(v)$.

We highlight that in CTC problems, the number $\ell \in \mathbb{Z}_{\geq 0}$ of sets that are returned is not fixed upfront; in the extreme case, we might even return an empty family, i.e., use $\ell = 0$. Moreover, we also allow the sets S_i to be empty or equal to V, opposed to the typical setting in cut problems where this is usually excluded. CTC problems inherit many structural properties from CCTU problems, including structural results on optimal solutions. These will allow us to further reduce CTC problems to directed congruency-constrained minimum cut problems, for which efficient algorithms are known for the case of the modulus m being a constant prime power [NSZ19]. In CTC problems, we call the constraint (iii) the *congruency constraint*, and we refer to the problem obtained after dropping that constraint as the *relaxation* of the CTC problem.

We start by showing the reduction from normalized CCTU problems to CTC problems. More concretely, to every normalized CCTU problem $\min\{c^\top x\colon Tx \leq b,\ \gamma^\top x \equiv r \pmod m,\ x \in \mathbb{Z}^n_{\geq 0}\}$ with T being the transpose of a network matrix and such that T does not contain identical rows (otherwise, one row of the identical rows corresponds to a redundant constraint and can be deleted), we associate the following CTC problem: The tree (V, U) and the extra arc set $A \subseteq V \times V$ are those coming with the network constraint matrix through Definition 2.31, $b\colon A \to \mathbb{Z}_{\geq 0}$ is the right-hand side vector of the CCTU problem (which is non-negative because we assume the CCTU problem to be normalized), $\alpha\colon V \to \mathbb{Z}$ is defined by $\alpha(v) \coloneqq \gamma(\delta^+(v)) - \gamma(\delta^-(v))$ for all $v \in V$, and costs c as well as r and m are left unchanged.[14] To relate feasible solutions of CCTU problems and the associated CTC problem, we prove the following result.

Lemma 2.41. *Consider a normalized CCTU problem whose constraint matrix has no identical rows and is the transpose of a network matrix, and the associated CTC problem. Let $S_1, \dots, S_\ell \subseteq V$ with $\delta^-(S_i) = \emptyset$ for all $i \in [\ell]$, and define $x = \sum_{i=1}^{\ell} \chi^{\delta^+(S_i)}$. Then x is a feasible CCTU solution if and only if $S_1, \dots, S_\ell$ is a feasible CTC solution. Moreover, if both are feasible, their objective value is the same.*

The main ingredient in Lemma 2.41 is to relate inequality constraints of the CCTU problem and the constraints (ii) in the associated CTC problems. We use this relation again later, and hence, state it independently here before using it to prove Lemma 2.41.

Lemma 2.42. *Let (V, U) be a directed spanning tree, let $S_1, \dots, S_\ell \subseteq V$ with $\delta^-(S_i) = \emptyset$ for all $i \in [\ell]$, and denote $x = \sum_{i=1}^{\ell} \chi^{\delta^+(S_i)}$. Then for any $v, w \in V$, the vector $t_{vw} \in \{-1, 0, 1\}^U$ defined by*

$$\forall u \in U\colon \quad t_{vw}(u) = \begin{cases} 1 & \text{if the unique } v\text{-}w \text{ path in } U \text{ passes through } u \text{ forwardly,} \\ 0 & \text{if the unique } v\text{-}w \text{ path in } U \text{ does not pass through } u, \\ -1 & \text{if the unique } v\text{-}w \text{ path in } U \text{ passes through } u \text{ backwardly} \end{cases}$$

satisfies $t_{vw}^\top x = |\{i \in [\ell]\colon v \in S_i\}| - |\{i \in [\ell]\colon w \in S_i\}|$.

[14] Assuming that T does not contain identical rows implies that no parallel arcs are needed in A, which justifies the assumption $A \subseteq V \times V$.

Proof. By definition of x, we have that

$$t_{vw}^\top x = \sum_{i=1}^{\ell} \sum_{u \in \delta^+(S_i)} t_{vw}(u) \ .$$

For fixed $i \in [\ell]$ and by definition of t_{vw}, the non-zero terms in the inner sum correspond to edges u that are oriented from a vertex outside S_i to a vertex inside S_i, and that lie on the unique v-w path P in U. Recall that $\delta^-(S_i) = \emptyset$, hence the sum in fact has one non-zero term for every time the path P crosses from one side of S_i to the other. More precisely, there is a term $+1$ for every time the path P crosses from a vertex inside S_i to one outside S_i, and a term -1 for every time the path P crosses from a vertex outside S_i to one inside S_i. Consequently, the total value of the sum only depends on where the start- and endpoints v and w are located with respect to S_i: If $v \in S_i$ and $w \notin S_i$, for example, P will cross from a vertex inside S_i to one outside S_i one more time than the other way round, hence the sum will be $+1$. Generally, we get that $\sum_{u \in \delta^+(S_i)} t_{vw}(u) = 1_{v \in S_i} - 1_{w \in S_i}$, and thus

$$t_{vw}^\top x = \sum_{i=1}^{\ell} \left(1_{v \in S_i} - 1_{w \in S_i}\right) = \sum_{i=1}^{\ell} 1_{v \in S_i} - \sum_{i=1}^{\ell} 1_{w \in S_i} = |\{i \in [\ell] \colon v \in S_i\}| - |\{i \in [\ell] \colon w \in S_i\}| \ . \quad \square$$

Proof of Lemma 2.41. We start by showing that x is feasible for the inequality system $Tx \leq b$ of the CCTU problem if and only if $S_1, \dots, S_\ell$ is feasible for constraint (ii) of the CTC problem. To this end, consider a row of the constraint matrix T that is indexed by the arc $a = (v, w) \in A \times A$, and note that this row is precisely the vector $t_{vw}^\top$, with t_{vw} as defined in Lemma 2.42. Consequently, the corresponding constraint $t_{vw}^\top x \leq b_a$ of the CCTU problem is, by Lemma 2.42, equivalent to $|\{i \in [\ell] \colon v \in S_i\}| - |\{i \in [\ell] \colon w \in S_i\}| \leq b_a$, which is one of the constraints in (ii) in the CTC problem (namely the one for the arc $a = (v, w) \in A$). Thus, we conclude that $Tx \leq b$ is equivalent to constraint (ii) in the CTC problem. Next, we observe that

$$\begin{aligned} \sum_{i=1}^{\ell} \alpha(S_i) &= \sum_{i=1}^{\ell} \sum_{v \in S_i} \left(\gamma(\delta^+(v)) - \gamma(\delta^-(v))\right) \\ &= \sum_{i=1}^{\ell} \left(\gamma(\delta^+(S_i)) - \gamma(\delta^-(S_i))\right) = \sum_{i=1}^{\ell} \gamma^\top \chi^{\delta^+(S_i)} = \gamma^\top x \ , \end{aligned}$$

and hence $\sum_{i=1}^{\ell} \alpha(S_i) \equiv r \pmod m$ if and only if $\gamma^\top x \equiv r \pmod m$. Together, we obtain that x is a feasible CCTU solution if and only if $S_1, \dots, S_\ell$ is a feasible CTC solution. To finish the proof of the lemma, we observe that the objectives of the CCTU solution x and the CTC solution $S_1, \dots, S_\ell$ are equal because $c^\top x = \sum_{i=1}^{\ell} c^\top \chi^{\delta^+(S_i)} = \sum_{i=1}^{\ell} c(\delta^+(S_i))$. $\square$

By showing that for any feasible CCTU solution x, there exist sets $S_1, \dots, S_\ell \subseteq V$ with $\delta^-(S_i) = \emptyset$ and $x = \sum_{i=1}^{\ell} \chi^{\delta^+(S_i)}$, and combining this with Lemma 2.41, we thus obtain the following.

Lemma 2.43. *Consider a normalized CCTU problem whose constraint matrix has no identical rows and is the transpose of a network matrix, and the associated CTC problem as constructed above.*

(i) For every feasible solution x of the CCTU problem, there is a feasible solution $S_1, \dots, S_\ell$ of the CTC problem with the same objective value such that $x = \sum_{i=1}^{\ell} \chi^{\delta^+(S_i)}$.

(ii) For every optimal solution x of the CCTU problem, there is an optimal solution $S_1, \dots, S_\ell$ of the CTC problem such that $x = \sum_{i=1}^{\ell} \chi^{\delta^+(S_i)}$.

Proof. (i) Note that because (V, U) is a tree, for every $u \in U$, there is a unique cut $C_u \subseteq V$ with $\delta^+(C_u) = \{u\}$ and $\delta^-(C_u) = \emptyset$. By definition, we have $x = \sum_{u \in U} x(u) \chi^{\delta^+(C_u)}$. Consequently, by Lemma 2.41, the collection consisting of $x(u)$ times the set C_u for all $u \in U$ is a feasible CTC solution, and its objective value is the same as the objective value of x in the CCTU problem.

(ii) By part (i), it is enough to prove that the associated CTC problem does not have solutions with objective value less than the value $c^\top x$ of x. If there was such a CTC solution, say $S'_1, \dots, S'_{\ell'}$, of value strictly less than $c^\top x$, then by Lemma 2.41, we know that $x' = \sum_{i=1}^{\ell'} \chi^{\delta^+(S'_i)}$ is a feasible CCTU solution of the same objective value—but this is a contradiction, since we assumed x to be optimal for the CCTU problem. □

In other words, the above immediately implies that CCTU problems can be reduced to CTC problems.

Corollary 2.44. *Every normalized CCTU problem whose constraint matrix has no identical rows and is the transpose of a network matrix can be strongly polynomially reduced to the associated CTC problem (i.e., the CTC problem can be obtained in strongly polynomial time), and any optimal CTC solution can in strongly polynomial time be transformed to an optimal CCTU solution.*

Proof. The CTC problem associated to a CCTU problem can be constructed in strongly polynomial time, in particular because from the constraint matrix T, the tree $T = (V, U)$ and the extra arcs $A \subseteq V \times V$ can be obtained in polynomial time (in the encoding size of T) through Lemma 2.32. Lemma 2.43 (ii) shows that optimal solutions of the CCTU problem and the CTC problem have the same values. Moreover, by Lemma 2.41, any solution $S_1, \dots, S_\ell$ of the CTC problem immediately gives a feasible solution $x = \sum_{i=1}^{\ell} \chi^{\delta^+(S_i)}$ of the CCTU problem with the same value (and note that x can be computed in strongly polynomial time). Thus if $S_1, \dots, S_\ell$ is optimal for the CTC problem, then so is x for the CCTU problem. □

We remark that the above reduction gives CTC instances with $\alpha(V) = 0$. It turns out that because the underlying graph (V, U) is a tree, this condition is enough to uniquely determine corresponding values $\gamma\colon U \to \mathbb{Z}$ such that $\alpha(v) = \gamma(\delta^+(v)) - \gamma(\delta^-(v))$ for all $v \in V$, which allows us to also reduce CTC problems to CCTU problems in that case. As for our purposes, the direction covered by Corollary 2.44 is enough, we leave the details of this argument to the reader. To be able to exploit the reduction given in Corollary 2.44, we continue with studying the structure of CTC solutions in more detail, with the goal to identify patterns that help for finding optimal CTC solutions efficiently.

Lemma 2.45. *Consider a CTC problem and let $S_1, \dots, S_\ell$ be a feasible solution. Then there exists a feasible solution $T_1, \dots, T_\ell$ such that $T_\ell \subseteq T_{\ell-1} \subseteq \dots \subseteq T_1$ and $\sum_{i=1}^{\ell} \chi^{\delta^+(S_i)} = \sum_{i=1}^{\ell} \chi^{\delta^+(T_i)}$.*

Proof. If for all $j, k \in [\ell]$, we have $S_j \subseteq S_k$ or $S_k \subseteq S_j$, there is nothing to prove, because relabeling the sets to satisfy $S_\ell \subseteq S_{\ell-1} \subseteq \dots \subseteq S_1$ will give the desired solution. Thus, assume that there are two sets S_j and S_k for $j, k \in [\ell]$ such that $S_j \not\subseteq S_k$ and $S_k \not\subseteq S_j$. We claim that removing the sets S_j, S_k from the solution and adding the sets in $S_j \cup S_k$ and $S_j \cap S_k$ instead gives another feasible solution for the CTC problem such that the sum $\sum_{i=1}^{\ell} \chi^{\delta^+(S_i)}$ is unchanged. To see this, observe the following:

- $\delta^-(S_j \cup S_k) = \delta^-(S_j \cap S_k) = \emptyset$ because an arc entering the union or intersection of the two sets would enter at least one of the sets, but we know that $\delta^-(S_j) = \delta^-(S_k) = \emptyset$. Thus, $\delta^-(S_i) = \emptyset$ holds for all S_i in the new solution.
- For any vertex $v \in V$, the number of sets in the solution that contain v is invariant under replacing two sets with their union and intersection, hence the left-hand side of any constraint in condition (ii) of CTC problems remains the same, and thus the constraints in condition (ii) of CTC problems holds for the new solution, as well.
- We have $\alpha(S_j) + \alpha(S_k) = \alpha(S_j \cup S_k) + \alpha(S_j \cap S_k)$, so the congruency-constraint is fulfilled by the new solution if and only if the initial solution fulfilled it.
- Finally, it generally holds that

$$\chi^{\delta^+(S_j)} + \chi^{\delta^+(S_k)} = \chi^{\delta^+(S_j \cup S_k)} + \chi^{\delta^+(S_j \cap S_k)} + \chi^{U(S_j \setminus S_k, S_k \setminus S_j)} + \chi^{U(S_k \setminus S_j, S_j \setminus S_k)} \quad ,$$

  where, for vertex sets $V_1, V_2 \subseteq V$, we denote by $U(V_1, V_2) \subseteq U$ all arcs of U with tail in V_1 and head in V_2. Because $\delta^-(S_j) = \delta^-(S_k) = \emptyset$, we have $U(S_j \setminus S_k, S_k \setminus S_j) = U(S_k \setminus S_j, S_j \setminus S_k) = \emptyset$, which

implies that the last two terms of the right-hand side above are zero. Consequently, $\chi^{\delta^+(S_j)}+\chi^{\delta^+(S_k)} = \chi^{\delta^+(S_j\cup S_k)} + \chi^{\delta^+(S_j\cap S_k)}$, and thus the sum $\sum_{i=1}^{\ell} \chi^{\delta^+(S_i)}$ is unchanged under the replacement step, as well.

Thus, as long as there are two sets S_j and S_k such that $S_j \not\subseteq S_k$ and $S_k \not\subseteq S_j$, we can replace them by $S_j \cup S_k$ and $S_j \cap S_k$ while maintaining feasibility for the CTC problem and not changing the sum $\sum_{i=1}^{\ell} \chi^{\delta^+(S_i)}$. To see that this procedure ends, note that in any step, the potential function $\Phi(S_1, \dots, S_\ell) \coloneqq \sum_{i=1}^{\ell} |S_i|^2 \in \mathbb{Z}$ strictly increases. The latter follows from the fact that, for any two sets A and B, we always have $|A|^2 + |B|^2 < |A \cap B|^2 + |A \cup B|^2$. Obviously, $\Phi(S_1, \dots, S_\ell) \leq \ell|V|^2$, so the procedure terminates after less than $\ell|V|^2$ many steps with a solution that has the desired properties. □

In the next lemma, we prove that in CTC problems that are obtained via a reduction from CCTU problems, there even exist optimal solutions that consist of a chain $S_\ell \subseteq \dots \subseteq S_1$ with a bounded number of sets, namely $\ell \leq m-1$. This closely links back to our general proximity result, Theorem 2.11, from which we know that a normalized CCTU problem has an optimal solution x^* such that for any vector $d \in \mathbb{Z}^n$ that is TU-appendable to the constraint matrix T, we have $d^\top x^* \leq m-1$. In the proof of the following lemma, we show that the optimal CTC solution corresponding to such a CCTU solution x^* has the desired properties.

Lemma 2.46. *Consider a normalized CCTU problem with modulus m whose constraint matrix has no identical rows and is the transpose of a network matrix. Then, the associated CTC problem has an optimal solution $S_1, \dots, S_\ell$ such that $S_\ell \subseteq S_{\ell-1} \subseteq \dots \subseteq S_1$ and $\ell \leq m-1$.*

Proof. Let x^* be an optimal solution of the CCTU problem such that for every vector $d \in \mathbb{Z}^n$ that is TU-appendable to the constraint matrix T, we have $d^\top x^* \leq m-1$. Such a solution exists due to Theorem 2.11 because the CCTU problem is normalized, and hence $x_0 = (0 \;\; 0 \;\; \dots \;\; 0)^\top \in \mathbb{Z}^n$ is an optimal solution of its relaxation. By Lemma 2.43, there exists an optimal solution $S_1, \dots, S_\ell$ of the associated CTC problem such that $x^* = \sum_{i=1}^{\ell} \chi^{\delta^+(S_i)}$, and by Lemma 2.45, we may even choose the sets $S_i \subseteq V$ such that they form a chain, i.e., $S_\ell \subseteq S_{\ell-1} \subseteq \dots \subseteq S_1$. Moreover, we may assume that $S_i \neq \emptyset$ and $S_i \neq V$ for all i: Such sets could be removed from the solution family without affecting feasibility of the solution (the left-hand sides of constraints in point (ii) of CTC problems will remain the same, and because $\alpha(\emptyset) = 0$ and $\alpha(V) = \sum_{v\in V} \alpha(v) = \sum_{v\in V} \gamma(\delta^+(v)) - \gamma(\delta^-(v)) = 0$, the congruency constraint will still be satisfied, as well) and the objective value (which is the same because $\delta^+(V) = \delta^+(\emptyset) = \emptyset$, and thus $c(\delta^+(V)) = c(\delta^+(\emptyset)) = 0$).

We claim that with the above assumptions, we have $\ell \leq m-1$. To see this, choose $v \in S_1$ and $w \in V \setminus S_\ell$. Note that such v and w exist by the assumption that $S_i \neq \emptyset$ and $S_i \neq V$ for all $i \in [\ell]$. Let $t_{vw} \in \{-1, 0, 1\}^U$ be defined as in Lemma 2.42, where (V, U) is the directed tree indexing the columns of the constraint matrix T of the CCTU problem according to Definition 2.31. By definition, t_{vw} is TU-appendable to the matrix T, as we can add the arc (v, w) to the arc (multi-)set indexing the rows of T according to Definition 2.31 and thereby obtain that the matrix T with extra row t_{vw} is the transpose of a network matrix again, and hence TU. Consequently, by the choice of the optimal solution x^*, we have $t_{vw}^\top x^* \leq m-1$. On the other hand, Lemma 2.42 implies that

$$t_{vw}^\top x^* = |\{i \in [\ell] \colon v \in S_i\}| - |\{i \in [\ell] \colon w \in S_i\}| = \ell \;\;,$$

because by choice of v and w, all sets S_i contain v, but none of them contain w. Altogether, this gives $\ell \leq m-1$, as desired. □

Thus, by Lemma 2.45, it is enough to find an optimal solution of a CTC problem associated to a CCTU problem such that the sets in the solution form a chain of (at most) $m-1$ cuts. This bounded number of cuts allows for a reduction to submodular minimization problems with congruency constraints of the following type.

Congruency-Constrained Submodular Minimization (CCSM): Given a submodular function $f\colon \mathcal{L} \to \mathbb{Z}$ defined on a lattice $\mathcal{L} \subseteq 2^N$, $\gamma\colon N \to \mathbb{Z}$, $m \in \mathbb{Z}_{>0}$, and $r \in \{0, \dots, m-1\}$, find a minimizer of $\min\{f(C)\colon C \in \mathcal{L}, \gamma(C) \equiv r \pmod m\}$.

Such problems were studied by Nägele, Sudakov, and Zenklusen [NSZ19], where an algorithm for solving problems of this kind if m is a constant prime power modulus was presented. We remark that [NSZ19] studies a slightly less general setup than stated above, namely $\gamma \equiv 1$, where the constraint $\gamma(S) \equiv r \pmod m$ translates to $|S| \equiv r \pmod m$. For that case, algorithms with running time $|N|^{2m+O(1)}$ were presented. However, the setting with general γ can be readily reduced to that with $\gamma \equiv 1$ by replacing every element $v \in N$ by $t = (\gamma(v) \bmod m)$ many elements $v_1, \dots, v_t$ with $\gamma(v_i) = 1$ and updating the lattice and the function correspondingly. Observing that this reduction blows up the ground set by a factor of at most m, we thus get the following immediate generalization of Theorem 1.1 in [NSZ19].

Theorem 2.47. *For any prime power $m \in \mathbb{Z}_{>0}$, CCSM problems can be solved in time $(m|N|)^{2m+O(1)}$.*

It remains to discuss our reduction from CTC problems to CCSM problems.

Lemma 2.48. *Consider a CTC problem with constant modulus m. Finding a feasible solution of minimum cost among all solutions that consist of at most $m-1$ sets $S_1, \dots, S_\ell$ with $S_\ell \subseteq S_{\ell-1} \subseteq \dots \subseteq S_1$ can be strongly polynomially reduced to a CCSM problem with modulus m, i.e., the CCSM problem can be obtained in strongly polynomial time, and an optimal solution of that problem can be transformed to an optimal CTC solution in strongly polynomial time.*

Proof. Consider a CTC instance with the usual notation. We construct a CCSM instance on a ground set N with a lattice $\mathcal{L} \subseteq 2^N$ whose sets correspond to feasible solutions of the relaxation of the given CTC problem that have the desired chain structure. Moreover, we show that the function $f\colon \mathcal{L} \to \mathbb{Z}$ assigning to each set in $\mathcal{L}$ the value of the corresponding CTC solution is a modular function. The last step will then be to observe that we can define a congruency constraint of the type appearing in CCSM problems that is equivalent to the congruency constraint in the CTC problem.

Let the ground set N consist of $m-1$ copies of the vertex set V of the tree in the CTC instance, i.e., $N := \bigcup_{i=1}^{m-1} V_i$, where $V_i = \{v_i\colon v \in V\}$. Subsets $C \subseteq N$ are in one-to-one correspondence with set families $S_1, \dots, S_\ell \subseteq V$ that satisfy $\ell \leq m-1$ as follows: Given C, the corresponding set family is given by $S_i = \{v \in V\colon v_i \in C\}$, and vice versa, given a set family $S_1, \dots, S_\ell$ with $\ell \leq m-1$, the corresponding subset of N is $C = \bigcup_{i=1}^{\ell} \{v_i\colon v \in S_i\}$. Now let us define a set $\mathcal{L} \subseteq 2^N$ such that $C \subseteq N$ is in $\mathcal{L}$ if and only if all three of the following are satisfied:

(i) If $v_i \in C$ for some $v \in V$ and $i \in [m-1]$, then $v_j \in C$ for all $j \leq i$.
(ii) For every $(v, w) \in U$, if $w_i \in C$ for some $i \in [m-1]$, then $v_i \in C$.
(iii) For every $(v, w) \in A$, if $v_i \in C$ and $i - b_a \geq 1$, then $w_{i-b_a} \in C$.

Claim 2.49. *Sets $C \in \mathcal{L}$ are precisely those subsets of N that correspond to set families $S_1, \dots, S_\ell$ with $\ell \leq m-1$ that have chain structure $S_\ell \subseteq \dots \subseteq S_1$ and are feasible solutions of the relaxation of the given CTC problem.*

To see the claim, we start by observing that a set $C \subseteq N$ satisfies (i) if and only if the corresponding sets $S_1, \dots, S_\ell$ satisfy $S_i \subseteq S_j$ for all $i \geq j$: If C satisfies (i), then $v \in S_i$, we get $v_i \in C$, which implies $v_j \in C$ because $i \geq j$, and thus $v \in S_j$. For the other way round, if $S_i \subseteq S_j$ for all $i \geq j$, then if $v_i \in C$ for some $v \in V$ and $i \in [m-1]$, we have $v \in S_i$, and thus for all $i \geq j$, it follows that $v \in S_j$, and thus $v_j \in C$.

Next, (ii) is satisfied by $C \subseteq N$ if and only if the corresponding sets $S_1, \dots, S_\ell$ satisfy $\delta^-(S_i) = \emptyset$: C does not satisfy (ii) if and only if there exist $(v, w) \in U$ and $i \in [m-1]$ such that $w_i \in C$, but $v_i \notin C$. But the latter is equivalent to $w \in S_i$ but $v \notin S_i$, i.e., $\delta^-(S_i) \neq \emptyset$.

Finally, consider a set C satisfying (i) above. We show that the corresponding sets $S_1, \dots, S_\ell$ then satisfy constraint (ii) of CTC problems if and only if C also satisfies (iii) above. To start with, note that by the previous arguments, we know that because C satisfies (i), we have $S_\ell \subseteq \dots \subseteq S_1$. Consequently,

$$|\{i \in [\ell] \colon v \in S_i\}| = \max\{i \in [m-1] \colon v_i \in C\} \qquad \forall v \in V \ ,$$

hence a constraint of the form $|\{i \in [\ell] \colon v \in S_i\}| - |\{i \in [\ell] \colon w \in S_i\}| \leq b_a$ for some $a = (v, w) \in A$ is satisfied if and only if $\max\{i \in [m-1] \colon v_i \in C\} - \max\{i \in [m-1] \colon w_i \in C\} \leq b_a$, which in turn is guaranteed to hold if and only if C satisfies (iii) above, as desired. This proves Claim 2.49.

Claim 2.50. *$\mathcal{L}$ is a lattice.*

To prove this claim, we show that for any $C_1, C_2 \in \mathcal{L}$, we also have $C_1 \cap C_2 \in \mathcal{L}$ and $C_1 \cup C_2 \in \mathcal{L}$. We do so by showing that the intersection and union satisfy (i) to (iii) above. Note that all three conditions are of the form "If $a \in C$, then $b \in C$", for different choices of $a, b \in N$. It is generally true that if such conditions hold for two sets C_1 and C_2, then they also hold for $C_1 \cap C_2$ and $C_1 \cup C_2$: If $a \in C_1 \cap C_2$, then $a \in C_1$ and $a \in C_2$, hence also $b \in C_1$ and $b \in C_2$, and thus $b \in C_1 \cap C_2$. Also, if $a \in C_1 \cup C_2$, then there is $\varepsilon \in \{0, 1\}$ such that $a \in C_\varepsilon$, hence also $b \in C_\varepsilon$, and thus $b \in C_1 \cup C_2$. This proofs Claim 2.50.

As already indicated above, let $f \colon \mathcal{L} \to \mathbb{Z}$ be defined as follows: For $C \in \mathcal{L}$, if $S_1, \dots, S_\ell$ is the corresponding solution of the relaxation of the CTC problem, then $f(C) = \sum_{i=1}^{m-1} c(\delta^+(S_i))$. In other words, f assigns to each $C \in \mathcal{L}$ the objective value of the corresponding CTC solution.

We claim that for any two sets $C, D \in \mathcal{L}$, we have $f(C) + f(D) = f(C \cap D) + f(C \cup D)$. To this end, observe that if $S_1, \dots, S_\ell \subseteq V$ correspond to C and $T_1, \dots, T'_\ell \subseteq V$ correspond to D, we may introduce $S_{\ell+1} = \dots = S_{m-1} = \emptyset$ and $T_{\ell'+1} = \dots = T_{m-1} = \emptyset$ and then obtain

$$f(C)+f(D) = \sum_{i=1}^{m-1} c(\delta^+(S_i))+c(\delta^+(T_i)) = \sum_{i=1}^{m-1} c(\delta^+(S_i \cap T_i))+c(\delta^+(S_i \cup T_i)) = f(C \cap D)+f(C \cup D) \ ,$$

where the middle inequality exploits that $\chi^{\delta^+(S_i)} + \chi^{\delta^+(T_i)} = \chi^{\delta^+(S_i \cap T_i)} + \chi^{\delta^+(S_i \cup T_i)}$, which holds because $\delta^-(S_i) = \delta^-(T_i) = \emptyset$ for $i \in \{1, \dots, m-1\}$ due to the fact that $S_1, \dots, S_\ell$ and $T_1, \dots, T_{\ell'}$ are feasible solutions for the relaxation of the CTC problem and thus satisfy constraint (i) of that problem type.

Finally, define $\gamma \colon C \to \mathbb{Z}$ by $\gamma(v_i) = \alpha(v)$ for all $v \in V$ and $i \in [m-1]$. This implies that for any $C \in \mathcal{L}$ and a corresponding solution $S_1, \dots, S_\ell$ of the CTC problem's relaxation,

$$\gamma(C) = \sum_{i=1}^{m-1} \gamma(C \cap S_i) = \sum_{i=1}^{\ell} \alpha(S_i) \ ,$$

and hence $\gamma(C) \equiv r \pmod m$ if and only if $\sum_{i=1}^{\ell} \alpha(S_i) \equiv r \pmod m$.

Altogether, we obtain that C is an optimal solution of the CCSM problem given by N, $\mathcal{L}$, f and γ if and only if the corresponding sets $S_1, \dots, S_\ell$ form an optimal solution of the CTC problem with chain structure $S_\ell \subseteq \dots \subseteq S_1$ and $\ell \leq m-1$. Observing that the CCSM problem can be obtained from the CTC problem in strongly polynomial time (recall that m is assumed to be a constant), and that transforming a CCSM solution to a CTC solution is immediate, finishes the proof. □

Finally, combining Corollary 2.44 and Lemmas 2.45 and 2.48, we can conclude Theorem 2.40.

Proof of Theorem 2.40. Given a CCTU problem, by Corollary 2.44 it is enough to solve the associated CTC problem. By Lemma 2.46, this problem has an optimal solution with chain structure and at most $m-1$ cuts—which is precisely the type of problem that can be strongly polynomially reduced to a congruency-constrained submodular minimization problem by Lemma 2.48. Note that in these reductions, the modulus m of the involved congruency-constraints is invariant, and m is a constant prime power by assumption. Hence, the final congruency-constrained submodular minimization problem is one with constant prime power modulus. Such problems can be solved in strongly polynomial time by Theorem 2.47. □

2.4.3 Matrices stemming from particular constant-size matrices

To complete the study of base block CCTU problems, we now cover CCTU problems with constraint matrices that fall into case (ii) of Theorem 2.14. In other words, we study matrices that can be obtained from the two matrices

$$\begin{pmatrix} 1 & -1 & 0 & 0 & -1 \\ -1 & 1 & -1 & 0 & 0 \\ 0 & -1 & 1 & -1 & 0 \\ 0 & 0 & -1 & 1 & -1 \\ -1 & 0 & 0 & -1 & 1 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 \end{pmatrix} \tag{2.16}$$

by repeatedly appending unit vector rows or columns, appending a copy of a row or column, and inverting the sign of a row or column. More generally, our arguments apply to any constraint matrices that can be obtained from constant-size matrices by repeatedly applying the aforementioned operations. More formally, let us introduce the following notion of a *core* of a totally unimodular matrix.

Definition 2.51. *Let T be a totally unimodular matrix. A submatrix of T is a core of T if it is a smallest possible submatrix of T that can be obtained by iteratively deleting*

(i) any row or column with at most one non-zero entry, or
(ii) any row or column appearing twice or whose negation is also in the matrix.

It can be observed that up to row and column permutations and sign changes of rows and columns, every totally unimodular matrix has a unique core, which we denote by $\operatorname{core}(T)$. Still, let us remark that we do not need uniqueness for our arguments and working with *any* core would be enough for us. In the context of CCTU problems, we show the following theorem.

Theorem 2.52. *CCTU problems with modulus m and a constraint matrix T that has a core of constant size can be solved in strongly polynomial time*

(i) by a randomized algorithm if the objective is unary encoded and m is constant, or
(ii) by a deterministic algorithm if m is a constant prime power.

In particular, Theorem 2.52 shows that CCTU problems with constant prime power modulus and constraint matrices that fall into case (ii) of Theorem 2.14 can be solved in strongly polynomial time. Theorem 2.52 immediately follows from the following more concrete lemma by solving each of the $m^{O(\ell)}$ many CCTU problems using Theorem 2.35 or Theorem 2.40.

Lemma 2.53. *Consider a CCTU problem with modulus m and constraint matrix T, and let ℓ be the number of columns of $\operatorname{core}(T)$. The CCTU problem can be reduced to $m^{O(\ell)}$ many CCTU problems, with constraint matrices of size linear in the size of T, that are network matrices and transposes of network matrices at the same time.*

Proof. Assume that we are given a normalized CCTU problem, which has the form

$$\min \left\{ c^\top x \;\middle|\; Tx \leq b,\ \gamma^\top x \equiv r \pmod m,\ x \in \mathbb{Z}^n_{\geq 0} \right\} \ ,$$

where T is a matrix that is obtained as follows: Start from the matrix $C = \operatorname{core}(T)$ that has ℓ many columns, and repeatedly append unit rows or columns, append a copy of a row or column, and invert the sign of a row or column. In this process, we say that a row or column *stems* from C if it either is a row or column of C, or it was obtained by copying a row or column that stems from C. Thus, we may rewrite the inequality system in the form

$$\begin{pmatrix} T^{11} & T^{12} \\ T^{21} & T^{22} \end{pmatrix} \begin{pmatrix} x^1 \\ x^2 \end{pmatrix} \leq \begin{pmatrix} b^1 \\ b^2 \end{pmatrix} \ , \tag{2.17}$$

where T^{11} comprises the rows and columns of T that stem from C, and the remaining matrix as well as the variables x and the right-hand side b are split accordingly. Note that while T is achieved as a construction starting from the TU matrix C, we could also start from the totally unimodular matrix obtained from C by appending a $\ell \times \ell$ identity matrix, and then perform the same operations to obtain a totally unimodular matrix of the form

$$\begin{pmatrix} S & 0 \\ T^{11} & T^{12} \\ T^{21} & T^{22} \end{pmatrix} . \tag{2.18}$$

Here S has ℓ many rows $s_i^\top$ for $i \in \{1, \dots, \ell\}$ where, without loss of generality, the support of $s_i^\top$ comprises precisely those columns that stem from column i of C in the construction. Our approach is to guess the ℓ many scalar products $s_i^\top x^1$ of an optimal solution $x^* = (x^1 \ \ x^2)$, and thereby reduce the problem to an easier one.

To this end, note that the rows $(s_i^\top \ \ 0)$ are TU-appendable to the constraint matrix T because the matrix in (2.18) is TU. Thus, because we work with a normalized problem, we know that there exists an optimal solution $x^* = (x^1 \ \ x^2)$ of the CCTU problem such that $s_i^\top x^1 \in \{-m+1, \dots, m-1\}$ (see Theorem 2.11). Consequently, it is enough to consider $(2m-1)^\ell$ many combinations of values that these scalar products may admit. Once we fix those values, we also know the value of $T^{11}x^1$: Indeed, it is easy to see that every row t of T^{11} is a linear combination of the rows s_i, and hence $t^\top x$ is a linear combination of $s_i^\top x$. Thus, for any guess $\sigma = (\sigma_1, \dots, \sigma_\ell)$ of the ℓ many scalar products $s_1^\top x^1, \dots, s_\ell^\top x^1$, and after computing $\tau = T^{11}x^1$, we may rewrite the system (2.17) in the form

$$\begin{pmatrix} S & 0 \\ -S & 0 \\ 0 & T^{12} \\ T^{21} & T^{22} \end{pmatrix} \begin{pmatrix} x^1 \\ x^2 \end{pmatrix} \leq \begin{pmatrix} \sigma \\ -\sigma \\ b^1 - \tau \\ b^2 \end{pmatrix} . \tag{2.19}$$

We claim that the new constraint matrix is a network matrix and the transpose of a network matrix at the same time. To this end, observe that the matrix

$$\begin{pmatrix} S & 0 \\ 0 & T^{12} \\ T^{21} & T^{22} \end{pmatrix} \tag{2.20}$$

can be obtained by performing the same steps as we perform to obtain the matrix in (2.18), but replacing the entries of C with zeros in the starting matrix. This makes the starting matrix being a network matrix and the transpose of a network matrix at the same time, and this property is invariant under the operations that we perform when constructing the matrix. Thus, the matrix in (2.20) is a network matrix and the transpose of a network matrix at the same time, and hence, so is the constraint matrix in (2.19).

To sum up, we reduce a CCTU problem with a constraint matrix that has core with ℓ columns, to $(2m-1)^\ell$ many CCTU problems with constraint matrices that are a network matrix and the transpose of a network matrix at the same time. Also note that the size of the new constraint matrix is linear in the size of the original constraint matrix. This proves the lemma. □

We remark that instead of guessing all ℓ many scalar products in the proof of Lemma 2.53, we could also guess all but four of them: This would guarantee that the resulting constraint matrix of the reduced problems has a core that consists of at most 4 rows, and hence does not fall into case (ii) of Theorem 2.14, and we can fall back to another case for solving the reduced problems. In particular, when applying Lemma 2.53 to a constraint matrix T falling into case (ii) of Theorem 2.14, guessing the scalar product of a single row would be enough.

2.5 Further details of our approach to R-CCTUF problems

In this section, we fill in details and formal proofs supplementing the overview of our approach to R-CCTUF problems given in Section 2.2.

2.5.1 Seymour's decomposition of TU matrices

Theorem 2.14 is, up to the constraints $n_A, n_B \geq 2$, one naturally equivalent way of stating Seymour's decomposition theorem for TU matrices (see, for example, [Sey80] or [Sch98]). The version presented in Theorem 2.14 is a variation thereof that additionally guarantees lower bounds on the number of rows n_A and n_B of the blocks A and B, respectively, obtained in 1-, 2-, and 3-sums, namely $n_A, n_B \geq 2$. Similar bounds were achieved by Artmann, Weismantel, and Zenklusen in [AWZ17]: They lower bound the number of *rows* k_A and k_B of the two blocks A and B by 2—hence applying their theorem to the transpose of a TU matrix gives the version that we need.

Although not exploited in our results, we remark that the method presented in [AWZ17] in fact allows for obtaining the lower bounds on the number of columns and the number of rows of A and B simultaneously, i.e., it can be guaranteed that in any 1-, 2-, and 3-sum, both matrices are at least 2×2 matrices.

2.5.2 Patterns

Recall that if the constraint matrix of the R-CCTUF problem that we consider is a 1-, 2-, or 3-sum, the problem can be written in the form

$$\begin{aligned} \begin{pmatrix} A & ef^\top \\ gh^\top & B \end{pmatrix} \cdot \begin{pmatrix} x_A \\ x_B \end{pmatrix} &\leq \begin{pmatrix} b_A \\ b_B \end{pmatrix} \\ \gamma_A^\top x_A + \gamma_B^\top x_B &\in R \pmod m \\ x_A \in \mathbb{Z}^{n_A},\ x_B &\in \mathbb{Z}^{n_B} \ , \end{aligned}$$

as also given in (2.1). After fixing $\alpha = f^\top x_B$ and $\beta = h^\top x_A$, the above problem splits into an A-problem and a B-problem as in (2.2) (whose only link is through the original congruency constraint, which translates into $r_A + r_B \in R$). Also recall that we let $\Pi \subseteq \mathbb{Z}^2$ denote all pairs (α, β) for which both the A-problem and the B-problem are feasible, and that by Lemma 2.16 we know that if the initial problem is feasible, then it is also feasible for a pair of scalar products $(\alpha, \beta) \in \Pi$ that additionally satisfy $\ell_0 \leq \alpha + \beta \leq u_0$, $\ell_1 \leq \alpha \leq u_1$, $\ell_2 \leq \beta \leq u_2$ for bounds ℓ_i, u_i that we can determine in strongly polynomial time, and that satisfy $u_i - \ell_i \leq m - |R|$. For this reason, we defined a narrowed down version of Π, namely

$$\Pi_{\text{narrowed}} := \Pi \cap \{(\alpha, \beta) \in \mathbb{Z}^2 \colon \ell_0 \leq \alpha + \beta \leq u_0,\ \ell_1 \leq \alpha \leq u_1,\ \ell_2 \leq \beta \leq u_2\} \ , \tag{2.21}$$

and only look for solutions with scalar products $(\alpha, \beta) \in \Pi_{\text{narrowed}}$. We also remind the reader that a narrowed pattern associated to the problem is given by $\pi \colon \Pi_{\text{narrowed}} \to 2^{\{0,\dots,m-1\}}$, where $\pi(\alpha, \beta)$ is the set of residues $r_B \in \{0, \dots, m-1\}$ for which the B-problem is feasible.

The shape of pattern supports

In what follows, we prove the following lemma on the shape of Π_{narrowed}.

Lemma 2.54. *In the above setup, we can in strongly polynomial time determine ℓ_i', u_i' for $i \in \{0, 1, 2\}$ with $u_i' - \ell_i' \leq m - |R|$ such that*

$$\Pi_{\textit{narrowed}} = \{(\alpha, \beta) \in \mathbb{Z}^2 \colon \ell_0' \leq \alpha + \beta \leq u_0',\ \ell_1' \leq \alpha \leq u_1',\ \ell_2' \leq \beta \leq u_2'\} \ .$$

We emphasize that the main contribution of Lemma 2.54 is not to find new bounds ℓ'_i, u'_i (they will simply be the tightest bounds such that Π_{narrowed} is contained in the resulting set), but that there are no holes within the shape given by the bounds. That is, there are no (α, β) satisfying the bounds, but such that there is no feasible solution of our R-CCTUF problem with scalar products (α, β). It turns out that Π has the same property in the following sense, and Lemma 2.54 will follow from that.

Lemma 2.55. *For $\Pi \subseteq \mathbb{Z}^2$ defined as above, $\operatorname{conv}(\Pi)$ is a polyhedron with edge directions in $\mathcal{D} := \{\pm\binom{1}{0}, \pm\binom{0}{1}, \pm\binom{1}{-1}\}$, i.e., there is an inequality description of all $(\alpha, \beta) \in \Pi$ that only consists of upper and/or lower bounds on α, β, and $\alpha + \beta$.*

We remark that when we refer to *edge directions* v of an integral polyhedron (with rational extremal rays in case of unboundedness), then we always choose v to be integral, i.e., $v \in \mathbb{Z}^n$, and such that the greatest common divisor of its coordinates is 1. In other words, a vector $v \in \mathbb{Z}^n$ is an *edge direction* of an integral polyhedron if there exist integral points x_1 and x_2 that lie on the same edge of P such that $x_1 = x_2 + v$, and the greatest common divisor of all components of v is 1.

Proof of Lemma 2.54. From Lemma 2.55 and (2.21), it follows immediately that Lemma 2.54 holds for

$$\begin{aligned}
\ell'_0 &= \min\{\alpha + \beta \colon (\alpha, \beta) \in \Pi_{\text{narrowed}}\} \quad \text{and} \quad u'_0 = \max\{\alpha + \beta \colon (\alpha, \beta) \in \Pi_{\text{narrowed}}\} \ , \\
\ell'_1 &= \min\{\alpha \colon (\alpha, \beta) \in \Pi_{\text{narrowed}}\} \quad \text{and} \quad u'_1 = \max\{\alpha \colon (\alpha, \beta) \in \Pi_{\text{narrowed}}\} \ , \quad \text{and} \\
\ell'_2 &= \min\{\beta \colon (\alpha, \beta) \in \Pi_{\text{narrowed}}\} \quad \text{and} \quad u'_2 = \max\{\beta \colon (\alpha, \beta) \in \Pi_{\text{narrowed}}\} \ .
\end{aligned}$$

To see that we can determine ℓ'_i and u'_i in strongly polynomial time, we exploit that by Observation 2.27,

$$\begin{array}{rcrcrcl}
 & & Ax_A & + & ef^\top x_B & \leq & b_A \\
 & & gh^\top x_A & + & Bx_B & \leq & b_B \\
\ell_0 & \leq & h^\top x_A & + & f^\top x_B & \leq & u_0 \\
\ell_1 & \leq & & & f^\top x_B & \leq & u_1 \\
\ell_2 & \leq & h^\top x_A & & & \leq & u_2
\end{array}$$

is an inequality system with a totally unimodular constraint matrix. Here, the last three constraints precisely encode the constraints $(\alpha = f^\top x_B, \beta = h^\top x_A) \in \Pi_{\text{narrowed}}$, so pairs in Π_{narrowed} correspond to feasible solutions of the above system, and vice versa. Due to total unimodularity, we can find integral solutions of this system minimizing or maximizing the linear functions $\alpha = f^\top x_B$, $\beta = h^\top x_A$, and $\alpha + \beta = f^\top x_B + h^\top x_A$ by solving the corresponding relaxations using the approach of Tardos [Tar86] in strongly polynomial time, and the corresponding optimal values are precisely the values ℓ'_i and u'_i for $i \in \{0, 1, 2\}$ that we are looking for, and we have $u'_i - \ell'_i \leq u_i - \ell_i \leq m - |R|$ for all $i \in \{0, 1, 2\}$. □

To prove Lemma 2.55, we will observe that Π can be seen to essentially be a projection of the set of feasible solutions of the relaxation of the initial R-CCTUF problem. The following result will provide the necessary properties to conclude Lemma 2.55.

Theorem 2.56. *Let $T \in \{-1, 0, 1\}^{n \times k}$ be a totally unimodular matrix, let $b \in \mathbb{Z}^n$, and let $I \subseteq [n]$ be a subset of the column indices. Then, the axis-parallel projection $Q \subseteq \mathbb{R}^I$ of $P := \{x \in \mathbb{R}^n \colon Tx \leq b\}$ on the variables $(x_i)_{i \in I}$ has the following property: For any edge direction $v \in \mathbb{Z}^I$ of Q, and any $w \in \mathbb{Z}^n$ that is TU-appendable to T and supported on I, we have $w_I^\top v \in \{-1, 0, 1\}$.*

Here, for a vector $w \in \mathbb{R}^n$ and a subset $I \subseteq [n]$, we denote by w_I the restriction of w to the coordinate indices in I. Generally, note that for $I = [n]$, Theorem 2.56 is a statement about edge directions of polyhedra that are defined by totally unimodular matrices, characterized in terms of TU-appendable vectors. This shows another use of the concept of TU-appendable vectors and gives a result that might find independent applications.

Proof of Theorem 2.56. Assume for the sake of deriving a contradiction that Q has an edge direction $v \in \mathbb{R}^I$ such that there exists a vector $w \in \mathbb{R}^n$ that is supported on I and TU-appendable to T such that $w_I^\top v \notin \{-1, 0, 1\}$. Let $x_1, x_2 \in \mathbb{Z}^I$ lie on an edge of Q such that $x_1 = x_2 + v$, and observe that there exists $\lambda \in (0, 1)$ such that $y := (1-\lambda)x_1 + \lambda x_2 = x_1 + \lambda v$ is not integral, but satisfies $w_I^\top y = \eta$ for some $\eta \in \mathbb{Z}$, for example $\lambda = {}^1/_{|w_I^\top v|}$.

Now let $\overline{y}$ be a preimage of y under the axis-parallel projection from P to Q, and observe that $\overline{y}$ is a fractional solution of the system

$$\begin{array}{rcl} Tx & \leq & b \\ w^\top x & = & \eta \end{array},$$

which has a totally unimodular constraint matrix and integral right-hand sides, which implies that it describes an integral polyhedron. Thus, $\overline{y}$ can be written as a convex combination $\overline{y} = \sum_{i=1}^k \lambda_i \overline{z}_i$ of integral vectors $\overline{z}_i \in \{x \in \mathbb{Z}^n \colon Tx \leq b, w^\top x = \eta\}$, with coefficients $\lambda_i \in (0, 1)$ such that $\sum_{i=1}^k \lambda_i = 1$. Let $z_i \in \mathbb{Z}^I$ be obtained from $\overline{z}_i$ through axis-parallel projection to $\mathbb{R}^I$. Hence, $z_i \in Q \cap \mathbb{Z}^I$, $w_I^\top z_i = \eta$, and $y = \sum_{i=1}^k \lambda_i z_i$. Thus, we expressed y as a convex combination of points $z_i \in Q$. But recall that y lies on an edge of Q, and the only way to express such a point as a convex combination of others with non-zero coefficients is to use points from the same edge only, hence all z_i lie on the same edge. However, as the edge direction is v and $w_I^\top v \neq 0$, the point y is the only point on the edge satisfying $w_I^\top x = \eta$, so we must have $z_i = y$ for all $i \in [k]$. This contradicts that z_i are integral, while y is not. Thus, our assumption was wrong and Theorem 2.56 follows. □

Proof of Lemma 2.55. Note that Π contains precisely those pairs $(\alpha, \beta) \in \mathbb{Z}^2$ for which there exist $(x_A, x_B) \in \mathbb{Z}^{n_A} \times \mathbb{Z}^{n_B}$ such that $(x_A, x_B, \alpha, \beta)$ is a solution of the system

$$\begin{pmatrix} A & ef^\top & 0 & 0 \\ gh^\top & B & 0 & 0 \\ 0 & f^\top & -1 & 0 \\ h^\top & 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} x_A \\ x_B \\ \alpha \\ \beta \end{pmatrix} \begin{matrix} \leq \\ \leq \\ = \\ = \end{matrix} \begin{pmatrix} b_A \\ b_B \\ 0 \\ 0 \end{pmatrix}. \tag{2.22}$$

In other words, Π contains all integral points in the axis-parallel projection of the polyhedron P defined by (2.22) to the variables (α, β). Observe that the constraint matrix T in (2.22) is totally unimodular by Observation 2.27, so by Theorem 2.56 applied with I containing the indices of the variables α and β, we obtain that all edge directions $v \in \mathbb{Z}^2$ of $Q := \operatorname{conv}(\Pi)$ (which is precisely the projection of P to the variables (α, β) because P is integral) satisfy that for any integral vector w that is TU-appendable to T with support on the last two columns only, we have $w_I^\top v \in \{-1, 0, 1\}$. Obviously, the unit vectors $w \in \{\pm e_\alpha, \pm e_\beta\}$ (i.e., the vectors that are all zero except for ± 1 entries in corresponding to the variables α and β, and hence correspond to $w_I \in \{\pm \binom{1}{0}, \pm \binom{0}{1}\}$, respectively) are TU-appendable, and we claim that $w = \pm(e_\alpha + e_\beta)$ (corresponding to $w_I = \pm \binom{1}{1}$) are, as well.

Assuming this claim, the conclusion is immediate: We know that all edge directions $v \in \mathbb{Z}^2$ of Q are such that $w_I^\top v \in \{-1, 0, 1\}$ for all $w_I \in \{\pm \binom{1}{0}, \pm \binom{0}{1}, \pm \binom{1}{1}\}$. This leaves $v \in \{\pm \binom{1}{0}, \pm \binom{0}{1}, \pm \binom{1}{-1}\}$ as the only possible feasible edge directions, and hence the polyhedron Q can be described by inequalities bounding α, β, and $\alpha + \beta$ from above and/or below, as claimed by Lemma 2.55. Thus, we conclude the proof by showing the claim. To this end, define the three matrices

$$T' := \begin{pmatrix} A & ef^\top & 0 & 0 \\ gh^\top & B & 0 & 0 \\ 0 & f^\top & -1 & 0 \\ h^\top & 0 & 0 & -1 \\ 0 & 0 & 1 & 1 \end{pmatrix}, \quad T'' := \begin{pmatrix} A & ef^\top \\ gh^\top & B \\ 0 & f^\top \\ h^\top & 0 \end{pmatrix}, \quad \text{and} \quad T''' := \begin{pmatrix} A & ef^\top & 0 & 0 \\ gh^\top & B & 0 & 0 \\ 0 & f^\top & -1 & 0 \\ h^\top & 0 & 0 & -1 \\ h^\top & f^\top & 0 & 0 \end{pmatrix}.$$

The matrices T'' and T''' are auxiliary matrices we use in the following. To show the claim we need to show that T' is totally unimodular. Indeed, this will show TU-appendability of both $\binom{1}{1}$ and $\binom{-1}{-1}$ because

changing the sign of a row preserve total unimodularity of a matrix. To this end, consider any square submatrix $S = T'_{IJ}$ of T', for two index subsets I and J. If I does not contain all of the last three rows, we can perform a Laplace expansion of the determinant along unit rows and columns, which will suffice to get rid of the last two columns and the last row of T' (if they are present in S), and get that the determinant of S equals the determinant of a square submatrix of T'' in absolute value. But T'' is totally unimodular due to Observation 2.27, and hence the determinant of the submatrix that we are considering is in $\{-1, 0, 1\}$. If, on the other hand, $S = T'_{IJ}$ contains all of the last three rows, we know that its determinant is equal to the determinant of the submatrix of $S' = T'''_{IJ}$, where T''' is obtained from T' by adding the penultimate and third to last row to the last one. This operation does not change determinants, i.e., $\det(S) = \det(S')$. But T''' is totally unimodular by Observation 2.27, and hence $\det(S') \in \{-1, 0, 1\}$. In both cases, we obtain $\det(S) \in \{-1, 0, 1\}$, so T' is totally unimodular. □

An averaging lemma and linear patterns

In the proof of Theorem 2.56, one key idea was to average two integral solutions x_1 and x_2 to obtain a fractional solution that has an integral scalar product with some TU-appendable vector w, and then decompose that fractional solution into other feasible vectors that have the same integral scalar product with w. This idea can also be exploited to obtain the following result. Here, for an R-CCTUF problem of the form given in (2.1) (or its relaxation), we say that an R-CCTUF solution (or solution to its relaxation) $x = (x_A, x_B) \in \mathbb{Z}^{n_A} \times \mathbb{Z}^{n_B}$ is a solution for $(\alpha, \beta) \in \mathbb{Z}^2$ if $f^\top x_B = \alpha$ and $h^\top x_A = \beta$.

Lemma 2.57 (Averaging Lemma)**.** *Consider the relaxation of an R-CCTUF problem of the form given in (2.1). Let x^1 and x^2 be solutions for (α_1, β_1) and (α_2, β_2), respectively. Then, there exist solutions x^3 and x^4 for (α_3, β_3) and (α_4, β_4), respectively, such that $x^1 + x^2 = x^3 + x^4$, as well as*

$$\begin{aligned} &\left\lfloor \frac{\alpha_1 + \alpha_2}{2} \right\rfloor \leq \alpha_3, \alpha_4 \leq \left\lceil \frac{\alpha_1 + \alpha_2}{2} \right\rceil , \qquad \left\lfloor \frac{\beta_1 + \beta_2}{2} \right\rfloor \leq \beta_3, \beta_4 \leq \left\lceil \frac{\beta_1 + \beta_2}{2} \right\rceil , \\ &\textit{and} \quad \left\lfloor \frac{\alpha_1 + \beta_1 + \alpha_2 + \beta_2}{2} \right\rfloor \leq \alpha_3 + \beta_3, \alpha_4 + \beta_4 \leq \left\lceil \frac{\alpha_1 + \beta_1 + \alpha_2 + \beta_2}{2} \right\rceil . \end{aligned} \tag{2.23}$$

Proof. Consider the linear inequality system

$$\begin{array}{rcrcrcl} & & Ax_A & + & ef^\top x_B & \leq & b_A \\ & & gh^\top x_A & + & Bx_B & \leq & b_B \\ \left\lfloor \frac{1}{2}(\alpha_1 + \beta_1 + \alpha_2 + \beta_2) \right\rfloor & \leq & h^\top x_A & + & f^\top x_B & \leq & \left\lceil \frac{1}{2}(\alpha_1 + \beta_1 + \alpha_2 + \beta_2) \right\rceil \\ \left\lfloor \frac{1}{2}(\alpha_1 + \alpha_2) \right\rfloor & \leq & & & f^\top x_B & \leq & \left\lceil \frac{1}{2}(\alpha_1 + \alpha_2) \right\rceil \\ \left\lfloor \frac{1}{2}(\beta_1 + \beta_2) \right\rfloor & \leq & h^\top x_A & & & \leq & \left\lceil \frac{1}{2}(\beta_1 + \beta_2) \right\rceil \end{array} \tag{2.24}$$

and note that the claim of the lemma is that this system has two integral solution x^3 and x^4 with $x^1 + x^2 = x^3 + x^4$. To find these solutions, let T and q be such that $Tx \leq q$ is the system (2.24), and observe that T is totally unimodular by Observation 2.27. Then, the system

$$\begin{cases} Tx \leq q \\ T(x^1 + x^2 - x) \leq q \end{cases} \tag{2.25}$$

also has a totally unimodular constraint matrix, and $z := \frac{1}{2}(x^1 + x^2)$ is a (potentially fractional) solution of it. Because the bounds in the inequality constraints are all integral, we conclude that the linear system in (2.25) also has an integral solution x^3. Additionally, by symmetry it is immediate that $x^4 := x^1 + x^2 - x^3$ is another integral solution. In particular, we thus found x^3 and x^4 that are feasible for (2.24), and they satisfy $x^1 + x^2 = x^3 + x^4$, as desired. □

The above averaging lemma gives us a tool to analyze (narrowed) patterns $\pi\colon \Pi_{\text{narrowed}} \to 2^{\{0,\dots,m-1\}}$, because if the difference of (α_1,β_1) and (α_2,β_2) is large enough, the inequalities in Lemma 2.57 will make sure that (α_3,β_3) and (α_4,β_4) are different from (α_1,β_1) and (α_2,β_2), and hence also the solutions x^3 and x^4 are different from x^1 and x^2. Still, the relation $x^1+x^2=x^3+x^4$ allows us to draw conclusions about feasible residues in $\pi(\alpha_3,\beta_3)$ and $\pi(\alpha_4,\beta_4)$, and in particular relate them to residues in $\pi(\alpha_1,\beta_1)$ and $\pi(\alpha_2,\beta_2)$. We start by applying these ideas to narrowed patterns π that satisfy $|\pi(\alpha,\beta)|=1$ for all $(\alpha,\beta)\in\Pi_{\text{narrowed}}$. Again, we use the notation

$$\mathcal{D} \coloneqq \left\{ \pm\begin{pmatrix}1\\0\end{pmatrix}, \pm\begin{pmatrix}0\\1\end{pmatrix}, \pm\begin{pmatrix}1\\-1\end{pmatrix} \right\}$$

to denote the set of potential edge directions of $\operatorname{conv}(\Pi_{\text{narrowed}})$.

Lemma 2.58. *Consider a narrowed pattern $\pi\colon \Pi_{\text{narrowed}} \to 2^{\{0,\dots,m-1\}}$, and let $d_1,d_2\in\mathcal{D}$, $(\alpha,\beta)\in\mathbb{Z}^2$ such that $(\alpha,\beta)+\varepsilon_1 d_1+\varepsilon_2 d_2\in\Pi_{\text{narrowed}}$ for all $\varepsilon_1,\varepsilon_2\in\{0,1\}$, and let $r_{\varepsilon_1,\varepsilon_2}\in\{0,\dots,m-1\}$ be such that $\pi((\alpha,\beta)+\varepsilon_1 d_1+\varepsilon_2 d_2)=\{r_{\varepsilon_1,\varepsilon_2}\}$ for all $\varepsilon_1,\varepsilon_2\in\{0,1\}$. Then $r_{1,1}-r_{0,1}\equiv r_{1,0}-r_{0,0} \pmod m$.*

Proof. We first observe that we can assume without loss of generality that either $d_1=d_2$, or

$$\{d_1,d_2\}\in\left\{ \left\{\begin{pmatrix}1\\0\end{pmatrix},\begin{pmatrix}0\\1\end{pmatrix}\right\}, \left\{\begin{pmatrix}1\\0\end{pmatrix},\begin{pmatrix}1\\-1\end{pmatrix}\right\}, \left\{\begin{pmatrix}0\\1\end{pmatrix},\begin{pmatrix}-1\\1\end{pmatrix}\right\} \right\} .$$

Indeed, the above situation can always be achieved by changing the sign of d_1 and/or d_2. Changing the sign of d_1 can be done by choosing $(\alpha',\beta')=(\alpha,\beta)+d_1$ and the directions $d_1'=-d_1$ and $d_2'=d_2$, as we have $\{(\alpha',\beta')+\varepsilon_1 d_1'+\varepsilon_2 d_2'\colon \varepsilon_1,\varepsilon_2\in\{0,1\}\}=\{(\alpha,\beta)+\varepsilon_1 d_1+\varepsilon_2 d_2\colon \varepsilon_1,\varepsilon_2\in\{0,1\}\}$, and the statement that we want to show transforms accordingly. Analogously, we may also change the sign of d_2.

Now let x^1 be a solution for $(\alpha_1,\beta_1)=(\alpha,\beta)$, and let x^2 be a solution for $(\alpha_2,\beta_2)=(\alpha,\beta)+d_1+d_2$. Applying Lemma 2.57 to these solutions, we obtain that there exist solutions x^3 and x^4 for (α_3,β_3) and (α_4,β_4), respectively, such that $x^1+x^2=x^3+x^4$, and the inequalities in (2.23) are satisfied. On a case-by-case basis, it is immediate to see that with the above assumptions, the inequalities in (2.23) imply that $(\alpha_3,\beta_3),(\alpha_4,\beta_4)\in\{(\alpha,\beta)+d_1,(\alpha,\beta)+d_2\}$. Moreover, because $x^1+x^2=x^3+x^4$ also implies $(\alpha_1,\beta_1)+(\alpha_2,\beta_2)=(\alpha_3,\beta_3)+(\alpha_4,\beta_4)$, we must even have $\{(\alpha_3,\beta_3),(\alpha_4,\beta_4)\}=\{(\alpha,\beta)+d_1,(\alpha,\beta)+d_2\}$. We thus assume without loss of generality that $(\alpha_3,\beta_3)=(\alpha,\beta)+d_1$ and $(\alpha_4,\beta_4)=(\alpha,\beta)+d_2$.

By definition, we then have $r_{0,0}=\gamma_B^\top x_B^1$, $r_{1,0}=\gamma_B^\top x_B^3$, $r_{0,1}=\gamma_B^\top x_B^4$, and $r_{1,1}=\gamma_B^\top x_B^2$. The equality $x^1+x^2=x^3+x^4$ also implies $x_B^1+x_B^2=x_B^3+x_B^4$, and hence

$$r_{1,1}-r_{0,1}\equiv \gamma_B^\top x_B^2-\gamma_B^\top x_B^4=\gamma_B^\top x_B^3-\gamma_B^\top x_B^1\equiv r_{1,0}-r_{0,0} \pmod m ,$$

as desired. □

In what follows, for any $(\alpha_1,\beta_1),(\alpha_2,\beta_2)\in\mathbb{Z}^2$, we define

$$D_{(\alpha_1,\beta_1),(\alpha_2,\beta_2)} \coloneqq \left\{ (\alpha,\beta)\in\mathbb{Z}^2 \;\middle|\; \begin{array}{c} \min\{\alpha_1+\beta_1,\alpha_2+\beta_2\}\le\alpha+\beta\le\max\{\alpha_1+\beta_1,\alpha_2+\beta_2\} \\ \min\{\alpha_1,\alpha_2\}\le\alpha\le\max\{\alpha_1,\alpha_2\} \\ \min\{\beta_1,\beta_2\}\le\beta\le\max\{\beta_1,\beta_2\} \end{array} \right\} .$$

In particular, if $(\alpha_1,\beta_1),(\alpha_2,\beta_2)\in\Pi_{\text{narrowed}}$ for some domain Π_{narrowed} of a narrowed pattern, then by Lemma 2.55, we always also have $D_{(\alpha_1,\beta_1),(\alpha_2,\beta_2)}\subseteq\Pi_{\text{narrowed}}$. Also, if $(\alpha_3,\beta_3)\in D_{(\alpha_1,\beta_1),(\alpha_2,\beta_2)}$, then $D_{(\alpha_1,\beta_1),(\alpha_3,\beta_3)}\subseteq D_{(\alpha_1,\beta_1),(\alpha_2,\beta_2)}$.

Moreover, we define a *distance* notion for two pairs $(\alpha_1,\beta_1),(\alpha_2,\beta_2)\in\mathbb{Z}^2$ as follows: Consider the graph G on $\mathbb{Z}^2$ where two points $x,y\in\mathbb{Z}^2$ are connected by an edge if and only if $x-y\in\mathcal{D}$, and define the distance between (α_1,β_1) and (α_2,β_2) to be the length of a shortest path in G that connects the two points. It is easy to see that such a shortest path has all intermediate points within $D_{(\alpha_1,\beta_1),(\alpha_2,\beta_2)}$. Concretely, if (α_1,β_1) and (α_2,β_2) are at distance t, there are $d_1,\dots,d_t\in\mathcal{D}$ such that

(i) $(\alpha_1, \beta_1) + \sum_{i=1}^{\ell} d_i \in D_{(\alpha_1,\beta_1),(\alpha_2,\beta_2)}$ for all $\ell \in \{1, \dots, t\}$, and
(ii) $(\alpha_1, \beta_1) + \sum_{i=1}^{t} d_i = (\alpha_2, \beta_2)$.

Lemma 2.59. *Consider a narrowed pattern $\pi\colon \Pi_{\text{narrowed}} \to 2^{\{0,\dots,m-1\}}$ and a subset $\Pi_0 \subseteq \Pi_{\text{narrowed}}$ of the form*

$$\Pi_0 = \big\{(\alpha, \beta) \in \mathbb{Z}^2 \colon \ell_0 \le \alpha + \beta \le u_0,\ \ell_1 \le \alpha \le u_1,\ \ell_2 \le \beta \le u_2\big\}$$

with $|\pi(\alpha, \beta)| = 1$ for all $(\alpha, \beta) \in \Pi_0$, and let $r(\alpha, \beta) \in \{0, \dots, m-1\}$ be such that $\pi(\alpha, \beta) = \{r(\alpha, \beta)\}$. Then, for every $d \in \mathcal{D}$, there exists $r_d \in \{0, \dots, m-1\}$ such that for any $(\alpha, \beta) \in \Pi_0$ with $(\alpha, \beta) + d \in \Pi_0$,

$$r((\alpha, \beta) + d) - r(\alpha, \beta) \equiv r_d \pmod{m} \ .$$

Proof. Fix $d \in \mathcal{D}$. To derive the lemma, it is enough to show that for all pairs $(\alpha_1, \beta_1), (\alpha_2, \beta_2) \in \Pi_0$ with $(\alpha_1, \beta_1) + d, (\alpha_2, \beta_2) + d \in \Pi_0$, we have

$$r((\alpha_1, \beta_1) + d) - r(\alpha_1, \beta_1) \equiv r((\alpha_2, \beta_2) + d) - r(\alpha_2, \beta_2) \pmod{m} \ . \tag{2.26}$$

Note that if the distance between (α_1, β_1) and (α_2, β_2) is 0, there is nothing to show. Moreover, if that distance is 1, then a corresponding shortest path connecting (α_1, β_1) and (α_2, β_2) consists of a single step $d' \in \mathcal{D}$, i.e. $(\alpha_2, \beta_2) = (\alpha_1, \beta_1) + d'$, and (2.26) follows from applying Lemma 2.58 to (α_1, β_1) and the directions $d, d' \in \mathcal{D}$.

More generally, let us assume by induction that (2.26) holds whenever the distance of (α_1, β_1) and (α_2, β_2) is less than t, for some $t \ge 2$, and take two such pairs of distance equal to t. Then, a corresponding shortest path connecting the two points can be represented by $d_1, \dots, d_t \in \mathcal{D}$. Let $(\alpha', \beta') = (\alpha_1, \beta_1) + d_1$. By applying Lemma 2.58 to (α_1, β_1) and the directions $d, d_1 \in \mathcal{D}$, we obtain

$$r((\alpha_1, \beta_1) + d) - r(\alpha_1, \beta_1) \equiv r((\alpha', \beta') + d) - r(\alpha', \beta') \pmod{m} \ . \tag{2.27}$$

Note that this invocation of Lemma 2.58 requires $(\alpha_1, \beta_1) + d + d_1 \in \Pi_{\text{narrowed}}$, which holds because of the following: A shortest path P connecting (α_1, β_1) and (α_2, β_2) is inside $D_{(\alpha_1,\beta_1),(\alpha_2,\beta_2)}$, and shifting the whole path by d gives a shortest path connecting $(\alpha_1, \beta_1) + d$ and $(\alpha_2, \beta_2) + d$ that is inside $D_{(\alpha_1,\beta_1)+d,(\alpha_2,\beta_2)+d} \subseteq \Pi_{\text{narrowed}}$. Thus, in particular, because (α', β') is on P, $(\alpha', \beta') + d = (\alpha_1, \beta_1) + d + d_1$ is on the shifted path, and thus in Π_{narrowed}.

Additionally, because (α', β') and (α_2, β_2) are of distance at $t - 1$, the inductive assumption gives that

$$r((\alpha', \beta') + d) - r(\alpha', \beta') \equiv r((\alpha_2, \beta_2) + d) - r(\alpha_2, \beta_2) \pmod{m} \ . \tag{2.28}$$

Together, (2.27) and (2.28) imply the desired (2.26), thus completing the inductive step. □

Corollary 2.60. *Consider a narrowed pattern $\pi\colon \Pi_{\text{narrowed}} \to 2^{\{0,\dots,m-1\}}$ and a subset $\Pi_0 \subseteq \Pi_{\text{narrowed}}$ of the form*

$$\Pi_0 = \big\{(\alpha, \beta) \in \mathbb{Z}^2 \colon \ell_0 \le \alpha + \beta \le u_0,\ \ell_1 \le \alpha \le u_1,\ \ell_2 \le \beta \le u_2\big\}$$

with $|\pi(\alpha, \beta)| = 1$ for all $(\alpha, \beta) \in \Pi_0$, and let $r(\alpha, \beta) \in \{0, \dots, m-1\}$ be such that $\pi(\alpha, \beta) = \{r(\alpha, \beta)\}$. Then, there exist $r_0, r_1, r_2 \in \{0, \dots, m-1\}$ such that for all $(\alpha, \beta) \in \Pi_0$,

$$r(\alpha, \beta) \equiv r_0 + r_1\alpha + r_2\beta \pmod{m} \ .$$

Proof. Fix $(\alpha_0, \beta_0) \in \Pi_0$. Then for any $(\alpha, \beta) \in \Pi_0$, there exists $t \in \mathbb{Z}_{\ge 0}$ and $d_1, \dots, d_t \in \mathcal{D}$ such that (i) $(\alpha_\ell, \beta_\ell) \coloneqq (\alpha_0, \beta_0) + \sum_{i=1}^{\ell} d_i \in \Pi_0$ for all $\ell \in \{1, \dots, t\}$, and (ii) $(\alpha_t, \beta_t) = (\alpha, \beta)$. Now observe that we can write

$$r(\alpha, \beta) = r(\alpha_0, \beta_0) + \sum_{i=0}^{t-1} r((\alpha_i, \beta_i) + d_i) - r(\alpha_i, \beta_i) \ .$$

By Lemma 2.59, we know that for every $d \in \mathcal{D} \cap \{d_1, \dots, d_t\}$, there exists $r_d \in \mathbb{Z}$ such that $r((\alpha_i, \beta_i) + d) - r(\alpha_i, \beta_i) \equiv r_d \pmod m$ for all $i \in [t]$ with $d_i = d$. Observe that by definition, we must also have $r_{-d} \equiv -r_d \pmod m$, hence by aggregating terms in the above sum, we obtain

$$r(\alpha, \beta) = r(\alpha_0, \beta_0) + a \cdot r_{\binom{1}{0}} + b \cdot r_{\binom{0}{1}} + c \cdot r_{\binom{1}{-1}} \ , \tag{2.29}$$

where the coefficients $a, b, c \in \mathbb{Z}$ satisfy

$$\begin{pmatrix}\alpha\\ \beta\end{pmatrix} = \begin{pmatrix}\alpha_0\\ \beta_0\end{pmatrix} + a \cdot \begin{pmatrix}1\\ 0\end{pmatrix} + b \cdot \begin{pmatrix}0\\ 1\end{pmatrix} + c \cdot \begin{pmatrix}1\\ -1\end{pmatrix} \ . \tag{2.30}$$

The latter equation follows from aggregating terms in the sum in $(\alpha, \beta) = (\alpha_0, \beta_0) + \sum_{i=1}^{t} d_i$. We now distinguish two cases:

Case 1: One constraint in Π_0 is tight for all points in Π_0.
In this case, two among the three coefficients a, b, and c will be zero for any choice of $(\alpha, \beta) \in \Pi_0$. If $c = 0$ is one of the zero coefficients, then (2.30) implies that $a = \alpha - \alpha_0$ and $b = \beta - \beta_0$, and (2.29) gives that for all $(\alpha, \beta) \in \Pi_0$ we have

$$r(\alpha, \beta) = r(\alpha_0, \beta_0) + (\alpha - \alpha_0) \cdot r_{\binom{1}{0}} + (\beta - \beta_0) \cdot r_{\binom{0}{1}} \ ,$$

which is linear in α and β, as required. Otherwise, $a = b = 0$ and (2.30) implies that $c = \alpha - \alpha_0$, and hence by (2.29),

$$r(\alpha, \beta) = r(\alpha_0, \beta_0) + (\alpha - \alpha_0) \cdot r_{\binom{1}{-1}} \ ,$$

which is of the desired form, as well.

Case 2: No constraint in Π_0 is tight for all points in Π_0.
This implies that there is a pair $(\alpha', \beta') \in \Pi_0$ such that either

(i) $\binom{\alpha'}{\beta'}, \binom{\alpha'}{\beta'} + \binom{1}{0}, \binom{\alpha'}{\beta'} + \binom{0}{1} \in \Pi_0$, or
(ii) $\binom{\alpha'}{\beta'}, \binom{\alpha'}{\beta'} - \binom{1}{0}, \binom{\alpha'}{\beta'} - \binom{0}{1} \in \Pi_0$.

In the first case, we get

$$\begin{aligned} r_{\binom{1}{-1}} &\equiv r\left((\alpha', \beta') + (1, 0)\right) - r\left((\alpha', \beta') + (0, 1)\right) \\ &= \Big(r\left((\alpha', \beta') + (1, 0)\right) - r\left(\alpha', \beta'\right)\Big) - \Big(r\left((\alpha', \beta') + (0, 1)\right) - r(\alpha', \beta')\Big) \equiv r_{\binom{1}{0}} - r_{\binom{0}{1}} \ , \end{aligned}$$

and in the second case, we get

$$\begin{aligned} r_{\binom{1}{-1}} &\equiv r\left((\alpha', \beta') - (0, 1)\right) - r\left((\alpha', \beta') - (1, 0)\right) \\ &= \Big(r(\alpha', \beta') - r\left((\alpha', \beta') - (1, 0)\right)\Big) - \Big(r(\alpha', \beta') - r\left((\alpha', \beta') - (0, 1)\right)\Big) \equiv r_{\binom{1}{0}} - r_{\binom{0}{1}} \ . \end{aligned}$$

Note that this gives the same relation among the different vectors r_d in both cases. Using this in (2.29) together with the fact that (2.30) implies $a + c = \alpha - \alpha_0$ and $b - c = \beta - \beta_0$, we obtain that for all $(\alpha, \beta) \in \Pi_0$, we have

$$\begin{aligned} r(\alpha, \beta) &\equiv r(\alpha_0, \beta_0) + a \cdot r_{\binom{1}{0}} + b \cdot r_{\binom{0}{1}} + c \cdot \Big(r_{\binom{1}{0}} - r_{\binom{0}{1}}\Big) \\ &= r(\alpha_0, \beta_0) + (\alpha - \alpha_0) \cdot r_{\binom{1}{0}} + (\beta - \beta_0) \cdot r_{\binom{0}{1}} \pmod m \ , \end{aligned}$$

which is again a relation of the desired form. □

Proof of Theorem 2.18

We actually prove a slightly more general version of Theorem 2.18, in order not only to apply it to linear patterns π, but also to linear sub-patterns of a pattern π. To this end, let us formally repeat the definition of sub-patterns.

Definition 2.61. *Let $\pi\colon \Pi_{narrowed} \to 2^{\{0,\dots,m-1\}}$ be a narrowed pattern stemming from an R-CCTUF problem of the form given in (2.1). We say that $\widetilde{\pi}\colon \widetilde{\Pi} \to 2^{\{0,\dots,m-1\}}$ is a* sub-pattern *of π if the following holds:*

(i) $\widetilde{\Pi} \subseteq \Pi_{narrowed}$.

(ii) There are $\ell_i, u_i \in \mathbb{Z}$ for $i \in \{0,1,2\}$ such that

$$\widetilde{\Pi} = \left\{(\alpha,\beta) \in \mathbb{Z}^2 \colon \ell_0 \leq \alpha+\beta \leq u_0,\ \ell_1 \leq \alpha \leq u_1,\ \ell_2 \leq \beta \leq u_2\right\} \ .$$

(iii) $\widetilde{\pi}(\alpha,\beta) \subseteq \pi(\alpha,\beta)$ for all $(\alpha,\beta) \in \widetilde{\Pi}$.

Moreover, we say that a solution $x = (x_A, x_B)$ of an R-CCTUF problem of the form given in (2.1) is *covered* by a sub-pattern $\widetilde{\pi}$ if $\gamma^\top x_B \in \widetilde{\pi}(\alpha,\beta)$ for $\alpha = f^\top x_B$ and $\beta = h^\top x_A$.

Theorem 2.62. *Consider an R-CCTUF problem of the form given in (2.1), let π be an associated narrowed pattern, and let $\widetilde{\pi}$ be a linear sub-pattern of π with domain given by $\widetilde{\Pi} = \{(\alpha,\beta) \in \mathbb{Z}^2 \colon \ell_0 \leq \alpha+\beta \leq u_0,\ \ell_1 \leq \alpha \leq u_1,\ \ell_2 \leq \beta \leq u_2\}$, where $\ell_i, u_i \in \mathbb{Z}$ for $i \in \{0,1,2\}$. Then, we can in strongly polynomial time determine $r_0, r_1, r_2 \in \{0,1,\dots,m-1\}$ such that the R-CCTUF problem*

$$\begin{array}{rcrcrcrcl}
 & & Ax_A & + & ey_1 & & & \leq & b_A \\
 & & h^\top x_A & & & - & y_2 & = & 0 \\
\ell_0 & \leq & & & y_1 & + & y_2 & \leq & u_0 \\
\ell_1 & \leq & & & y_1 & & & \leq & u_1 \\
\ell_2 & \leq & & & & & y_2 & \leq & u_2 \\
 & & \gamma_A^\top x_A & + & r_1 y_1 & + & r_2 y_2 & \in & r_0 + R \pmod m \\
 & & x_A & & & & & \in & \mathbb{Z}^{n_A} \\
 & & & & y_1 & , & y_2 & \in & \mathbb{Z}
\end{array} \tag{2.31}$$

has a feasible solution if and only if the original R-CCTUF problem has one that is covered by $\widetilde{\pi}$. Moreover, a solution of one problem can be transformed into one of the other in strongly polynomial time.

Proof. By Corollary 2.60, there exist $r_0, r_1, r_2 \in \{0,\dots,m-1\}$ such that $r(\alpha,\beta) := -r_0 + \alpha r_1 + \beta r_2$ has the following property for each $(\alpha,\beta) \in \widetilde{\Pi}$: If x_B is a solution of the B-problem for (α,β), then $\gamma_B^\top x_B \equiv r(\alpha,\beta) \pmod m$. We claim that Theorem 2.62 holds for this choice of r_0, r_1, and r_2.

To see this, first let $(x_A, x_B) \in \mathbb{Z}^{n_A+n_B}$ be a solution of the original R-CCTUF problem that is covered by $\widetilde{\pi}$, i.e., a solution with scalar products $(\alpha,\beta) \in \widetilde{\Pi}$. We claim that (x_A, α, β) is a solution of (2.31). Indeed, feasibility for the original problem gives

$$\begin{array}{rcrcrcl}
Ax_A & + & ef^\top x_B & \leq & b_A & & \\
gh^\top x_A & + & Bx_B & \leq & b_B & & \\
\gamma_A^\top x_A & + & \gamma_B^\top x_B & \in & R & \pmod m & ,
\end{array}$$

and the first constraint is equivalent to $Ax_A + e\alpha \leq b_A$. Moreover, $h^\top x_A - \beta = 0$ is satisfied by definition of β, and the constraints in the third, forth, and fifth line of (2.31) are satisfied by $(y_1, y_2) = (\alpha,\beta)$ because $(\alpha,\beta) \in \widetilde{\Pi}$. Finally, the congruency constraint is satisfied because $\gamma_A^\top x_A - r_0 + \alpha r_1 + \beta r_2 = \gamma_A^\top x_A + r(\alpha,\beta) \equiv \gamma_A^\top x_A + \gamma_B^\top x_B \in R \pmod m$ is equivalent to $\gamma_A^\top x_A + r_1\alpha + r_2\beta \in r_0 + R \pmod m$.

On the other hand, for any solution (x_A, α, β) of (2.31), we get that $(\alpha,\beta) \in \widetilde{\Pi}$ due to the constraints in (2.31), and hence any solution x_B of the relaxation of the B-problem satisfies $\gamma_B^\top x_B \equiv r(\alpha,\beta) \pmod m$. From the same arguments as before, it follows that (x_A, x_B) is feasible for the original R-CCTUF problem.

To conclude the proof, observe that transforming the solution of the original problem to a solution of (2.31) only requires the computation of α and β. For the other way round, we need to compute a feasible solution to the relaxation of the B-problem, which can be done in strongly polynomial time using the algorithm of Tardos [Tar86]. □

Proof of Theorem 2.18. Because π is a linear pattern by assumption, we can apply Theorem 2.62 with $\widetilde{\pi} = \pi$, and Theorem 2.18 immediately follows. □

More properties of patterns and a proof of Lemma 2.21

After having studied linear patterns and sub-patterns so far in this section, we now focus on non-linear patterns π, i.e., patterns that have at least one pair (α, β) with $|\pi(\alpha, \beta)| \geq 2$ in their domain. The first lemma below shows how the property of having $|\pi(\alpha, \beta)| \geq 2$ propagates over the domain of a pattern, again using our averaging lemma, Lemma 2.57.

With the ultimate goal of this subsection being to prove Lemma 2.21, we first show Lemma 2.64, which showcases one important and repeatedly used situation in which Lemma 2.21 holds. We remark at this point that the requirement $|R| \geq m - 2$ stated in Lemma 2.21 is only due to Lemma 2.64. Hence, future attempts of overcoming this barrier using the ideas presented here will have to exploit setups beyond the one in Lemma 2.64. In contrast, the assumption in Lemma 2.21 of m being a prime number is exploited in several places.

Also, we remark that in this part, we aim at providing tools for analyzing (narrowed) patterns in a slightly more general setup than what we actually need. More precisely, in our concrete case it would be enough to analyze narrowed patterns that are contained in a rectangular box of scalar product pairs (α, β) of dimensions 3×3 (this follows by Lemma 2.16, for example, and our assumption $|R| \geq m - 2$). Still, we aim for the slightly more general presentation of our methods, which may be useful in potential future work on these topics, in particular when dropping the assumption $|R| \geq m - 2$.

We start by observing that Lemma 2.21 trivially holds in the case where the pattern π is linear, as we can then choose $\widetilde{\pi} = \pi$. In case of a non-linear pattern, we know that there is at least one pair (α, β) of scalar products in the domain of the pattern such that $|\pi(\alpha, \beta)| \geq 2$. If there exists a solution for such scalar products (α, β), item (ii) of Lemma 2.21 applies, so having many (α, β) with $|\pi(\alpha, \beta)| \geq 2$ is desirable. Luckily, the subsequent lemma proves that such (α, β) cannot appear in a very isolated way.

Lemma 2.63. *Consider a narrowed pattern $\pi \colon \Pi_{\text{narrowed}} \to 2^{\{0,\dots,m-1\}}$, let $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ and $d \in \mathcal{D}$ such that $(\alpha, \beta) + d, (\alpha, \beta) + 2d \in \Pi_{\text{narrowed}}$. If $|\pi(\alpha, \beta)| \geq 2$, then $|\pi((\alpha, \beta) + d)| \geq 2$, as well.*

Proof. Let x^1 and y^1 be feasible solutions of the relaxation of the underlying problem for scalar products (α, β) with different residues, i.e., $\gamma_B^\top x_B^1 \not\equiv \gamma_B^\top y_B^1 \pmod m$, and let x^2 be any solution of the relaxation of the problem for scalar products $(\alpha, \beta) + 2d$.

Applying the averaging lemma (Lemma 2.57) to the solutions x^1 and x^2, and to the solutions y^1 and x^2, we obtain solutions x^3, x^4 and solutions y^3, y^4, respectively, such that $x^1 + x^2 = x^3 + x^4$ and $y^1 + x^2 = y^3 + y^4$. Moreover, the inequalities (2.23) in Lemma 2.57 state that all of x^3, x^4, y^3, and y^4 are solutions for the scalar products $(\alpha, \beta) + d$. Consequently, $\{(\gamma_B^\top x_B^3 \bmod m), (\gamma_B^\top x_B^4 \bmod m), (\gamma_B^\top y_B^3 \bmod m), (\gamma_B^\top y_B^4 \bmod m)\} \subseteq \pi((\alpha, \beta) + d)$. To get that $|\pi((\alpha, \beta) + d)| \geq 2$, note that these residues satisfy

$$\gamma_B^\top x_B^1 + \gamma_B^\top x_B^2 \equiv \gamma_B^\top x_B^3 + \gamma_B^\top x_B^4 \pmod m \ , \quad \text{and} \quad \gamma_B^\top y_B^1 + \gamma_B^\top x_B^2 \equiv \gamma_B^\top y_B^3 + \gamma_B^\top y_B^4 \pmod m \ ,$$

and hence, because $\gamma_B^\top x_B^1 \not\equiv \gamma_B^\top y_B^1$, at least two of the residues among $(\gamma_B^\top x_B^3 \bmod m)$, $(\gamma_B^\top x_B^4 \bmod m)$, $(\gamma_B^\top y_B^3 \bmod m)$, and $(\gamma_B^\top y_B^4 \bmod m)$ must be distinct, which proves the lemma. □

Even non-linear patterns π might have several (α, β) in their support that satisfy $|\pi(\alpha, \beta)| = 1$. In Lemma 2.21, such squares may be covered by a linear sub-pattern, but it turns out that in general, there is no sub-pattern covering all pairs (α, β) with $|\pi(\alpha, \beta)| = 1$. The following lemma describes a configuration that allows for dealing with such pairs in a different way.

Lemma 2.64. *Consider an R-CCTUF problem of the form given in (2.1) with prime modulus m and $|R| \geq m-2$. Let $\pi\colon \Pi_{\text{narrowed}} \to 2^{\{0,\dots,m-1\}}$ be an associated narrowed pattern, and let $(\alpha,\beta) \in \mathbb{Z}^2$ and $d \in \mathcal{D}$ with*

$$(\alpha,\beta), (\alpha,\beta)+d, (\alpha,\beta)+2d \in \Pi_{\text{narrowed}} \ , \quad |\pi(\alpha,\beta)| = 1 \ , \quad \text{and} \quad |\pi((\alpha,\beta)+d)| \geq 2 \ .$$

If the problem has a solution with scalar products (α,β), one of the following holds:

(i) (α,β) satisfies case (i) of Lemma 2.21.
(ii) There is a solution with scalar products $(\alpha,\beta)+d$, i.e., $(\alpha,\beta)+d$ satisfies case (ii) of Lemma 2.21.

Proof. Assume that (α,β) does not satisfy case (i) of Lemma 2.21, i.e. it is not true that for any solution x_A of the A-problem for scalar products (α,β), there exists a solution x_B of the B-problem such that (x_A,x_B) is feasible for the original problem. Recall that given a feasible solution x_A of the relaxation of the A-problem and a feasible solution x_B of the relaxation of the B-problem for the same scalar products (α,β), the combined solution (x_A,x_B) is feasible for the original problem if and only if it satisfies the congruency constraint $\gamma_A^\top x_A + \gamma_B^\top x_B \in R \pmod m$. As $|\pi(\alpha,\beta)| = 1$ by assumption, $r = (\gamma_B^\top x_B \bmod m)$ is the same for all feasible solutions x_B of the relaxation of the B-problem. Consequently, the only reason why a combined solution (x_A,x_B) can be infeasible is that $\gamma_A^\top x_A \notin R - r \pmod m$. On the other hand, because by assumption, the problem has a feasible solution with scalar products (α,β), there must also be another solution x'_A with $\gamma_A^\top x'_A \in R - r \pmod m$.

Define $\pi_A\colon \Pi_{\text{narrowed}} \to 2^{\{0,1,\dots,m-1\}}$ such that $\pi_A(\alpha',\beta')$ denotes, for every $(\alpha',\beta') \in \Pi_{\text{narrowed}}$, the set of residues $\gamma_A^\top x_A$ that can be achieved by solutions of the relaxation of the A-problem with scalar products (α',β'). Hence, π_A is defined identically to π, with the only difference that π_A captures attainable residues in the A-problem instead of the B-problem. Hence, by symmetry between the A-problem and B-problem, properties holding for π and the B-problem also hold for π_A and the A-problem. In particular, the previous argument showed that $|\pi_A(\alpha,\beta)| \geq 2$, and by definition of Π_{narrowed}, we know that the relaxation of the A-problem is feasible for all $(\alpha',\beta') \in \Pi_{\text{narrowed}}$. Thus, applying Lemma 2.63 to π_A, we obtain that $|\pi_A((\alpha,\beta)+d)| \geq 2$, as well.

The residues that a solution (x_A,x_B) of the relaxation of the original problem can achieve for scalar product $(\alpha,\beta)+d$ are given by the set $\pi_A((\alpha,\beta)+d) + \pi((\alpha,\beta)+d)$. By the Cauchy-Davenport Inequality (Lemma 2.20), which we can apply as m is a prime number by assumption, we have

$$|\pi_A((\alpha,\beta)+d) + \pi((\alpha,\beta)+d)| \geq \min\{m, |\pi_A((\alpha,\beta)+d)| + |\pi((\alpha,\beta)+d)| - 1\} \geq \min\{m,3\} \ .$$

As the set R of target residues satisfies $|R| \geq m-2$, we conclude that at least one of the achievable residues is a target residue, and hence there exists a solution of the problem with scalar products $(\alpha,\beta)+d$. □

To prove Lemma 2.21, we distinguish two cases based on whether the *interior* of the pattern domain is empty or not, where interior is defined as follows.

Definition 2.65. *For a set $\Pi \subseteq \mathbb{Z}^n$ of the form*

$$\Pi = \left\{(\alpha,\beta) \in \mathbb{Z}^2 \colon \ell_0 \leq \alpha+\beta \leq u_0,\ \ell_1 \leq \alpha \leq u_1,\ \ell_2 \leq \beta \leq u_2\right\} \tag{2.32}$$

with $\ell_i, u_i \in \mathbb{Z}$ for $i \in \{0,1,2\}$, we say that $(\alpha,\beta) \in \Pi$ is in the interior *of Π if none of the constraints in (2.32) are tight for (α,β). Else, we say that (α,β) is on the* boundary *of Π.*

In fact, for non-linear patterns π whose support has non-empty interior, we show that any pair (α,β) with $|\pi(\alpha,\beta)| = 1$ is part of a configuration of the type described by Lemma 2.64, leading to the following.

Lemma 2.66. *Consider a non-linear narrowed pattern π for a feasible R-CCTUF problem as given in (2.1) with prime modulus m and $|R| \geq m-2$. If the domain of π has non-empty interior, then (i) or (ii) in Lemma 2.21 holds.*

To prove this lemma, we study the structure of patterns more closely. We start with an observation, where again, $\mathcal{D} \coloneqq \{\pm\binom{1}{0}, \pm\binom{0}{1}, \pm\binom{1}{-1}\}$ denotes the set of all potential edge directions of the convex hull of a pattern domain.

Observation 2.67. *Let $\Pi \subseteq \mathbb{Z}^n$ be of the form*

$$\Pi = \big\{(\alpha, \beta) \in \mathbb{Z}^2 \colon \ell_0 \leq \alpha + \beta \leq u_0,\ \ell_1 \leq \alpha \leq u_1,\ \ell_2 \leq \beta \leq u_2\big\} \tag{2.33}$$

with $\ell_i, u_i \in \mathbb{Z}$ for $i \in \{0,1,2\}$. Then $(\alpha, \beta) \in \Pi$ is in the interior of Π if and only if $(\alpha, \beta) + d \in \Pi$ for all $d \in \mathcal{D}$.

Lemma 2.68. *Consider a narrowed pattern $\pi \colon \Pi_{\text{narrowed}} \to 2^{\{0,\dots,m-1\}}$ such that there exists $(\alpha_1, \beta_1) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha_1, \beta_1)| \geq 2$. Then, for every $(\alpha_2, \beta_2) \in \Pi_{\text{narrowed}} \setminus \{(\alpha_1, \beta_1)\}$, there exists $d \in \mathcal{D}$ such that $(\alpha_2, \beta_2) + d \in D_{(\alpha_1,\beta_1),(\alpha_2,\beta_2)}$ and $|\pi((\alpha_2, \beta_2) + d)| \geq 2$.*

Proof. For any two pairs $(\alpha, \beta), (\alpha', \beta') \in \mathbb{Z}^2$, denote

$$\Delta((\alpha, \beta), (\alpha', \beta')) \coloneqq \max\{|\alpha - \alpha'|, |\beta - \beta'|, |(\alpha + \beta) - (\alpha' + \beta')|\} \ .$$

We prove that Lemma 2.68 holds by induction on $\Delta = \Delta((\alpha_1, \beta_1), (\alpha_2, \beta_2))$. For the base case, note that $\Delta = 1$ implies that there exists $d \in \mathcal{D}$ such that $(\alpha_1, \beta_1) = (\alpha_2, \beta_2) + d$, so there is nothing to show. Thus, assume that Lemma 2.68 holds if $\Delta < t$ for some $t \in \mathbb{Z}_{\geq 2}$, and consider a situation with $\Delta = t$. Let x^1 and y^1 be two solutions for scalar products (α_1, β_1) with $\gamma_B^\top x_B^1 \not\equiv \gamma_B^\top y_B^1 \pmod m$. These solutions exist because by assumption, $|\pi(\alpha_1, \beta_1)| \geq 2$. Additionally, let x^2 be a solution for scalar products (α_2, β_2). Applying the averaging lemma (Lemma 2.57) to x^1 and x^2, and to y^1 and x^2, we obtain solutions x^3, x^4 and y^3, y^4, respectively, where $x^1 + x^2 = x^3 + x^4$ and $y^1 + x^2 = y^3 + y^4$. Observe that the inequalities (2.23) leave only one option for each of (α_3, β_3) and (α_4, β_4), and both of these are within $D_{(\alpha_1,\beta_1),(\alpha_2,\beta_2)}$. In particular, they satisfy

$$\Delta((\alpha_3, \beta_3), (\alpha_2, \beta_2)) \leq \lceil t/2 \rceil \quad \text{and} \quad \Delta((\alpha_4, \beta_4), (\alpha_2, \beta_2)) \leq \lceil t/2 \rceil \ .$$

We claim that either $|\pi(\alpha_3, \beta_3)| \geq 2$ or $|\pi(\alpha_4, \beta_4)| \geq 2$, which allows us to apply the inductive assumption, thus finishing the proof.

To show the claim, assume for the sake of deriving a contradiction that $|\pi(\alpha_3, \beta_3)| = |\pi(\alpha_4, \beta_4)| = 1$. Without loss of generality, let x^3 and y^3 be solutions for (α_3, β_3), while x^4 and y^4 are solutions for (α_4, β_4). Then, by the assumption, $\gamma_B^\top x_B^3 \equiv \gamma_B^\top y_B^3 \pmod m$, and $\gamma_B^\top x_B^4 \equiv \gamma_B^\top y_B^4 \pmod m$. Thus, we also obtain

$$\gamma_B^\top x_B^1 + \gamma_B^\top x_B^2 = \gamma_B^\top x_B^3 + \gamma_B^\top x_B^4 \equiv \gamma_B^\top y_B^3 + \gamma_B^\top y_B^4 = \gamma_B^\top y_B^1 + \gamma_B^\top x_B^2 \pmod m \ ,$$

but this contradicts the choice of x^1 and y^1 such that $\gamma_B^\top x_B^1 \not\equiv \gamma_B^\top y_B^1 \pmod m$. □

Lemma 2.69. *Consider a non-linear narrowed pattern $\pi \colon \Pi_{\text{narrowed}} \to 2^{\{0,\dots,m-1\}}$. Then, for every (α, β) in the interior of Π_{narrowed}, we have $|\pi(\alpha, \beta)| \geq 2$.*

Proof. Because the pattern π is non-linear, there exists $(\alpha_1, \beta_1) \in \Pi_{\text{narrowed}}$ such that $|\pi(\alpha_1, \beta_1)| \geq 2$. Consider any (α, β) in the interior of Π_{narrowed}. Then by Lemma 2.68, there exists $d \in \mathcal{D}$ such that $(\alpha, \beta) + d \in \Pi_{\text{narrowed}}$ and $|\pi((\alpha, \beta) + d)| \geq 2$. Because (α, β) is in the interior of Π_{narrowed}, we also have that $(\alpha + \beta) - d \in \Pi_{\text{narrowed}}$. Consequently, applying Lemma 2.63, we obtain that $|\pi(\alpha, \beta)| \geq 2$, as well. □

Having the above at hand, we are now ready to prove Lemma 2.66.

Proof of Lemma 2.66. If the problem has a solution for scalar products $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha, \beta)| \geq 2$, then case (ii) of Lemma 2.21 holds. Thus, assume that this is not the case, i.e., the problem only has solutions for scalar products $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha, \beta)| = 1$.

By Lemma 2.69, there is a scalar product (α', β') in the interior of Π_{narrowed} with $|\pi(\alpha', \beta')| \geq 2$. Applying Lemma 2.68 to (α', β') and (α, β), we obtain that there exists $d \in \mathcal{D}$ such that $(\alpha, \beta) + d \in D_{(\alpha,\beta),(\alpha',\beta')} \subseteq \Pi_{\text{narrowed}}$ and $|\pi((\alpha, \beta) + d)| \geq 2$. Note that because (α', β') is in the interior of Π_{narrowed}, we have $(\alpha', \beta') + d \in \Pi_{\text{narrowed}}$, and thus also $D_{(\alpha,\beta),(\alpha',\beta')+d} \subseteq \Pi_{\text{narrowed}}$. As $(\alpha, \beta) + d \in D_{(\alpha,\beta),(\alpha',\beta')}$, it is also true that $(\alpha, \beta) + 2d \in D_{(\alpha,\beta),(\alpha',\beta')+d}$, so we conclude that $(\alpha, \beta) + 2d \in \Pi_{\text{narrowed}}$.

Observe that (α, β) and d thus satisfy the assumptions of Lemma 2.64. Because we assumed that there are no scalar product pairs satisfying case (ii) of Lemma 2.21, Lemma 2.64 implies that here, (α, β) satisfies case (i) of Lemma 2.21. □

To prove Lemma 2.21, it remains to deal with patterns whose domain has empty interior, which is covered by the statement below.

Lemma 2.70. *Consider a non-linear narrowed pattern π for a feasible R-CCTUF problem as given in (2.1) with prime modulus m and $|R| \geq m - 2$. If the domain of π has empty interior, Lemma 2.21 holds.*

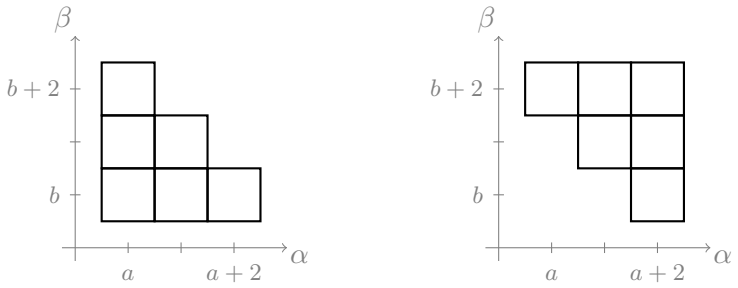


Figure 2.2: Shapes $\Pi_1^{(a,b)}$ (left) and $\Pi_2^{(a,b)}$ (right).

Before proving Lemma 2.70, we first observe structural properties of pattern domains with an empty interior. The possible shapes of such domains is very restricted. In particular, the subsequent lemma shows that they are either flat, or contained in small shapes $\Pi_0^{(a,b)}$ and $\Pi_1^{(a,b)}$ for $a, b \in \mathbb{Z}$ given by

$$\Pi_1^{(a,b)} := \left\{ (\alpha, \beta) \in \mathbb{Z}^2 \colon \begin{array}{c} a \leq \alpha \leq a+2 \\ b \leq \beta \leq b+2 \\ a+b \leq \alpha+\beta \leq a+b+2 \end{array} \right\}$$

$$\text{and} \quad \Pi_2^{(a,b)} := \left\{ (\alpha, \beta) \in \mathbb{Z}^2 \colon \begin{array}{c} a \leq \alpha \leq a+2 \\ b \leq \beta \leq b+2 \\ a+b+2 \leq \alpha+\beta \leq a+b+4 \end{array} \right\},$$

as depicted in Fig. 2.2.

In what follows, we define $\mathcal{D}^\perp := \{\pm\binom{1}{0}, \pm\binom{0}{1}, \pm\binom{1}{1}\}$, which is a set of vectors orthogonal to the potential edge directions $\mathcal{D}$ of the convex hull of a pattern support (see Lemma 2.55).

Lemma 2.71. *Let $\Pi \subseteq \mathbb{Z}^n$ be of the form*

$$\Pi = \left\{(\alpha, \beta) \in \mathbb{Z}^2 \colon \ell_0 \leq \alpha + \beta \leq u_0,\ \ell_1 \leq \alpha \leq u_1,\ \ell_2 \leq \beta \leq u_2\right\} \tag{2.34}$$

with $\ell_i, u_i \in \mathbb{Z}$ for $i \in \{0, 1, 2\}$, and assume that Π has empty interior. Then at least one of the following holds:

(i) A direction in $\mathcal{D}^\perp$ is a flat direction of width at most 1 for Π.

(ii) There are $a, b \in \mathbb{Z}$ and $i \in \{1, 2\}$ such that $\Pi \subseteq \Pi_i^{(a,b)}$.

Proof. Assume that item (i) does not hold, i.e., the three directions $\binom{1}{0}$, $\binom{0}{1}$, and $\binom{1}{1}$ are all of width at least 2, and let $(\alpha_0, \beta_0) \in \arg\max_{(\alpha,\beta)\in\Pi} (\alpha - \beta)$. Starting from a general Π of the form in (2.34), there are three cases to distinguish:

Case 1: $(\alpha_0, \beta_0) = (u_1, \ell_2)$ and the edge directions at (α_0, β_0) are $d_1 = \binom{0}{1}$ and $d_2 = \binom{-1}{0}$.
This implies that $(\alpha_0, \beta_0) + d_1, (\alpha_0, \beta_0) + d_2 \in \Pi$, hence we must have $\ell_0 \leq \alpha_0 + \beta_0 - 1$ and $u_0 \geq \alpha_0 + \beta_0 + 1$. Also note that because $\binom{1}{0}$, $\binom{0}{1}$ are directions of width at least 2, we must also have $\ell_1 \leq \alpha_0 - 2$ and $u_2 \geq \beta_0 + 2$. But this implies that $(\alpha_0 - 1, \beta_0 + 1)$ is in the interior of Π, contradicting the assumption.

Case 2: $(\alpha_0, \beta_0) = (u_1, \ell_0 - u_1)$ and the edge directions at (α_0, β_0) are $d_1 = \binom{0}{1}$ and $d_2 = \binom{-1}{1}$.
Because of the width 2 assumption, we must have $\ell_1 \leq u_1 - 2 = \alpha_0 - 2$ and $u_0 \geq \ell_0 + 2 = \alpha_0 + \beta_0 + 2$. Also, we must have $u_2 \geq \beta_0 + 2$. If $u_2 = \beta_0 + 2$, we obtain $\Pi \subseteq \Pi_2^{(\alpha_0-2,\beta_0)}$; if $u_2 > \beta_0 + 2$, then $(\alpha_0 - 1, \beta_0 + 2)$ is in the interior of Π, which is a contradiction.

Case 3: $(\alpha_0, \beta_0) = (u_0 - \ell_2, \ell_2)$ and the edge directions at (α_0, β_0) are $d_1 = \binom{-1}{0}$ and $d_2 = \binom{-1}{1}$.
Because of the width 2 assumption, we must have $u_2 \geq \ell_2 + 2 = \beta_0 + 2$ and $\ell_0 \leq u_0 - 2 = \alpha_0 + \beta_0 - 2$. Also, we must have $\ell_1 \leq \alpha_0 - 2$. If $\ell_1 = \alpha_0 - 2$, we obtain $\Pi \subseteq \Pi_1^{(\alpha_0-2,\beta_0)}$; if $\ell_1 < \alpha_0 - 2$, then $(\alpha_0 - 2, \beta_0 + 1)$ is in the interior of Π, which is a contradiction. □

Proof of Lemma 2.70. When dealing with patterns and showing that Lemma 2.21 holds, observe the following: If there exists a solution for scalar products $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha, \beta)| \geq 2$, then item (ii) of Lemma 2.21 applies, so we can assume from now on that any scalar products $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ for which there is a solution satisfy $|\pi(\alpha, \beta)| = 1$. To this end, we exploit two options: The first one is that such squares are contained in configurations of the type described by Lemma 2.64; the second one is to find a linear sub-pattern that has the corresponding (α, β) in its domain and thus covers potential solutions for these scalar products. We distinguish three cases based on the shape of the narrowed pattern domain Π_{narrowed}, which cover all the options by Lemma 2.71:

Case 1: There is a direction of width 0 for Π_{narrowed} in $\mathcal{D}^\perp$, but $\Pi_{\text{narrowed}} \not\subseteq \Pi_i^{(a,b)}$ for any $a, b \in \mathbb{Z}$ and $i \in \{1, 2\}$.

Case 2: There is no direction of width 0 but one of width 1 for Π_{narrowed} in $\mathcal{D}^\perp$, and $\Pi_{\text{narrowed}} \not\subseteq \Pi_i^{(a,b)}$ for any $a, b \in \mathbb{Z}$ and $i \in \{1, 2\}$.

Case 3: There are $a, b \in \mathbb{Z}$ and $i \in \{1, 2\}$ such that $\Pi_{\text{narrowed}} \subseteq \Pi_i^{(a,b)}$.

In case 1, observe that Π_{narrowed} is bounded, hence there exist $(\alpha_0, \beta_0) \in \Pi_{\text{narrowed}}$, $d \in \mathcal{D}$ and $t \in \mathbb{Z}_{\geq 0}$ such that

$$\Pi_{\text{narrowed}} = \{(\alpha_0, \beta_0) + id \colon i \in \{0, \dots, t\}\} \ ,$$

and because $\Pi_{\text{narrowed}} \not\subseteq \Pi_i^{(a,b)}$ for any a, b, and i, we must have $t \geq 3$. Because the pattern is non-linear, there exists $(\alpha_1, \beta_1) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha_1, \beta_1)| \geq 2$. We claim that $|\pi((\alpha_0, \beta_0) + id)| \geq 2$ for all $i \in \{1, \dots, t-1\}$.

To see the claim, we first show that $|\pi((\alpha_0, \beta_0) + d)| \geq 2$. If $(\alpha_0, \beta_0) \neq (\alpha_1, \beta_1)$, we may apply Lemma 2.68 to (α_0, β_0) and (α_1, β_1) to obtain that $|\pi((\alpha_0, \beta_0) + d)| \geq 2$. If, on the other hand, $(\alpha_0, \beta_0) = (\alpha_1, \beta_1)$, then apply Lemma 2.68 to $(\alpha_0, \beta_0) + 2d$ and (α_1, β_1), which also gives $|\pi((\alpha_0, \beta_0) + d)| \geq 2$. Finally, applying Lemma 2.68 once again to $(\alpha_0, \beta_0) + (i + 1)d$ and $(\alpha_0, \beta_0) + d$ for $i \in \{2, \dots, t-1\}$, we get that $|\pi(\alpha_0, \beta_0) + id)| \geq 2$. Thus, the only potential scalar product pairs with $|\pi(\alpha, \beta)| = 1$ are $(\alpha, \beta) \in \{(\alpha_0, \beta_0), (\alpha_0, \beta_0) + td\}$. These (α, β) are part of a configuration as described by Lemma 2.64, hence we get that if there is a solution for such (α, β), then either item (i) or (ii) of Lemma 2.21 holds.

In case 2, we note that the condition on a flat direction of width 1 implies that there exists $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ and two directions $d_1, d_2 \in \mathcal{D}$ with $d_1 \neq d_2$ and $d_1 \neq -d_2$ such that

$$\Pi_{\text{narrowed}} \subseteq \{(\alpha_0, \beta_0) + id_1 + \varepsilon d_2 \colon i \in \mathbb{Z}, \varepsilon \in \{0, 1\}\} \ .$$

Define $\Pi_0 \coloneqq \Pi_{\text{narrowed}} \cap \{(\alpha_0, \beta_0) + id_1 \colon i \in \mathbb{Z}\}$, and $\Pi_1 \coloneqq \Pi_{\text{narrowed}} \cap \{(\alpha_0, \beta_0) + id_1 + d_2 \colon i \in \mathbb{Z}\}$. If $|\Pi_0| < 3$ or $|\Pi_1| < 3$, then $\Pi_{\text{narrowed}} \subseteq \Pi_i^{(a,b)}$ for some $a, b \in \mathbb{Z}$ and $i \in \{0, 1\}$, which we excluded in this case. Thus, $|\Pi_0| \geq 3$ and $|\Pi_1| \geq 3$. Observe that because the pattern π is non-linear, for at least one $\varepsilon \in \{0, 1\}$, there exist $(\alpha, \beta) \in \Pi_\varepsilon$ with $|\pi(\alpha, \beta)| \geq 2$, and hence we may apply the analysis from case 1 to such Π_ε to see that if there is a solution for scalar products in Π_ε, then one of items (i) or (ii) of Lemma 2.21 applies. If not both Π_ε fall into the previous case, then there is one remaining, say $\Pi_{\varepsilon'}$, such that for all $(\alpha, \beta) \in \Pi_{\varepsilon'}$, $|\pi(\alpha, \beta)| = 1$. Then $\widetilde{\pi} \coloneqq \pi|_{\Pi_{\varepsilon'}}$ is a linear sub-pattern of π, hence if there is a solution covered by $\widetilde{\pi}$, then item (iii) of Lemma 2.21 applies. This completes the analysis of case 2.

Finally, we deal with case 3 on a case-by-case basis, by going through potential narrowed pattern domain shapes that are contained in sets of the form $\Pi_0^{(a,b)}$ or $\Pi_1^{(a,b)}$ for some $(a, b) \in \mathbb{Z}^2$, presented here by increasing size of $|\Pi_{\text{narrowed}}|$.

- $|\Pi_{\text{narrowed}}| \leq 3$: If $\Pi_{\text{narrowed}} = \{(\alpha_0, \beta_0) + id \colon i \in \{0, 1, 2\}\}$ for some $(\alpha_0, \beta_0) \in \mathbb{Z}^2$ and $d \in \mathcal{D}$, then the arguments from case 1 apply. Otherwise, restricting π to the subset of all $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha, \beta)| = 1$ gives a sub-pattern $\widetilde{\pi}$ for which Lemma 2.21 follows immediately.
- $|\Pi_{\text{narrowed}}| = 4$: Because we require $\Pi_{\text{narrowed}} \subseteq \Pi_i^{(a,b)}$ for some $(a, b) \in \mathbb{Z}^2$ and $i \in \{0, 1\}$, the only possible shapes of Π_{narrowed} are the ones given in Fig. 2.3.

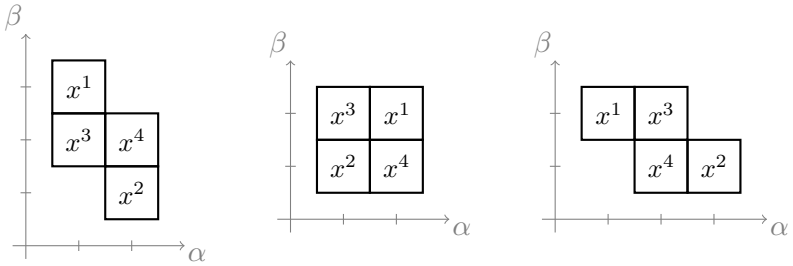


Figure 2.3: Narrowed pattern domains if $|\Pi_{\text{narrowed}}| = 4$ and $\Pi_{\text{narrowed}} \subseteq \Pi_i^{(a,b)}$ for some $(a, b) \in \mathbb{Z}^2$ and $i \in \{0, 1\}$.

Now in any of these three cases, let x^1 and x^2 be solutions of the relaxation of the R-CCTUF problem for scalar products (α, β) located in the pattern Π_{narrowed} as indicated in Fig. 2.3. Applying the averaging lemma (Lemma 2.57) to x^1 and x^2, we obtain solutions x^3 and x^4, and by the inequalities (2.23) in Lemma 2.57 and the property that $x^1 + x^2 = x^3 + x^4$, we may assume that x^3 and x^4 are solutions for scalar products located in the pattern Π_{narrowed} as indicated in Fig. 2.3, as well.
Now observe that the residues $\gamma_B^\top x_B^i$ satisfy $\gamma_B^\top x_B^1 + \gamma_B^\top x_B^2 = \gamma_B^\top x_B^3 + \gamma_B^\top x_B^4$, and hence the sub-pattern $\widetilde{\pi}$ that maps $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ to the residue $\gamma_B^\top x_B^i$, where x^i is the solution for (α, β), is a linear sub-pattern. More importantly, it is a linear sub-pattern that covers all $(\alpha, \beta) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha, \beta)| = 1$, and hence Lemma 2.21 follows.

- $|\Pi_{\text{narrowed}}| = 5$: Again, requiring $\Pi_{\text{narrowed}} \subseteq \Pi_i^{(a,b)}$ for some $(a, b) \in \mathbb{Z}^2$ and $i \in \{0, 1\}$, we can immediately enumerate the possible shapes of Π_{narrowed}, giving the list in Fig. 2.4.
Observe that each of the six pattern domains in Fig. 2.4 contains a segment of the form $\{(\alpha_0, \beta_0) + id \colon i \in \{0, 1, 2\}\}$ for some $(\alpha_0, \beta_0) \in \Pi_{\text{narrowed}}$ and $d \in \mathcal{D}$, namely the segments marked in gray in Fig. 2.4. If for some (α, β) on such a segment, we have $|\pi(\alpha, \beta)| \geq 2$, then the arguments of case 1 apply, and they show that if there is a solution with scalar products on the segment, then either item (i) or (ii) of Lemma 2.21 applies. The remaining scalar products (i.e., those not covered by the segment) can then be treated as in the case $|\Pi_{\text{narrowed}}| \leq 3$.

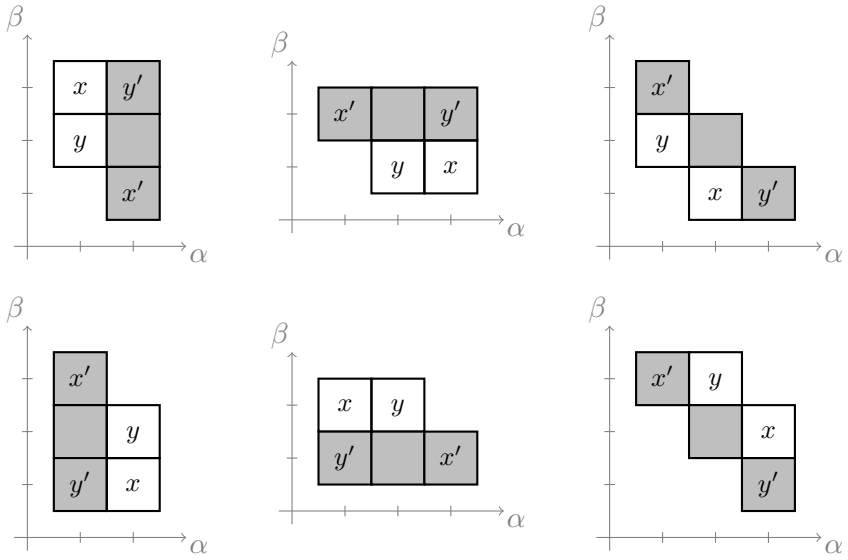


Figure 2.4: Narrowed pattern domains if $|\Pi_{\text{narrowed}}| = 5$ and $\Pi_{\text{narrowed}} \subseteq \Pi_i^{(a,b)}$ for some $(a,b) \in \mathbb{Z}^2$ and $i \in \{0,1\}$.

Thus, let us assume that for all (α,β) that are marked gray in Fig. 2.4, $|\pi(\alpha,\beta)| = 1$. This implies that at least one of the other two (α,β) in the pattern (marked with x and y in Fig. 2.4) must satisfy $|\pi(\alpha,\beta)| \geq 2$. In fact, we claim that in this case, both of the other two have that property. This is enough to conclude because then, if there exist solutions for these scalar product pairs, item (ii) of Lemma 2.21 applies. Hence, restricting π to the subset of all $(\alpha,\beta) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha,\beta)| = 1$ (i.e., those in the segment) gives a sub-pattern $\widetilde{\pi}$ for which Lemma 2.21 follows immediately.
To see the claim, we first assume that the pair (α,β) marked with an x in Fig. 2.4 satisfies $|\pi(\alpha,\beta)| \geq 2$. Let the pair marked x' be (α',β'), and apply Lemma 2.68 to (α,β) and (α',β') to obtain that there exists $d' \in \mathcal{D}$ such that $(\alpha',\beta')+d' \in \Pi_{\text{narrowed}}$ and $|\pi((\alpha',\beta')+d')| \geq 2$. By assumption, $(\alpha',\beta')+d'$ can thus not be within the gray segment, and in all cases, it is immediate to see that $(\alpha',\beta')+d$ must correspond to the spot in the pattern marked with y in Fig. 2.4. The same argument works with the roles of x and x' interchanged with y and y', so the claim follows.

- $|\Pi_{\text{narrowed}}| = 6$, i.e., $\Pi_{\text{narrowed}} = \Pi_i^{(a,b)}$ for some $(a,b) \in \mathbb{Z}^2$ and $i \in \{1,2\}$. Note that in such domains, every (α,β) is contained in a boundary segment of the form $\{(\alpha_0,\beta_0) + id \colon i \in \{0,1,2\}\}$ for some $(\alpha_0,\beta_0) \in \Pi_{\text{narrowed}}$ and $d \in \mathcal{D}$. As the pattern is non-linear, at least one of these segments contains (α,β) such that $|\pi(\alpha,\beta)| \geq 2$. Hence, the arguments of case 1 apply again, and if there is a solution with scalar products in that segment, then item (i) or (ii) of Lemma 2.21 applies. The remaining three scalar products (i.e., those not covered by the segment) can then be treated as in the case $|\Pi_{\text{narrowed}}| = 3$.

To finish the proof, observe that in every case where a linear sub-pattern $\tilde{\pi}$ was identified, this could be done in strongly polynomial time in the size of the underlying R-CCTUF problem. □

Proof of Lemma 2.21. If π is linear, we may choose $\widetilde{\pi} = \pi$, and item (iii) of Lemma 2.21 applies. For non-linear π, by Lemma 2.66 we have that Lemma 2.21 holds if the domain of π has non-empty interior, and by Lemma 2.70 it holds for domains with empty interior. □

2.5.3 Proof of Theorem 2.22

We can (after potentially permuting rows and columns of the constraint matrix such that A and B change their roles) assume that the matrix B has at most as many columns as A, i.e., $p = \min\{n_A, n_B\} = n_B$.

Furthermore, by Lemma 2.16, we can in strongly polynomial time determine $\ell_i, u_i \in \mathbb{Z}$ with $u_i - \ell_i \leq m - |R|$ for $i \in \{0,1,2\}$ such that if the R-CCTUF problem has a solution, then it has one with $\ell_0 \leq \alpha + \beta \leq u_0$, $\ell_1 \leq \alpha \leq u_1$, and $\ell_2 \leq \beta \leq u_2$. By Lemma 2.54, we can even choose these ℓ_i and u_i for $i \in \{0,1,2\}$ such that the corresponding narrowed pattern $\pi\colon \Pi_{\text{narrowed}} \to 2^{\{0,\dots,m-1\}}$ has domain

$$\Pi_{\text{narrowed}} = \left\{(\alpha,\beta) \in \mathbb{Z}^2 \colon \ell_0 \leq \alpha+\beta \leq u_0, \ell_1 \leq \alpha \leq u_1, \ell_2 \leq \beta \leq u_2\right\} \ .$$

For each $(\alpha,\beta) \in \Pi_{\text{narrowed}}$, we can now in strongly polynomial time compute the following:

- A solution $x_A^{\alpha,\beta}$ for the relaxation of the A-problem for scalar products (α,β).
- Exactly $t^{\alpha,\beta} := \min\{|\pi(\alpha,\beta)|, m-\ell+1\}$ solutions $x_{B,i}^{\alpha,\beta}$ of the relaxation of the B-problem with pairwise different residues $r_i^{\alpha,\beta} := \gamma_B^\top x_{B,i}^{\alpha,\beta}$ for $i \in [t^{\alpha,\beta}]$.

Note that computing the solutions $x_A^{\alpha,\beta}$ boils down to obtaining optimal vertex solutions to linear programs with a constraint matrix with bounded entries, which we can do in strongly polynomial time using the framework of Tardos [Tar86]. For fixed (α,β), computing the solutions $x_{B,i}^{\alpha,\beta}$ can be done by solving $m-\ell+1$ many B-problems with scalar products (α,β), i.e., by recursively calling our procedure, where we start with the full set $R_{B,1} = \{0,\dots,m-1\}$ of feasible target residues to get a solution $x_{B,1}^{\alpha,\beta}$, and then iterate using $R_{B,i+1} = R_{B,i} \setminus \{r_i^{\alpha,\beta}\}$, until we have $m-\ell+1$ many different residues, or we arrive at an infeasible problem. In the latter case, we computed $\pi(\alpha,\beta) = \{r_i^{\alpha,\beta} \colon i = 1,\dots,t^{\alpha,\beta}\}$, while in the first case, we just obtained a subset of $\pi(\alpha,\beta)$. Also note that each of the solutions $x_{B,i}^{\alpha,\beta}$ is obtained from an R-CCTUF problem with p variables, modulus m, and at most ℓ many target residues, and we solved at most $m-\ell+1 \leq 3$ of them for each $(\alpha,\beta) \in \Pi_{\text{narrowed}}$. As $|\Pi_{\text{narrowed}}| < (m-\ell+1)^2$, this procedure needed less than $3(m-\ell+1)^2$ many recursive calls in total, in accordance with the claim in Theorem 2.22.

Now invoke Lemma 2.21. We can directly check whether case (i) applies by going through all $(\alpha,\beta) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha,\beta)| = 1$. If case (i) applies for some $(\alpha,\beta) \in \Pi_{\text{narrowed}}$, the combination $(x_A^{\alpha,\beta}, x_{B,1}^{\alpha,\beta})$ must be a solution to the R-CCTUF problem.

If case (ii) of Lemma 2.21 applies, we can find a solution as follows: For $(\alpha,\beta) \in \Pi_{\text{narrowed}}$ with $|\pi(\alpha,\beta)| \geq m-\ell+1$, we can in fact immediately find a solution because by construction, all combinations $(x_A^{\alpha,\beta}, x_{B,i}^{\alpha,\beta})$ for $i = 1,\dots,m-\ell+1$ are feasible for the relaxation of the R-CCTUF problem, and then have pairwise different residues $\gamma_A^\top x_A^{\alpha,\beta} + \gamma_B^\top x_{B,i}^{\alpha,\beta}$. But in this case, one of them must have a residue in the set R that has size ℓ, thus giving a feasible solution. If on the other hand $1 < |\pi(\alpha,\beta)| \leq m-\ell$, we reduce the problem to the modified A-problem

$$\begin{array}{rcl} Ax_A & \leq & b_A - \alpha e \\ h^\top x_A & = & \beta \\ \gamma_A^\top x_A & \in & R' \pmod m \end{array} \ ,$$

where $R' = R - \pi(\alpha,\beta)$. This problem has a solution if and only if the original R-CCTUF problem has one: Note that any solution x_A of its relaxation can be combined with any solution x_B of the relaxation of the B-problem to obtain a solution (x_A, x_B) that is feasible for the relaxation of the original R-CCTUF problem. Moreover, the residues in R' are precisely those that allow us to obtain a combined solution (x_A, x_B) that also satisfies the original congruency constraint. Since m is a prime number and $|\pi(\alpha,\beta)| > 1$, Lemma 2.20 guarantees that $|R'| \geq |R| + 1 = \ell + 1$. To sum up, if case (ii) of Lemma 2.21 applies, we either find a feasible solution in strongly polynomial time, or we construct at most $|\Pi_{\text{narrowed}}| \leq (m-\ell+1)^2$ many new R-CCTUF problems with $n-p$ variables, modulus m, and at least $\ell+1$ target residues such that at least one of them has a feasible solution that, as seen immediately from the above discussion, can be transformed to a solution of the initial problem in strongly polynomial time.

If the above strategy to obtain a solution in case (ii) of Lemma 2.21 fails, we know that case (iii) of Lemma 2.21 applies. In this case, we know that the problem has a solution that is covered by the linear sub-pattern $\widetilde{\pi}$. Applying Theorem 2.62, we reduce the problem to an R-CCTUF problem with $n-p+2$ variables, modulus m, and ℓ target residues, with the additional property that the inequality system has an

equality constraint. This equality constraint allows for applying Theorem 2.19 to eliminate one variable and obtain an equivalent R-CCTUF problem with $n - p + 1$ variables, modulus m and ℓ target residues. It remains to observe that a solution of this problem can be immediately transformed back to a solution of the intermediate problem, and that solution can, by Theorem 2.62, be transformed back to a solution of the original problem in strongly polynomial time.

Altogether, after solving less than $3(m-\ell+1)^2$ many R-CCTUF problems with at most p variables and further strongly polynomial time operations, we can either obtain a feasible solution, or construct a family $\mathcal{F}$ of problems that have the properties claimed by Theorem 2.22. □

2.5.4 Proof of Theorem 2.19

By performing a pivoting operation (see Definition 2.13) on the element α, we obtain a new TU matrix which has $A - \alpha a_i a_2^\top$ as a submatrix. Hence the latter is also TU. Moreover, the two systems are equivalent since

$$\begin{array}{l} Ax \;+\; a_1 y \le b \\ a_2^\top x \;+\; \alpha y = \beta \end{array} \iff \begin{array}{r} Ax + a_1\alpha(\beta - a_2^\top x) \le b \\ y = \alpha(\beta - a_2^\top x) \end{array} \iff \begin{array}{r} (A - \alpha a_1 a_2^\top)x \le b - \alpha\beta a_1 \\ y = \alpha(\beta - a_2^\top x) \end{array} ,$$

where we use that $\alpha \in \{-1, 1\}$ since the matrix is TU and $\alpha \neq 0$. This completes the proof. □

2.5.5 Proof of Theorem 2.23

Consider an R-CCTUF problem

$$Tx \le b,\ \gamma^\top x \in R \pmod m,\ x \in \mathbb{Z}^n \ ,$$

where T falls into case (iv) of Theorem 2.14, and assume without loss of generality that the desired pivoted matrix arises from T by pivoting on the element in the first row and column.

Observe that due to Lemma 2.25, we can determine $u \in \mathbb{Z}$ such that the initial R-CCTUF problem is feasible if and only if

$$Tx \le b,\ y_1 \le u,\ \gamma^\top x \in R \pmod m,\ x \in \mathbb{Z}^n \tag{2.35}$$

is. Let $T := \begin{pmatrix} \varepsilon & p^\top \\ q & C \end{pmatrix}$, and let $Q \in \mathbb{Z}^{n\times n}$ be the unimodular matrix that corresponds to the column operations such that the first row of TQ is equal to the vector $(1, 0, \dots, 0)$. Then, if e_1 denotes the first n-dimensional unit vector,

$$\begin{pmatrix} T \\ e_1^\top \end{pmatrix} Q = \begin{pmatrix} \varepsilon & p^\top \\ q & C \\ 1 & 0 \end{pmatrix} Q = \begin{pmatrix} 1 & 0 \\ \varepsilon q & C - \varepsilon q p^\top \\ \varepsilon & -\varepsilon p^\top \end{pmatrix} \quad .$$

Thus, substituting $x = Qy$ and observing that $x \in \mathbb{Z}^n$ if and only if $y \in \mathbb{Z}^n$, we can rewrite the system in (2.35) as

$$\begin{pmatrix} 1 & 0 \\ \varepsilon q & C - \varepsilon q p^\top \\ \varepsilon & -\varepsilon p^\top \end{pmatrix} y \le \begin{pmatrix} b \\ u \end{pmatrix},\ (\gamma^\top Q) y \in R \pmod m,\ y \in \mathbb{Z}^n \ , \tag{2.36}$$

which is of the desired form. □

Appendix 2.A Detecting unboundedness of CCTU problems

Lemma 2.72. *A CCTU problem is unbounded if and only if it is feasible and its relaxation is unbounded. Moreover, given a feasible solution $x_0 \in \mathbb{Z}^n$ to an unbounded CCTU problem $\min\{c^\top x \colon Tx \le b,\ \gamma^\top x \equiv r \pmod m,\ x \in \mathbb{Z}^n\}$, one can efficiently determine a vector $v \in \mathbb{Z}^n$ such that $x_0 + k \cdot v$ is feasible for any $k \in \mathbb{Z}_{\ge 0}$ and $c^\top v < 0$.*

Proof. If a CCTU problem is unbounded, then it obviously has a feasible solution and its relaxation is unbounded. To show the other direction, consider a feasible CCTU problem

$$\min\left\{c^\top x \colon Tx \leq b,\ \gamma^\top x \equiv r \pmod m,\ x \in \mathbb{Z}^n\right\}$$

whose relaxation is unbounded. Thus, there exists a point $x_0 \in \mathbb{Z}^n$ that is feasible for the problem, and a direction $r \in \mathbb{Z}^n$ with $c^\top r < 0$ such that for any point x that is feasible for the relaxation, $x + r$ is feasible for the relaxation, as well. This implies that $x_k = x_0 + mkr$ satisfies $Tx_k \leq b$, $x_k \in \mathbb{Z}^n$, and $\gamma^\top x_k \equiv \gamma^\top x_0 \equiv r \pmod m$ for all $k \in \mathbb{Z}_{>0}$, and thus every such x_k is feasible for the CCTU problem. As $c^\top x_k = c^\top x_0 + mkc^\top r \to -\infty$ for $k \to \infty$, we conclude that the CCTU problem is unbounded.

Moreover, note that if the relaxation is unbounded, then one can obtain in polynomial time a vector $r \in \mathbb{Z}^n$ as described above, i.e., with $c^\top r < 0$ and $Tr \leq 0$. Hence, the vector $v := m \cdot r$ can be computed efficiently and has the properties claimed by the lemma. □

We remark that Lemma 2.72 extends to R-CCTUF problems and their optimization counterparts, as well.

Chapter 3

A new contraction technique with applications to congruency-constrained cuts

3.1 Introduction

Cuts in undirected graphs are a basic structure in Combinatorial Optimization with a multitude of applications. The global minimum cut problem, the minimum s-t cut problem, and the minimum odd cut problem are among the best known efficiently solvable minimum cut variants, and have been the cradle of many exciting algorithmic techniques. We study a generalization of these problems that we call *congruency-constrained minimum cut (CCMC)*, where a congruency constraint on the vertices in the cut is imposed, as described in the box below.[1]

Congruency-Constrained Minimum Cut (CCMC): Let $G = (V, E)$ be an undirected graph with edge weights $w\colon E \to \mathbb{R}_{\geqslant 0}$ and let $\gamma\colon V \to \mathbb{Z}_{\geqslant 0}$. Let $m \in \mathbb{Z}_{>0}$ and $r \in \mathbb{Z}_{\geqslant 0}$. The task is to find a minimizer of

$$\min \left\{ w(\delta(C)) \;\middle|\; \emptyset \subsetneq C \subsetneq V,\ \sum_{v \in C} \gamma(v) \equiv r \pmod{m} \right\} .$$

We call m the *modulus* of the problem, and we will typically consider m to be constant. Moreover, allowing general γ-values—instead of setting $\gamma(v) = 1$ for all $v \in V$, i.e., requiring that $|C| \equiv r \pmod{m}$—is merely for convenience. Indeed, the case of general γ-values can be reduced to the unit case by replacing each vertex v by a clique of $(\gamma(v) \bmod m)$-many vertices with large edge values if $\gamma(v) \not\equiv 0 \pmod{m}$, and a clique of size m if $\gamma(v) \equiv 0 \pmod{m}$.[2]

Apart from generalizing well-known cut problems, we are interested in the study of CCMC problems due to a link to an intriguing open question in Integer Programming, namely whether integer linear programs (ILPs) defined by an integer constraint matrix with bounded subdeterminants can be solved efficiently. Recently it was shown in [AWZ17] that ILPs of the form $\min\{c^\top x \mid Ax \leqslant b,\ x \in \mathbb{Z}^n\}$ can be solved efficiently if $A \in \mathbb{Z}^{m \times n}$ is *bimodular*, i.e., A has full column-rank and the determinant of every $n \times n$ submatrix of A is in $\{-2, -1, 0, 1, 2\}$. This result implies that if A is totally bimodular, i.e., all subdeterminants of A are in $\{-2, -1, 0, 1, 2\}$, then the corresponding ILP can be solved in polynomial time even without the requirement of A having full column rank (see [AWZ17] for details). This extends the well-known fact that ILPs with a totally unimodular constraint matrix can be solved efficiently; here, the absolute value of subdeterminants is bounded by 1. Only very limited techniques are known for attacking the question whether ILPs remain

[1] The minimum odd cut problem is captured by CCMC by choosing $m = 2$, $r = 1$, and $\gamma(v) = 1$ for all $v \in V$. Global minimum cuts correspond to $m = 1$, r arbitrary, and $\gamma(v) = 0$ for all $v \in V$, and s-t cuts can be modeled as minimum $\{s, t\}$-odd cuts, i.e., $m = 2$, $r = 1$, $\gamma(s) = \gamma(t) = 1$, and $\gamma(v) = 0$ for all $v \in V \setminus \{s, t\}$.

[2] We denote by $(\gamma(v) \bmod m)$ the smallest non-negative integer congruent to $\gamma(v)$ modulo m. Reducing modulo m is crucial to obtain a blow-up bounded by m, which, as mentioned, we will typically assume to be constant.

efficiently solvable in the Δ-modular case for some constant $\Delta > 2$, i.e., $\operatorname{rank}(A) = n$ and any $n \times n$ subdeterminant of A is in $\{-\Delta, -\Delta+1, \dots, \Delta\}$. Interestingly, to approach the bimodular case, classical combinatorial optimization problems with congruency constraints play a crucial role, and the problem can be reduced to certain types of congruency-constrained cut and flow problems (see [AWZ17]). In particular, CCMC with modulus m can be reduced to m-modular ILPs.[3] Hence, if one believes that Δ-modular ILPs can be solved efficiently for $\Delta = O(1)$, then CCMC should admit an efficient algorithm. Conversely, despite the fact that, for $\Delta \geq 3$, further gaps have to be overcome to reduce Δ-modular ILPs to congruency-constrained cut and flow problems, the results in [AWZ17] give hope that congruency-constrained cuts may be a useful building block for attacking Δ-modular ILPs, besides merely being a special case thereof.

Unfortunately, not much is known in terms of techniques to deal with congruency constraints in Combinatorial Optimization beyond parity constraints ($m = 2$). These constraints introduce an algebraic component to the underlying problem, which is a main additional hurdle to overcome. The results presented in Chapter 2 provide a first step towards $\Delta = 3$ by showing (among others) that for ILPs with totally unimodular constraint matrices and a congruency constraint with modulus 3, we can efficiently decide feasibility. These results are obtained through a significant extension of the approach in [AWZ17] in combination with new structural insights, and they further nurture the belief that general Δ-modular ILPs should be polynomial-time solvable for constant Δ. Closer to the congruency-constrained minimum cut setup, some progress has also been achieved for moduli m that are constant prime powers: it was shown in [NSZ19] that submodular function minimization under congruency constraints with such moduli can be solved efficiently. As the cut function is submodular, this implies that CCMC can be solved efficiently for m being a constant prime power. However, the techniques in [NSZ19] do not extend to general constant moduli m, due to intrinsic additional complications appearing when m has two different prime divisors.

The goal of this chapter is to show that contraction techniques, in the spirit of Karger's algorithm for global minimum cuts [Kar93; KS96], can be employed to approach CCMC. A naive way of using Karger for

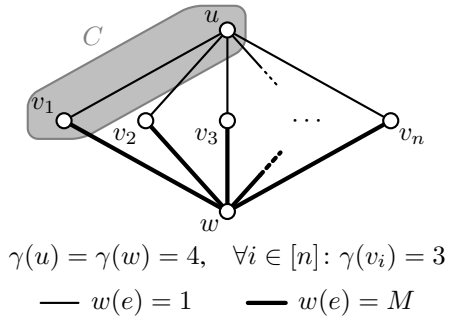


Figure 3.1: A CCMC instance with $m = 6$ and $r = 1$. Its optimal value is $n + M - 1$, achieved by the highlighted cut C.

CCMC faces several hurdles, which we exemplify through the CCMC instance in Fig. 3.1, parameterized by an even number n and a weight $M \geqslant 1$.[4] It consists of n paths of length 2 between two vertices u and w. An optimal cut is highlighted in gray. Karger's algorithm returns any global minimum cut in a graph $G = (V, E)$ with probability $\Omega(|V|^{-2})$, implying that there are at most $O(|V|^2)$ global minimum cuts. However, for

[3] If M is the incidence matrix of the digraph $H = (V, A)$ obtained by bidirecting G, then

$$\min \left\{ \sum_{a \in A} w_a y_a \;\middle|\; \begin{array}{c} Mx - y \leqslant 0,\ \sum_{v \in V} \gamma(v) x_v + zm = r,\ x_s = 1,\ x_t = 0, \\ x \in \{0,1\}^V,\ y \in \{0,1\}^A,\ z \in \mathbb{Z}\,. \end{array} \right\}$$

solves the congruency-constrained minimum s-t cut problem in G with edge weights w and congruency constraint $\gamma(C) \equiv r \pmod m$, where the cut corresponds to the set $C = \{v \in V \mid x_v = 1\}$. Moreover, the constraint matrix of the above ILP can be seen to be m-modular. Analogously to how global min cut problems can be reduced to min s-t cut problems, every CCMC problem can be reduced to solving linearly many problems of the above type.

[4] Even n ensures that $S = \{w, v_1, v_2, \dots, v_n\}$ is infeasible, i.e., $\gamma(S) \not\equiv 1 \pmod 6$.

$M = 1$, the CCMC problem in Fig. 3.1 has exponentially many optimal solutions. Hence, we cannot hope for an algorithm that returns any optimal solution with probability $\Omega(1/\mathrm{poly}(|V|))$. Moreover, if one of the n many u-w paths gets contracted, then the problem turns infeasible. It is not clear how to fix this. Even if we forbid contractions that make the instance infeasible, it is likely that in many of the u-w paths, one would contract the edge of weight 1. It is not hard to verify that Karger-type contractions would with high probability lead to a cut that is about twice as large as the minimum cut if M is chosen large (and this factor of 2 can be boosted further).

To overcome these and further hurdles, substantial changes seem necessary, and we introduce new techniques to employ contraction algorithms in our context. A key difference between our method and Karger's algorithm, as well as other contraction algorithms in a similar spirit (see a recent result of Chandrasekaran, Xu, and Yu [CXY18] for a nice adaptation of Karger's algorithm to the hypergraph k-cut problem), is that we do not contract edges of the graph. Instead, we define a distribution over pairs of vertices to contract that may not be connected by an edge. Moreover, we only look for contractions among certain vertices, namely those $v \in V$ fulfilling $\gamma(v) \not\equiv 0 \pmod{m}$. We show that splitting-off techniques from Graph Theory can be leveraged to design an efficient procedure to sample from a distribution of vertex pairs to contract with strong properties.

3.1.1 Our results

Our main result for CCMC via our new contraction technique is the following.

Theorem 3.1. *CCMC with constant modulus m admits a PRAS.*

Recall that a *PRAS* (*polynomial time randomized approximation scheme*) is an efficient procedure that, for any fixed $\varepsilon > 0$, returns a $(1+\varepsilon)$-approximate solution with high probability, by which we mean with probability at least $1 - 1/|V|$. As the focus of this chapter is existence of the PRAS claimed by Theorem 3.1, no efforts were made to optimize its running time. Nevertheless, let us mention that for $\varepsilon < 1$, we can bound the running time of our PRAS by $\log\left(\frac{w_{\max}}{w_{\min}}\right) \cdot |V|^{O\left(\frac{m \log m}{\varepsilon}\right)} \cdot 2^{O(m^2)}$, where $w_{\max}$ and $w_{\min}$ are the maximum and minimum edge weights occurring in the CCMC instance, respectively. A short discussion of this bound is given at the end of Section 3.2.

Moreover, for a constant composite modulus m that is the product of only two primes, we obtain an exact procedure.

Theorem 3.2. *CCMC with a constant modulus that is the product of two primes admits an efficient randomized algorithm that w.h.p. returns an optimal solution.*

This is in stark contrast to prior procedures, in particular for congruency-constrained submodular function minimization [NSZ19], which employ techniques that face hard barriers for moduli beyond prime powers.

Finally, in a similar spirit to Karger's algorithm for global minimum cuts, our contraction algorithm allows us to derive structural results on near-minimum congruency-constrained cuts. Whereas Karger's analysis shows that there are only polynomially many cuts of value at most a constant factor higher than the minimum cut, we cannot hope for results of this type: The example in Fig. 3.1 shows that CCMC problems can have exponentially many optimal solutions. For prime moduli, we show that near-minimum CCMC cuts are near-minimum cuts (without congruency constraint) in one of only a polynomial number of minimum s-t cut instances. These instances are defined on *contractions of* G, i.e., graphs obtained from $G = (V, E)$ by successively contracting nonempty node sets $S \subseteq V$. When *contracting* a set S, all vertices of S are replaced by a single vertex v_S with $\gamma(v_S) := \sum_{v \in S} \gamma(v)$, all edges with both endpoints in S are deleted, and each edge connecting a vertex in S to a vertex $u \in V \setminus S$ is replaced by an edge between u and v_S of the same weight. By construction, a cut C in a contraction of G naturally corresponds to a cut $\overline{C}$ in G of the same weight with $\gamma(C) = \gamma(\overline{C})$, and thus, we can identify these cuts.

Theorem 3.3. *Consider a CCMC problem on $G = (V, E)$ with constant prime modulus m and nonzero optimal solution value, and let $\kappa \geq 1$ be a constant. Then there is an efficient randomized method returning $\mathrm{poly}(|V|)$ many minimum s-t cut instances defined on contractions of G such that the following holds with high probability, where OPT denotes the optimal solution value of the CCMC problem. A cut $C \subsetneq V$, $C \neq \emptyset$, is a solution to CCMC of value at most $\kappa \cdot \mathrm{OPT}$ if and only if C is a feasible solution of value at most $\kappa \cdot \mathrm{OPT}$ in one of the minimum s-t cut instances (without congruency constraint).*

Theorems 3.2 and 3.3 are in fact consequences of more general structural properties of CCMC instances that are exhibited by our contraction algorithm (see Section 3.4 for more details).

3.1.2 Further discussion on related results

Work on minimum cut problems with constraints of congruency type date back to the early '80s, when Padberg and Rao [PR82] presented a method to efficiently find a minimum cut among all cuts with an odd number of vertices. Barahona and Conforti [BC87] later showed that efficient minimization is also possible over all cuts with an even number of vertices. Later works by Grötschel, Lovász, and Schrijver [GLS84], and by Goemans and Ramakrishnan [GR95] generalized these results, by showing that even any submodular function can be minimized over so-called triple families and, more generally, parity families. Submodular functions generalize cut functions, and triple as well as parity families capture congruency constraints with modulus 2. More generally, these approaches even allow for minimizing over all cuts $C \subseteq V$ of cardinality *not* congruent to r modulo m, for any integers r and m, which turns out to be a much simpler constraint than requiring a cardinality congruent to r modulo m. Indeed, CCMC for unbounded m quickly leads to NP-hard problems, as one could model an arbitrary cardinality constraint through a congruency constraint. In particular, if $G = (V, E)$ is a graph with an even number of vertices, then seeking a minimum cut C with $|C| \equiv 0 \pmod{|V|/2}$ captures the well-known minimum bisection problem. Khot [Kho06] showed that, unless NP has randomized sub-exponential time algorithms, the minimum bisection problem does not admit a polynomial time approximation scheme. Hence, it seems unlikely that a PRAS can be obtained for CCMC without a bound on the modulus.

We briefly mention further works linked to matrices with bounded subdeterminants. This includes the problem of finding a maximum weight independent set in a graph with constant odd-cycle packing number, for which a PTAS was obtained by Bock, Faenza, Moldenhauer, Vargas, and Jacinto [BFMR14]. This problem readily reduces to ILPs with bounded subdeterminants, due to an observation of Grossman, Kulkarni, and Schochetman [GKS95]. Another recent result by Eisenbrand and Vempala [EV17] is a randomized simplex-type algorithm for linear programming that is strongly polynomial whenever all subdeterminants of the constraint matrix defining the LP are bounded by a polynomial in the dimension of the problem. Furthermore, there has been interesting recent progress on the problem of approximating the largest subdeterminant of a matrix (see Di Summa, Eisenbrand, Faenza, and Moldenhauer [DEFM15], and Nikolov [Nik15]).

3.1.3 Organization of the chapter

We provide a discussion of the techniques leading to the main contribution of this chapter (Theorem 3.1) in Section 3.2. Section 3.3 expands on how to find a good distribution of vertex pairs to contract through splitting-off techniques, completing the proof of Theorem 3.1 given in Section 3.2. Section 3.4 is devoted to the exploration of furter structural properties of CCMC instances, leading to proofs of Theorems 3.2 and 3.3. Finally, Section 3.5 presents an alternative splitting-off approach for obtaining a suitable distribution of vertex pairs to contract.

3.2 An overview of our approach

As mentioned, the core of our approach is a contraction procedure inspired by Karger's global minimum cut algorithm, where we sample vertex pairs to be contracted from a certain distribution. In fact, the analysis of Karger's random contraction algorithm exploits that, whenever a random edge is contracted in a graph $G = (V, E)$, this contraction is *bad* with probability at most $k/|V|$ for some constant $k \in \mathbb{Z}_{>0}$. More precisely, in the analysis, an arbitrary minimum cut C is fixed, and a contraction is *bad* if it contracts two vertices on different sides of C. The probability of bad contractions being at most $k/|V|$ implies that by contracting until only k vertices remain, and then enumerating all cuts among those vertices, each minimum cut is found with probability at least $1/\binom{|V|}{k}$.

For CCMC, an important observation is that it suffices to decide which vertices in

$$V_{\not\equiv 0} := \{v \in V \mid \gamma(v) \not\equiv 0 \pmod m\}$$

are part of a solution. Indeed, for any cut C, the value of $\gamma(C)$ is determined by the intersection $C \cap V_{\not\equiv 0}$. Moreover, for any $U \subseteq V_{\not\equiv 0}$, the value

$$\nu(U) := \min\{w(\delta(C)) \mid \emptyset \subsetneq C \subsetneq V,\ C \cap V_{\not\equiv 0} = U\}$$

and a minimizer C_U can be obtained efficiently by a minimum cut computation in a contraction of G.[5] As $C_U \cap V_{\not\equiv 0} = U$, we have $\gamma(C_U) \equiv \gamma(U) \pmod m$.

Due to the above, instead of performing contractions over the full graph, as done in Karger's algorithm, we only contract pairs in $V_{\not\equiv 0}$, with the goal to reduce $V_{\not\equiv 0}$ to a constant-size set. If we achieve this, it suffices to enumerate over all $U \subseteq V_{\not\equiv 0}$ with $\gamma(U) \equiv r \pmod m$, minimize $\nu(U)$, and return a corresponding cut C_U. The theorem below is a key technical result of this chapter, and shows that a suitable distribution over vertex pairs in $V_{\not\equiv 0}$ to contract exists whenever the sum $\sum_{v \in V_{\not\equiv 0}} \nu(\{v\})$ is large enough.

Theorem 3.4. *Let $\mathcal{I} = (G, w, \gamma, m, r)$ be a CCMC instance on $G = (V, E)$. Let $\alpha \geqslant 0$ and $c > 0$ with $\sum_{v \in V_{\not\equiv 0}} \nu(\{v\}) > (2\alpha/c) \cdot |V_{\not\equiv 0}|$. Then, there is a distribution $\mathcal{D}$ over pairs in $V_{\not\equiv 0}$ such that $\Pr_{\{u,v\} \sim \mathcal{D}}\big[|\{u, v\} \cap C| = 1\big] \leqslant c/|V_{\not\equiv 0}|$ for any feasible solution C of $\mathcal{I}$ with $w(\delta(C)) \leqslant \alpha$. Moreover, there is a procedure to sample from $\mathcal{D}$ with running time polynomial in $|V|$ (independent of any other input parameters).*

To prove Theorem 3.4, we use weighted splitting-off techniques on G to construct a weighted auxiliary graph H on the vertex set $V_{\not\equiv 0}$. We show that by choosing edges of H with probabilities proportional to the edge weights, a distribution with the properties highlighted in Theorem 3.4 is obtained. Details of the proof are discussed in Section 3.3.

Theorem 3.4 with $\alpha = \mathrm{OPT}$ (or α slightly larger than OPT) implies that, whenever $\sum_{v \in V_{\not\equiv 0}} \nu(\{v\})$ is large compared to OPT, a contraction step has good success probability, similar to Karger's analysis. Otherwise, instead of performing a contraction, we approximately reduce the problem to another CCMC instance with smaller modulus. More precisely, if $\sum_{v \in V_{\not\equiv 0}} \nu(\{v\})$ is sufficiently small, then there are many vertices $v \in V_{\not\equiv 0}$ where the smallest cut $C_{\{v\}} \subseteq V$ separating v from $V_{\not\equiv 0} \setminus \{v\}$ has weight no more than $\beta = \kappa \cdot \mathrm{OPT}$ for a small constant κ. Such cuts are useful to modify a cut with wrong residue class. Indeed, consider a cut C with small weight $w(\delta_G(C))$, but $\gamma(C) \not\equiv r \pmod m$. Then, $\overline{C} := C \mathbin{\triangle} C_{\{v\}}$ satisfies $\gamma(\overline{C}) \equiv \gamma(C) \pm \gamma(v)$ (where the sign depends on whether $v \in C$), while the weight $w(\delta(\overline{C}))$ increased by at most β compared to $w(\delta(C))$; we recall that β is small with respect to OPT. Our plan is that if we have enough small cuts $C_{\{v\}}$, we can simplify the congruency constraint to one with smaller modulus, because

[5] If $U \notin \{\emptyset, V_{\not\equiv 0}\}$, then C_U can be computed by contracting U and $V_{\not\equiv 0} \setminus U$ in G, and by determining a minimum cut in the contracted graph that separates the two vertices corresponding to the contracted sets. If $U \in \{\emptyset, V_{\not\equiv 0}\}$, then $\nu(U)$ is obtained by contracting $V_{\not\equiv 0}$ and finding a global minimum cut C in the contracted graph, where C is chosen such that $C \cap V_{\not\equiv 0} = U$; this is achieved by replacing the computed global minimum cut by its complement if necessary.

the small cuts of type $C_{\{v\}}$ allow for moving solutions into the right residue class. This idea leads to the following notion of a *reduction family*.

Definition 3.5 (Reduction family)**.** *Let $\mathcal{I} = (G, w, \gamma, m, r)$ be a CCMC instance on the graph $G = (V, E)$. For $\beta \in \mathbb{R}_{\geqslant 0}$ and $q \in [m-1]$, a family $\mathcal{R}(\beta, q) \subseteq 2^V$ is a* reduction family for $\mathcal{I}$ *if*

(i) $\mathcal{R}(\beta, q) = \{R_1, R_2, \ldots, R_{2m_q-1}\}$ *with* $m_q \coloneqq \frac{m}{\gcd(m,q)}$, [6]
(ii) for each $i \in [2m_q - 1]$*, there is one vertex* $u_i \in R_i$ *with* $\gamma(u_i) \equiv q \pmod m$*, and* $\gamma(u) \equiv 0 \pmod m$ *for all other* $u \in R_i \setminus \{u_i\}$*,*
(iii) the vertices $u_1, \ldots u_{2m_q-1}$ *are distinct, and*
(iv) $w(\delta(R_i)) \leqslant \beta$ *for all* $i \in [2m_q - 1]$*.*

A reduction family $\mathcal{R}(\beta, q)$ allows for correcting the residue class $\gamma(C)$ of a solution C by any multiple of q modulo m, with losses in terms of cut weight controlled by the parameter β. Given a reduction family $\mathcal{R}(\beta, q)$, it is thus sufficient to find a solution C' satisfying $\gamma(C') \equiv r \pmod{m'}$ for $m' = \gcd(m, q)$. This is formalized in the following lemma.

Lemma 3.6 (Reduction lemma)**.** *Let $\mathcal{R}(\beta, q)$ be a reduction family for a CCMC instance (G, w, γ, m, r), and let $m' = \gcd(m, q)$. Given a cut $C' \subsetneq V$, $C' \neq \emptyset$, with $\gamma(C') \equiv r \pmod{m'}$, one can efficiently (in running time polynomial in $|V|$ and m) obtain a cut $C \subsetneq V$, $C \neq \emptyset$, such that*

(i) $w(\delta(C)) \leqslant w(\delta(C')) + \left(\frac{m}{m'} - 1\right)\beta$*, and*
(ii) $\gamma(C) \equiv r \pmod m$*.*

Proof. Let $\mathcal{R}(\beta, q) = \{R_1, R_2, \ldots, R_{2m_q-1}\}$ with distinct $u_i \in R_i$ for all $i \in [2m_q - 1]$ as given in item (ii) of Definition 3.5. We distinguish two cases: Either, there are m_q many vertices among the u_i with $u_i \in C'$, or there are m_q many with $u_i \notin C'$.

In the first case, assume w.l.o.g. that $u_1, \ldots, u_{m_q} \in C'$, and let $U_k \coloneqq \bigcup_{i=1}^{k} R_i$ for $k \in \{0, \ldots, m_q - 1\}$. We show that for some k, the set $C_k \coloneqq C' \triangle U_k$ has the desired properties. First observe that all C_k are cuts, as $C_0 = C'$ is a cut, and $u_1 \notin C_k \ni u_{m_q}$ for $k \in [m_q - 1]$. Moreover, $k \leqslant m_q - 1$ implies

$$w(\delta(C_k)) \leqslant w(\delta(C')) + \sum_{i=1}^{k} w(\delta(R_i)) \leqslant w(\delta(C')) + (m_q - 1)\beta \ . \tag{3.1}$$

Using $m_q = \frac{m}{\gcd(m,q)} = \frac{m}{m'}$, we see that (3.1) is precisely point (i) of Lemma 3.6 for C_k. To conclude, we show that there exists k such that C_k satisfies $\gamma(C_k) \equiv r \pmod m$, i.e., point (ii). Using that $\gamma(u) \equiv 0 \pmod m$ for all $u \in R_i \setminus \{u_i\}$, and $u_i \in C'$ for all $i \in [m_q]$, we obtain $\gamma(C_k) \equiv \gamma(C') - \sum_{i=1}^{k} \gamma(u_i) \equiv \gamma(C') - kq \pmod m$. It thus suffices to find $k \in \{0, \ldots, m_q - 1\}$ with $\gamma(C') - kq \equiv r \pmod m$, or equivalently,

$$kq \equiv \gamma(C') - r \pmod m \ . \tag{3.2}$$

By assumption, $\gamma(C') - r \equiv 0 \pmod{m'}$, so $r' \coloneqq \frac{\gamma(C')-r}{m'} \in \mathbb{Z}$, and $q' \coloneqq \frac{q}{m'} \in \mathbb{Z}$ because $m' = \gcd(m, q)$. Dividing (3.2) by m', we obtain the equivalent equation $kq' \equiv r' \pmod{m_q}$, which has a solution $k \in \{0, \ldots, m_q - 1\}$ as $\gcd(q', m_q) = 1$.

The second case, i.e., $u_1, \ldots, u_{m_q} \notin C'$, is similar: C_k always is a cut because $C_0 = C'$ is a cut, and $u_1 \in C_k \not\ni u_{m_q}$ for $k \geq 1$. Equation (3.1) remains true and implies point (i). For point (ii), we use $\gamma(C_k) \equiv \gamma(C') + \sum_{i=1}^{k} \gamma(u_i)$, and the above analysis results in $kq' \equiv -r' \pmod{m_q}$, admitting a solution $k \in \{0, \ldots, m_q - 1\}$.

Finally, given $\mathcal{R}(\beta, q)$ and C', checking which of the two cases applies can be done efficiently, as well as solving the respective congruence equation for k. Altogether, we conclude that a cut C with the desired properties can be obtained in running time polynomial in $|V|$ and m. □

[6] $\gcd(m, q)$ denotes the greatest common divisor of m and q.

The above reduction lemma applied with a reduction family $\mathcal{R}(\beta, q)$ allows for reducing the modulus from m to a divisor m' of m, which is strictly smaller than m, as $0 < q < m$. We call such a reduction to a smaller modulus through a reduction family a *reduction step*. Reduction families exist (and can be found efficiently) whenever Theorem 3.4 fails to guarantee a distribution with the desired properties for Karger-type contraction steps, i.e., whenever $\sum_{v \in V_{\neq 0}} \nu(\{v\})$ is small. In this case, there are many vertices $v \in V_{\neq 0}$ for which $\nu(\{v\})$ is small, i.e., the cut $C_{\{v\}}$ has small value. A subset of these cuts can then be used as a reduction family. This idea is concretized in Theorem 3.7 below.

Theorem 3.7. *Let $\mathcal{I}$ be a CCMC instance with modulus m and let $B > 0$. Assume that $|V_{\neq 0}| \geqslant 4m^2$ and $\sum_{v \in V_{\neq 0}} \nu(\{v\}) \leqslant B \cdot |V_{\neq 0}|$. Then, for some $q \in [m-1]$, one can efficiently (in running time polynomial in $|V|$ and m) obtain a reduction family $\mathcal{R}(2B, q)$ for $\mathcal{I}$.*

Proof. The conditions of Theorem 3.7 imply that at least $|V_{\neq 0}|/2 \geq 2m^2$ many vertices $v \in V_{\neq 0}$ satisfy $\nu(\{v\}) \leqslant 2B$; indeed, for otherwise

$$\sum_{v \in V_{\neq 0}} \nu(\{v\}) > 2B \cdot \frac{|V_{\neq 0}|}{2} = B \cdot |V_{\neq 0}| \ ,$$

However, the above inequality contradicts the second assumption, namely $\sum_{v \in V_{\neq 0}} \nu(\{v\}) \leqslant B \cdot |V_{\neq 0}|$, which implies the claim.

For every one of those $2m^2$ many vertices $v \in V_{\neq 0}$ with $\nu(\{v\}) \leqslant 2B$, there is a corresponding cut $C_{\{v\}}$ in G separating v and $V_{\neq 0} \setminus \{v\}$ of value $w(\delta(C_{\{v\}})) \leqslant 2B$; moreover, $\gamma(C_{\{v\}}) \equiv \gamma(v) \not\equiv 0 \pmod m$ because $v \in V_{\neq 0}$. Hence, by the pigeonhole principle, there exists $q \in [m-1]$ such that at least $\frac{2m^2}{m-1} > 2m$ many cuts $C_{\{v\}}$ satisfy $\gamma(C_{\{v\}}) \equiv q \pmod m$. Let $v_1, \ldots, v_{2m} \in V_{\neq 0}$ be distinct vertices such that $\{C_{\{v_i\}} \mid i \in [2m]\}$ are precisely $2m$ such cuts. For $m_q = \frac{m}{\gcd(m,q)}$, the family

$$\mathcal{R} = \left\{C_{\{v_1\}}, \ldots, C_{\{v_{2m_q-1}\}}\right\}$$

is well-defined and fulfils points (i) to (iv) in Definition 3.5 with parameters $\beta = 2B$ and q. To conclude the proof of Theorem 3.7, observe that $\mathcal{R}$ can be obtained in running time polynomial in $|V|$ and m by following the above constructive proof. □

A reduction step reduces the modulus m to a divisor strictly smaller than m, hence we can perform at most $\log_2(m)$ many reduction steps, and might end up solving a problem with modulus 1, i.e., an unconstrained minimum cut problem.

Altogether, the ingredients discussed above lead to Algorithm 3.1. This algorithm requires a guess α for the value of the optimal solution, which we can assume to know up to a factor of $(1+\varepsilon)$ by trying all polynomially many values

$$\alpha \in \{0\} \cup \left\{(1+\varepsilon)^i \cdot w_{\min} \,\middle|\, 0 \leqslant i \leqslant \lceil \log_{1+\varepsilon}(w_{\text{tot}}/w_{\min}) \rceil \right\} \ , \tag{3.3}$$

where $w_{\min} \coloneqq \min\{w(e) \mid e \in E,\ w(e) \neq 0\}$ and $w_{\text{tot}} \coloneqq w(E)$.

As long as $|V_{\neq 0}|$ is large, Algorithm 3.1 contracts two vertices of $V_{\neq 0}$ whenever the conditions of Theorem 3.4 are met with $c = 4m/\rho$. Note that every contraction step reduces the number of vertices in $V_{\neq 0}$ by one or two, depending on whether $\gamma(u) + \gamma(v) \not\equiv 0 \pmod m$ or not. The if-block in Algorithm 3.1 performs the enumeration step described earlier once there are at most $\max\left\{4m^2, 2 \cdot \left\lceil \frac{4m}{\rho} \right\rceil\right\}$ vertices left in $V_{\neq 0}$. If neither of the above is possible, then Theorem 3.7 and Lemma 3.6 allow for a reduction step, which is executed in the else-block, where we recursively invoke Algorithm 3.1 on an instance with strictly smaller modulus. Combining the above insights, we can prove the following guarantee for Algorithm 3.1.

Theorem 3.8. *Consider a CCMC instance (G, w, γ, m, r) with optimal solution value OPT. Let $\alpha \geqslant \mathrm{OPT}$ and $\rho > 0$. Algorithm 3.1 is an efficient procedure with running time bounded by $|V|^{O(1)} + 2^{O(m^2 + m/\rho)}$ that, by using α as an optimal value guess and ρ as error parameter, returns a solution with value at most $\mathrm{OPT} + \rho\, \alpha \log_2 m$ with probability at least $1/\binom{|V|}{\lceil 4m/\rho \rceil}$.*

Algorithm 3.1: Contraction-Reduction algorithm for CCMC.

Input : CCMC instance $\mathcal{I} = (G, w, \gamma, m, r)$ on $G = (V, E)$, error parameter $\rho > 0$, optimal value guess $\alpha \geqslant 0$.

while $|V_{\not\equiv 0}| > \max\left\{4m^2, 2 \cdot \left\lceil \frac{4m}{\rho} \right\rceil\right\}$ **and** $\sum_{v \in V_{\not\equiv 0}} \nu(\{v\}) > \frac{\rho\alpha}{2m} \cdot |V_{\not\equiv 0}|$ **do**

1. Sample a pair $\{u, v\}$ from the distribution $\mathcal{D}$ guaranteed by Theorem 3.4.
2. Modify G by contracting the set $\{u, v\}$.

if $|V_{\not\equiv 0}| \leqslant \max\left\{4m^2, 2 \cdot \left\lceil \frac{4m}{\rho} \right\rceil\right\}$ **then**

1. For every $S \subseteq V_{\not\equiv 0}$ with $\gamma(S) \equiv r \pmod m$, let
$$C_S \in \arg\min\{w(\delta(C)) \mid \emptyset \subsetneq C \subsetneq V,\ C \cap V_{\not\equiv 0} = S\} \ .$$
2. Among all cuts C_S obtained in step 1, let C be one of smallest value $w(\delta(C))$.

return *Cut corresponding to C in input graph before contractions.*

else

1. Use Theorem 3.7 to get reduction family $\mathcal{R}(\beta, q)$ for $\beta = \frac{\rho\alpha}{m}$ and some $q \in [m-1]$.
2. Let $m' = \gcd(m, q)$. Apply Algorithm 3.1 recursively to $\mathcal{I}' = (G, w, \gamma, m', r)$ with error parameter ρ and optimal value guess α to obtain a solution C' of $\mathcal{I}'$.
3. Apply Lemma 3.6 to get a solution C of $\mathcal{I}$ from C' and $\mathcal{R}(\beta, q)$.

return *Cut corresponding to C in input graph before contractions.*

Proof. The only randomized step of Algorithm 3.1 occurs in the while-loop, where pairs $\{u, v\}$ for contraction are sampled. For the analysis, we fix an optimal solution C_0 of $\mathcal{I}$, and first assume that no contraction is bad w.r.t. C_0, i.e., that no contraction step contracts two vertices on different sides of C_0 throughout Algorithm 3.1. Under this assumption, we prove by induction on m that Algorithm 3.1 returns a cut C satisfying $w(\delta(C)) \leqslant \mathrm{OPT} + \rho\alpha \log_2 m$.

If $m = 1$, then $V_{\not\equiv 0} = \emptyset$, hence the algorithm directly executes the if-block, where an unconstrained minimum cut problem is solved, giving an exact solution. This reflects that for $m = 1$, CCMC is an unconstrained minimum cut problem.

Now let $m > 1$. If no bad contraction is performed, C_0 remains feasible after the termination of the while-loop, and α remains an upper bound on the optimal solution value in the new contracted graph. If $|V_{\not\equiv 0}| \leqslant \max\left\{4m^2, 2 \cdot \left\lceil \frac{4m}{\rho} \right\rceil\right\}$, then, in the if-block, all remaining options are enumerated, and an optimal solution is found. Else, we have $|V_{\not\equiv 0}| \geqslant 4m^2$ and $\sum_{v \in V_{\not\equiv 0}} \nu(\{v\}) \leqslant \frac{\rho\alpha}{2m} \cdot |V_{\not\equiv 0}|$, hence by Theorem 3.7 with $B = \frac{\rho\alpha}{2m}$, a reduction family $\mathcal{R}(\frac{\rho\alpha}{m}, q)$ can be found efficiently. We have $q \in [m-1]$ by Theorem 3.7, so $m' = \gcd(m, q) < m$. Thus, by the inductive assumption, the recursive application of Algorithm 3.1 in step 2 of the else-block returns a solution $C' \subsetneq V$, $C' \neq \emptyset$, of $\mathcal{I}'$ with

$$\gamma(C') \equiv r \pmod{m'} \quad \text{and} \quad w(\delta(C')) \leqslant \mathrm{OPT} + \rho\alpha \log_2(m') \ . \tag{3.4}$$

Note that in the inequality, we used $\mathrm{OPT}(\mathcal{I}') \leqslant \mathrm{OPT}$, which follows from the fact that C_0 remains feasible for $\mathcal{I}'$. By (3.4) and Lemma 3.6, the solution C of $\mathcal{I}$ constructed in step 3 is a cut, satisfies $\gamma(C) \equiv r \pmod m$, and

$$w(\delta(C)) \leqslant w(\delta(C')) + \left(\frac{m}{m'} - 1\right) \frac{\rho\alpha}{m} \leqslant \mathrm{OPT} + \rho\alpha(\log_2 m' + 1) \leqslant \mathrm{OPT} + \rho\alpha \log_2 m \ ,$$

where the last inequality follows from $m' \leqslant m/2$, as m' is a divisor of m and strictly smaller than m. This concludes the induction. Thus, if no bad contraction steps are performed, a solution of value at most $\mathrm{OPT} + \rho\alpha \log_2 m$ is returned.

We now show that with probability at least $1/\binom{|V|}{\lceil 4m/\rho \rceil}$, no contraction step is bad w.r.t. C_0 throughout all recursive calls of Algorithm 3.1. Assume that overall, there are q contraction steps, and let $m_1, \ldots, m_q \in \mathbb{Z}_{\geq 0}$

and $s_1, \dots, s_q \in \mathbb{Z}_{\geq 0}$ be such that when the i^{th} contraction step is performed, the modulus is m_i and $|V_{\not\equiv 0}| = s_i$. By the condition in the while-loop, we know that when the i^{th} contraction is performed, then $\sum_{v \in V_{\not\equiv 0}} \nu(\{v\}) > \frac{\rho\alpha}{2m_i} \cdot |V_{\not\equiv 0}|$. Hence, by Theorem 3.4 with $c = \frac{4m_i}{\rho}$,

$$\Pr[\text{contraction } i \text{ is bad w.r.t. } C_0] \leqslant \frac{4m_i}{\rho \cdot |V_{\not\equiv 0}|} \leqslant \frac{k_i}{s_i} \ ,$$

where $k_i \coloneqq \lceil \frac{4m_i}{\rho} \rceil$, holds for all $i \in [q]$. Consequently,

$$\Pr[\text{no contraction is bad w.r.t. } C_0] \geqslant \prod_{i=1}^{q} \left(1 - \frac{k_i}{s_i}\right) \ . \tag{3.5}$$

To bound the latter product from below, we exploit the following three facts.

(i) For $x \in \mathbb{Z}$, let $\kappa(x) \coloneqq \max\{k_i \mid i \in [q] \colon x > 2k_i\}$, which is finite if $x > \min\{2k_1, \dots, 2k_q\}$. As $s_i > 2k_i$ by the contraction conditions, $\kappa(s_i) \geqslant k_i$.

(ii) The sequence $(s_i)_{i \in [q]}$ measures the size of $V_{\not\equiv 0}$, which decreases in each contraction step, and never increases. In particular,

$$|V| \geq s_1 > s_2 > \ldots > s_q \geq 2k_q + 1 \ .$$

(iii) The sequence $(k_i)_{i \in [q]}$ is decreasing, and it drops precisely when reduction steps are performed. Thus, there exists $p \leqslant q$ and $a_1, \dots, a_p \in [q]$ such that $k_{a_1} > k_{a_2} > \ldots > k_{a_p}$ are all values taken by the sequence $(k_i)_{i \in [q]}$. Note that in particular, $k_{a_1} = k_1 = \lceil \frac{4m}{\rho} \rceil$.

Using this, we get

$$\begin{aligned}
\prod_{i=1}^{q} \left(1 - \frac{k_i}{s_i}\right) &\overset{\text{(i)}}{\geqslant} \prod_{i=1}^{q} \left(1 - \frac{\kappa(s_i)}{s_i}\right) \overset{\text{(ii)}}{\geqslant} \prod_{i=2k_q+1}^{|V|} \left(1 - \frac{\kappa(i)}{i}\right) \\
&\overset{\text{(iii)}}{\geqslant} \prod_{i=2k_{a_p}+1}^{2k_{a_{p-1}}} \left(1 - \frac{k_{a_p}}{i}\right) \cdot \ldots \cdot \prod_{i=2k_{a_2}+1}^{2k_{a_1}} \left(1 - \frac{k_{a_2}}{i}\right) \cdot \prod_{i=2k_{a_1}+1}^{|V|} \left(1 - \frac{k_{a_1}}{i}\right) \\
&= \frac{\binom{2k_{a_p}}{k_{a_p}}}{\binom{2k_{a_{p-1}}}{k_{a_p}}} \cdot \ldots \cdot \frac{\binom{2k_{a_2}}{k_{a_2}}}{\binom{2k_{a_1}}{k_{a_2}}} \cdot \frac{\binom{2k_{a_1}}{k_{a_1}}}{\binom{|V|}{k_{a_1}}} \geqslant \frac{\binom{2k_{a_p}}{k_{a_p}}}{\binom{|V|}{k_{a_1}}} \geqslant \frac{1}{\binom{|V|}{\lceil 4m/\rho \rceil}} \ .
\end{aligned} \tag{3.6}$$

For the penultimate inequality we use $\binom{2k_{a_j}}{k_{a_j}} \geq \binom{2k_{a_j}}{k_{a_{j+1}}}$ for $j \in [p-1]$, and the last inequality follows from $\binom{2k_{a_p}}{k_{a_p}} \geq 1$. Combining (3.5) and (3.6), we get the desired bound.

To prove the running time guarantee, first assume that $|V| \leqslant m$, in which case the enumeration step is executed directly, giving a running time of $2^{O(|V|)}$, which is less than the claimed bound. Thus, assume that $|V| > m$ from now on. Overall, there are less than $|V|$ many contraction steps, each with a running time polynomial in $|V|$ by Theorem 3.4, giving a bound of the form $|V|^{O(1)}$ for all contraction steps together. Moreover, there are at most $\log_2(m)$ many reduction steps as observed earlier, each with running time polynomial in m and $|V|$ by Lemma 3.6 and Theorem 3.7. As $|V| > m$ by assumption, this shows that reduction steps take time bounded by $|V|^{O(1)}$, as well. Finally, the algorithm enumerates subsets of a set of size at most $\max\{4m^2, 2 \cdot \lceil 4m/\rho \rceil\}$, i.e., $2^{O(m^2 + m/\rho)}$ many sets. Adding these bounds gives the result. □

Guessing the optimal solution value up to a factor $(1+\varepsilon)$ and repeating Algorithm 3.1 polynomially often independently implies our main result, Theorem 3.1.

Proof of Theorem 3.1. For all polynomially many values of α given in (3.3), we run Algorithm 3.1 with $\rho = \frac{\varepsilon}{(1+\varepsilon)\log_2(m)}$ for $\binom{|V|}{\lceil 4m/\rho \rceil} \log |V|$ many times independently, and we return the best solution found over

all iterations. By Theorem 3.8, for $\alpha \in [\mathrm{OPT}, (1+\varepsilon)\mathrm{OPT})$, a single iteration returns a $(1+\varepsilon)$-approximate solution with probability at least $1/\binom{|V|}{\lceil 4m/\rho \rceil}$. Hence, among all iterations with this α, a $(1+\varepsilon)$-approximate solution is found with probability at least

$$1 - \left(1 - \frac{1}{\binom{|V|}{\lceil 4m/\rho \rceil}}\right)^{\binom{|V|}{\lceil 4m/\rho \rceil} \cdot \log |V|} \geqslant 1 - \mathrm{e}^{-\log |V|} = 1 - \frac{1}{|V|} \ . \qquad \square$$

From the above proof of Theorem 3.1, we can immediately obtain a bound on the running time of our PRAS for CCMC: There are

$$\left\lceil \log_{1+\varepsilon} \left(\frac{w_{\mathrm{tot}}}{w_{\min}} \right) \right\rceil + 2 = O\left(\frac{\log |V| + \log(\frac{w_{\max}}{w_{\min}})}{\log(1+\varepsilon)} \right)$$

many guesses for the value of α, where we exploit that $w_{\mathrm{tot}} \leqslant |V|^2 \cdot w_{\max}$. For each of these guesses, we run Algorithm 3.1 for $\binom{|V|}{\lceil 4m/\rho \rceil} \log |V| = |V|^{O\left(\frac{1+\varepsilon}{\varepsilon} m \log m\right)}$ many times independently. Finally, by Theorem 3.8, every such run takes time $|V|^{O(1)} + 2^{O\left(m^2 + \frac{1+\varepsilon}{\varepsilon} m \log m\right)}$. Together, this gives a running time bound of the form

$$\frac{\log(\frac{w_{\max}}{w_{\min}})}{\log(1+\varepsilon)} \cdot |V|^{O\left(\frac{1+\varepsilon}{\varepsilon} m \log m\right)} \cdot 2^{O(m^2)} \ ,$$

which can be simplified to $\log\left(\frac{w_{\max}}{w_{\min}}\right) \cdot |V|^{O\left(\frac{m \log m}{\varepsilon}\right)} \cdot 2^{O(m^2)}$ for $\varepsilon < 1$. We remark that, by using Megiddo's parametric search technique [Meg79; Meg83], we can get rid of the factor $\log\left(\frac{w_{\max}}{w_{\min}}\right)$ and thus obtain a strongly polynomial time algorithm at the expense of a larger constant in the exponent of $|V|$ that is hidden by the O-notation.

3.3 Good contraction distributions through splitting-off

To obtain a good distribution for Karger-type contractions (Theorem 3.4), we construct a weighted auxiliary graph $H = (V_{\not\equiv 0}, F)$, and then select a pair of vertices $\{u, v\} \in F$ for contraction in G with probabilities proportional to the edge weights in H. The construction of H is based on splitting-off techniques, which, loosely speaking, allow for modifying a given graph such that certain connectivity properties are preserved. Our interest lies in preserving the values $\nu(\{v\}) = \mu_{G,w}(\{v\}, V_{\not\equiv 0} \setminus \{v\})$ for all $v \in V_{\not\equiv 0}$, where we use the notation $\mu_{G,w}(A, B) := \min\{w(\delta(C)) \mid A \subseteq C \subseteq V \setminus B\}$. This is achieved by the following theorem.

Theorem 3.9. *Let $G = (V, E)$ be a graph with edge weights $w\colon E \to \mathbb{R}_{\geqslant 0}$, and let $Q \subseteq V$. There is a strongly polynomial time algorithm to obtain a graph $H = (Q, F)$ and edge weights $w_H\colon F \to \mathbb{R}_{\geqslant 0}$ such that*

(i) $w_H(\delta_H(q)) = \mu_{G,w}(\{q\}, Q \setminus \{q\})$ for all $q \in Q$, and
(ii) $w_H(\delta_H(C \cap Q)) \leqslant w(\delta_G(C))$ for all $C \subseteq V$.

We remark that similar theorems are known in literature, and there are various ways to derive the version above, which we need for our purposes. A splitting-off theorem of Lovász [Lov76] gives the existential result in an unweighted setting, and allows an immediate generalization to the weighted setting. Alternatively, the non-algorithmic version of Theorem 3.9 can also be seen to be an implication of a result on weakly parsimonious set functions by Bertsimas and Teo [BT97], which uses splitting-off, as well. In order to obtain strongly polynomial algorithms complementing the existential results, ideas of Frank [Fra92] can be used. A full proof of Theorem 3.9 combining Lovász' splitting-off result and Frank's ideas is given in Section 3.3.1. Let us now show how Theorem 3.9 is used to prove Theorem 3.4.

Proof of Theorem 3.4. Apply Theorem 3.9 to (G, w) with $Q = V_{\neq 0}$ to obtain the graph $H = (V_{\neq 0}, F)$ with weights w_H. The distribution $\mathcal{D}$ over vertex pairs $\{u, v\}$ we use is given by choosing $\{u, v\} \in F$ with probability proportional to $w_H(\{u, v\})$. By Theorem 3.9 (i),

$$2 \cdot w_H(F) = \sum_{v \in V_{\neq 0}} w_H(\delta_H(v)) = \sum_{v \in V_{\neq 0}} \mu_{G,w}(\{v\}, Q \setminus \{v\}) = \sum_{v \in V_{\neq 0}} \nu(\{v\}) \ .$$

If C is a solution of $\mathcal{I}$ with $w(\delta(C)) \leqslant \alpha$, then by choice of $\mathcal{D}$ and the above,

$$\Pr_{\{u,v\} \sim \mathcal{D}}\big[|\{u, v\} \cap C| = 1\big] = \frac{w_H(\delta_H(C \cap V_{\neq 0}))}{w_H(F)} \leqslant \frac{2 \cdot w(\delta_G(C))}{\sum_{v \in V_{\neq 0}} \nu(\{v\})} \leqslant \frac{c}{|V_{\neq 0}|} \ ,$$

as desired, where the inequalities are due to Theorem 3.9 (ii), $w(\delta_G(C)) \leqslant \alpha$, and the assumption $\sum_{v \in V_{\neq 0}} \nu(\{v\}) > \frac{2\alpha}{c} \cdot |V_{\neq 0}|$ in Theorem 3.4. To finish the proof, observe that the auxiliary graph H can be constructed in time polynomial in $|V|$ by Theorem 3.9, and the sampling procedure can be realized in running time polynomial in $|V|$, too. □

3.3.1 Proof of Theorem 3.9

As indicated above, Theorem 3.9 is a consequence of splitting-off techniques from Graph Theory, a fundamental tool dating back to the '70s [Lov76; Mad78; Lov79]. In this context, a graph is typically modified by repeatedly *splitting off* two edges from a vertex v, i.e., replacing two non-parallel edges $\{v, x\}$ and $\{v, y\}$ by a new edge $\{x, y\}$, or deleting two parallel edges incident to v. Denoting $\mu_G(A, B) := \min\{|\delta_G(C)| \mid A \subseteq C \subseteq V \setminus B\}$ for a graph $G = (V, E)$ and $A, B \subseteq V$, Lovász proved the following.

Theorem 3.10 (Lovász [Lov76])**.** *Let $G = (V, E)$ be Eulerian, let $Q \subseteq V$, and let $v \in V \setminus Q$. For every edge $\{v, x\} \in E$, there exists another edge $\{v, y\} \in E$ such that the graph G' arising from G by splitting off $\{v, x\}$ and $\{v, y\}$ from v satisfies*

$$\mu_G(\{q\}, Q \setminus \{q\}) = \mu_{G'}(\{q\}, Q \setminus \{q\}) \qquad \forall q \in Q \ .$$

Iterative applications of Theorem 3.10 for fixed $Q \subseteq V$ and $v \in V \setminus Q$ result in a new graph on the vertex set $V \setminus v$ only, without changing the value of minimum cuts separating a single vertex q from $Q \setminus \{q\}$, for all $q \in Q$. We aim for a generalization of this statement to a weighted setting, where the graph $G = (V, E)$ has edge weights $w \colon E \to \mathbb{R}_{\geqslant 0}$, a splitting operation consists of decreasing the weight on two edges $\{v, x\}$ and $\{v, y\}$ by some $\varepsilon > 0$ while increasing the weight on the edge $\{x, y\}$ by ε, and we want the weighted cut values $\mu_{G,w}(\{q\}, Q \setminus \{q\})$ to be invariant. We claim that this is achieved by Algorithm 3.2. We highlight that efficient weighted versions of other splitting-off results (than Theorem 3.10) have already been studied extensively (see [Fra92; Gab94; NI94; Ben95; NI96; BK98; NI00; BHKP08; LY13]), and our method is heavily inspired by an approach of Frank [Fra92].

In each iteration of the outer for-loop in Algorithm 3.2, we split off $\varepsilon \geqslant 0$ from $\{v, x\}$ and $\{v, y\}$, with ε chosen maximal so that all weights remain non-negative and the connectivities of interest are preserved. This choice of ε implies that once the outer for-loop terminated, there is no pair of edges incident to v from which a positive weight can be split off. Uniformly scaling all weights of this remaining graph to even integral weights (which we interpret as edge multiplicities) and employing Theorem 3.10, we can prove that there can only be a single edge with positive weight incident to v in the remaining graph, which we can thus safely delete without affecting connectivities within $V \setminus \{v\}$.

The following lemma summarizes the guarantees that we thereby obtain for Algorithm 3.2.

[7] If $q \in \{x, y\}$, then $c_2^q = \mu_{G,w}(\{q, v\}, (Q \setminus \{q\}) \cup \{x, y\})$ is the value of an infeasible cut problem (because both arguments of $\mu_{G,w}$ contain q), which we interpret as ∞.

Algorithm 3.2: Fractionally splitting off a single vertex.

Input : Graph $G = (V, E)$ with edge weights $w\colon E \to \mathbb{R}_{\geqslant 0}$, $Q \subseteq V$, $v \in V \setminus Q$.

foreach $x, y \in N_G(v) \coloneqq \{z \in V \setminus \{v\} \mid \{v, z\} \in E\}$, $x \neq y$ **do**

  foreach $q \in Q$ **do**

    Calculate the min cut sizes

$$c_1^q = \mu_{G,w}(\{q\}, Q \setminus \{q\}) \ , \quad c_2^q = \mu_{G,w}(\{q, v\}, (Q \setminus \{q\}) \cup \{x, y\}) \ ,$$ [7]

$$\text{and} \quad c_3^q = \mu_{G,w}(\{q, x, y\}, (Q \setminus \{q\}) \cup \{v\}) \ .$$

  Split off ε from $e_1 = \{v, x\}$ and $e_2 = \{v, y\}$, where

$$\varepsilon = \min_{q \in Q} \min \left\{ {}^{(c_2^q - c_1^q)}/_2, {}^{(c_3^q - c_1^q)}/_2, w(e_1), w(e_2) \right\} \ .$$

return *Modified graph G with vertex v deleted and modified weights w.*

Lemma 3.11. *Let $G = (V, E)$ be a graph with edge weights $w\colon E \to \mathbb{R}_{\geqslant 0}$, let $Q \subsetneq V$ and $v \in V \setminus Q$. On this input, Algorithm 3.2 returns, in running time dominated by $\mathcal{O}(|V|^3)$ many minimum s-t cut computations in contractions of (G, w), a graph $H = (V \setminus \{v\}, F)$ with edge weights $w_H\colon F \to \mathbb{R}_{\geqslant 0}$ such that*

(i) $\mu_{H,w_H}(\{q\}, Q \setminus \{q\}) = \mu_{G,w}(\{q\}, Q \setminus \{q\})$ for all $q \in Q$, and
(ii) $w_H(\delta_H(C \setminus \{v\})) \leqslant w(\delta_G(C))$ for all $C \subseteq V$.

Proof. Consider a splitting operation performed in Algorithm 3.2 on edges $e_1 = \{v, x\}$ and $e_2 = \{v, y\}$ for $x \neq y$, i.e., the weights on e_1 and e_2 are decreased by ε while the weight on $\{x, y\}$ is increased by ε. Such an operation changes the values of precisely those cuts that separate v from $\{x, y\}$, and their values all decrease by 2ε. Thus, cut values never increase in splitting steps, and neither do they when deleting v at the end of Algorithm 3.2, implying point (ii).

Moreover, observe that c_2^q and c_3^q computed in Algorithm 3.2 are precisely the minimum values of cuts separating q from $Q \setminus \{q\}$ and v from $\{x, y\}$. Thus, choosing $\varepsilon \leqslant \min \left\{ {}^{(c_2^q - c_1^q)}/_2, {}^{(c_3^q - c_1^q)}/_2 \right\}$ guarantees that the values of these cuts do not decrease below $\mu(\{q\}, Q \setminus \{q\})$. In other words, $\mu(\{q\}, Q \setminus \{q\})$ remains invariant under all splitting operations in Algorithm 3.2. Additionally, $\varepsilon \leqslant \min\{w(e_1), w(e_2)\}$ ensures that edge weights are always non-negative.

The extremal choice of ε implies that after the splitting operation is applied to a pair of edges (e_1, e_2), either one of $w(e_1)$ and $w(e_2)$ is zero, or there is a vertex $q \in Q$ and a cut $C \subseteq V$ with the following property: C separates q from $Q \setminus \{q\}$ as well as v from $\{x, y\}$, and $w(\delta(C)) = \mu(q, Q \setminus \{q\})$. We call such a cut *tight* for the pair (e_1, e_2), as any further reduction of $w(e_1)$ or $w(e_2)$ would reduce the value of C and hence also $\mu(q, Q \setminus \{q\})$. Observe that once a cut is tight for a pair of edges, it remains tight under all subsequent splitting operations.

Let (G', w') be the weighted graph obtained from (G, w) after performing all $\mathcal{O}(n^2)$ splitting operations in Algorithm 3.2. We claim that (G', w') has at most one edge with non-negative weight incident to v. If so, deleting v (and all its incident edges) from G' does not reduce $\mu(q, Q \setminus \{q\})$ for any $q \in Q$, hence the resulting graph has the desired properties. To see the claim, assume by contradiction that in (G', w'), there is more than one edge with positive weight incident to v. Then, there is a tight cut for each pair of such edges, implying that none of the edge weights can be reduced without reducing $\mu(q, Q \setminus \{q\})$ for some $q \in Q$. Now scale w' by an integer $M > 0$ such that all edge weights become even integers, and interpret these edge weights as edge multiplicities. Doing so, we obtain an Eulerian graph to which Theorem 3.10 is applicable, resulting in a pair of (potentially parallel) edges that can be split off from v without affecting $\mu(q, Q \setminus \{q\})$. But deleting an edge incident to v in this new graph corresponds to reducing the weight of the corresponding edge in G' by ${}^1/_M$. By assumption, the latter does reduce $\mu(q, Q \setminus \{q\})$, a contradiction.

Finally, observe that the values $\mu_{G,w}(A, B)$ needed in Algorithm 3.2 are infinite if $A \cap B \neq \emptyset$, and else they can be computed as the values of minimum s-t cuts in the contraction of G where A and B are contracted to vertices s and t, respectively. Three such values are computed for every triple (x, y, q) consisting of

$x, y \in N_G(v)$ with $x \neq y$ and $q \in Q$, and these $\mathcal{O}(|V|^3)$ many minimum s-t cut computations indeed dominate the overall running time. □

Applying Lemma 3.11 iteratively for all $v \in V \setminus Q$ reduces the graph G to the vertex set Q while maintaining the desired cut sizes, and thus immediately yields Theorem 3.9.

As indicated earlier in this section, there are different versions of splitting-off techniques. Some better known ones, for which strongly polynomial algorithms are already known, preserve pairwise connectivities among vertices in $Q \subseteq V$ instead of fulfilling the guarantees stated in Theorem 3.10. In Section 3.5, we show that under slightly stronger assumptions, these standard splitting-off techniques can also be used to obtain contraction distributions with the properties given in Theorem 3.4, with the necessary stronger assumptions leading to a weaker running time guarantee for Algorithm 3.1.

3.4 Further structural properties and their implications

Karger's mininum cut algorithm also provides a means of proving that a minimum cut problem has only polynomially many optimal solutions, and repeated applications of Karger's algorithm can find all these solutions with high probability. As discussed, analogous results cannot hold for CCMC problems. Note that in contrast to Karger's algorithm, our Contraction-Reduction Algorithm does not contract pairs of vertices until only two of them are left, but it stops early and terminates in an enumeration phase, solving reduced s-t cut problems. Following the spirit of the above-mentioned implications of Karger's algorithm, we obtain a structural result on these s-t cut instances. To state this result in full generality, we need the following congruency-constrained version of minimum s-t cut problems.

Congruency-Constrained Minimum *s-t* Cut (*s-t* CCMC): Let $G = (V, E)$ be an undirected graph with edge weights $w\colon E \to \mathbb{R}_{\geqslant 0}$ and let $\gamma\colon V \to \mathbb{Z}_{\geqslant 0}$. Let $m \in \mathbb{Z}_{>0}$ and $r \in \mathbb{Z}_{\geqslant 0}$, and let $s, t \in V$ be two distinct vertices. The task is to find a minimizer of

$$\min \left\{ w(\delta(C)) \;\middle|\; \{s\} \subseteq C \subseteq V \setminus \{t\},\ \sum_{v \in C} \gamma(v) \equiv r \pmod{m} \right\} .$$

Note that s-t CCMC problems can easily be modeled by CCMC problems if one allows to increase the modulus by an additional factor.[8] The subsequent theorem shows that the opposite reduction can be done, as well: Every CCMC problem can be reduced to polynomially many s-t CCMC problems with a smaller modulus.

Theorem 3.12. *Consider a CCMC problem on $G = (V, E)$ with constant modulus $m > 1$ and nonzero optimal value, and let $\kappa \geq 1$ be a constant. Then there is an efficient randomized algorithm returning $\mathrm{poly}(|V|)$ many s-t CCMC instances that (i) are defined on contractions of G with modified vertex multiplicities, and (ii) have a modulus that is a divisor of m strictly smaller than m, such that the following holds with high probability, where OPT denotes the optimal solution value of the initial CCMC problem: A cut $C \subsetneq V$, $C \neq \emptyset$, is a solution to the initial CCMC problem of value at most $\kappa \cdot \mathrm{OPT}$ if and only if C is a feasible solution of value at most $\kappa \cdot \mathrm{OPT}$ in one of the returned s-t CCMC instances.*

While the reduction of the modulus m obtained in the above theorem looks promising, the additional hurdle introduced by the transition to s-t CCMC instances seems to be substantial. We know efficient algorithms only for very special cases of s-t CCMC instances, one of them being the case of prime moduli, where a reduction to congruency-constrained submodular minimization is possible. This can be exploited to prove Theorem 3.2.

[8] An s-t CCMC instance $(G, w, \gamma, m, r, s, t)$ is captured by the CCMC instance $(G, w, \widehat{\gamma}, \widehat{m}, \widehat{r})$, where $\widehat{\gamma}(v) = 3 \cdot \gamma(v)$ for $v \notin \{s, t\}$, $\widehat{\gamma}(s) = 3 \cdot \gamma(s) + 1$, $\widehat{\gamma}(t) = 3 \cdot \gamma(t) + 2$, $\widehat{m} = 3 \cdot m$, and $\widehat{r} = 3 \cdot r + 1$, for example.

Proof of Theorem 3.2. Let $\mathcal{I}$ be the given CCMC instance with modulus m that is a product of two primes. Let C be an optimal solution of $\mathcal{I}$ and denote its value by OPT. If $\mathrm{OPT} = 0$, an optimal solution can be found easily by contracting the components and finding a union of them satisfying the congruency constraint. Else, an application of Theorem 3.12 to $\mathcal{I}$ with $\kappa = 1$ results in a polynomial number of s-t CCMC instances $\mathcal{I}_1, \dots, \mathcal{I}_\ell$. By Theorem 3.12, C is a feasible solution to at least one of these instances with high probability. On the other hand, Theorem 3.12 also asserts that the instances $\mathcal{I}_1, \dots, \mathcal{I}_\ell$ are defined on contractions of the initial graph G with weights induced by the initial weights, hence their optimal values are all at least OPT. Thus, we conclude that with high probability, C is an optimal solution to at least one of the instances $\mathcal{I}_1, \dots, \mathcal{I}_\ell$.

To conclude Theorem 3.2, it is enough to show that the instances $\mathcal{I}_1, \dots, \mathcal{I}_\ell$ can all be solved in polynomial time. To see this, fix an instance I_k. By Theorem 3.12, its modulus m' is a divisor of m that is strictly smaller than m. As m is a product of two primes, m' equals 1 or is a prime number. In the first case, $\mathcal{I}_k$ is an unconstrained minimum s-t cut problem, which can be solved efficiently. In the other case, $\mathcal{I}_k$ is a s-t CCMC instance with modulus equal to a prime number. This problem can easily be seen to be a special case of congruency-constrained submodular function minimization with prime modulus, which can be solved efficiently and to optimality as shown in [NSZ19]. □

Finally, if the modulus of the input problem in Theorem 3.12 is a prime number, the only feasible reduction of the modulus to one of its divisors is a reduction to modulus 1—and hence, to s-t cut problems without congruency constraint, which we exploit to prove Theorem 3.3.

Proof of Theorem 3.3. An application of Theorem 3.12 to the given instance with the given parameter κ results in polynomially many s-t CCMC instances. As the modulus of the given instance is a prime number, Theorem 3.12 implies that all the returned instances have modulus 1, i.e., they are in fact minimum s-t cut problems (without congruency constraint). Thus, Theorem 3.12 asserts that these instances have precisely the properties claimed by Theorem 3.3. □

3.4.1 Proof of Theorem 3.12

In this section, we show that Theorem 3.12 can be deduced from Algorithm 3.1. To this end, we add some further insights to the discussion of Algorithm 3.1 given in Section 3.2. In particular, observe that during a call of the algorithm on a CCMC instance $\mathcal{I} = (G, w, \gamma, m, r)$, the input graph G is repeatedly modified by random contractions, until the if-block of Algorithm 3.1 is reached (potentially only after several recursive calls to itself). Within the if-block, problems of the form

$$\min\{w(\delta(C)) \mid \emptyset \subsetneq C \subsetneq V,\ C \cap V_{\neq 0} = S\} \tag{3.7}$$

are solved for certain sets $S \subseteq V_{\neq 0}$. Problems of this type can be immediately reduced to minimum s-t cut problems in a further contracted graph: If $S \notin \{\emptyset, V_{\neq 0}\}$, then contract S and $V_{\neq 0} \setminus S$ to vertices s and t, respectively, and the problem in (3.7) is equivalent to the minimum s-t cut problem in the contracted graph. If $S = \emptyset$, contract $V_{\neq 0}$ to a vertex t, and a solution of (3.7) can be obtained by solving the minimum v-t cut problems for all $v \in V \setminus V_{\neq 0}$ and returning the solution of minimum value. Similarly, if $S = V_{\neq 0}$, contract $V_{\neq 0}$ to a vertex s, and the best solution among all solutions to the minimum s-v cut problems for $v \in V \setminus V_{\neq 0}$ solves (3.7).

Recall that these minimum s-t cut instances on contractions of G come with weights and vertex multiplicities induced by the original weights w and vertex multiplicities γ such that any cut C in the contracted graph has the same weight $w(\delta(C))$ and value $\gamma(C)$ as the corresponding cut in the initial graph. Hence, after imposing the original congruency constraint $\gamma(C) \equiv r \pmod m$, we obtain s-t CCMC instances which we call the instances *reached by* Algorithm 3.1. Note that these instances have modulus m equal to the input modulus, and not the potentially smaller modulus that is used in the call where the if-block is reached. The instances defined this way have several useful properties, and we will see that they are essentially the

instances claimed by Theorem 3.12. More precisely, the only missing property compared to the instances in Theorem 3.12 is that they still have modulus m. In Lemma 3.17, we will see that their structure allows for reducing the modulus to a divisor of m that is strictly smaller than m. Finally, analogous to the proof of Theorem 3.1, to obtain a sufficiently high success probability, we will run Algorithm 3.1 multiple times independently, and consider all s-t CCMC problems reached by these runs to construct the desired family of s-t CCMC problems as claimed in Theorem 3.12.

We start with two quick observations. First, note that the enumeration is done only if $V_{\not\equiv 0}$ reaches constant size, hence there are at most $\mathrm{poly}(|V|)$ many candidates $S \subseteq V_{\not\equiv 0}$ to be enumerated over. For every such choice of S, Algorithm 3.1 reaches either a single or linearly many s-t CCMC instances. Combining these arguments, we obtain Observation 3.13.

Observation 3.13. *The family of s-t CCMC instances reached by Algorithm 3.1 in a single run has size at most* $\mathrm{poly}(|V|)$.

Additionally, note that contractions and the transition from a global cut problem to an s-t cut problem for certain vertices s and t of the graph only reduce the set of feasible solutions, implying Observation 3.14.

Observation 3.14. *A cut C that is feasible for an s-t CCMC instance reached by Algorithm 3.1 is feasible for the input problem, and the weight of the cut is the same with respect to the two instances.*

For the other direction, we saw in the proof of Theorem 3.8 that for a suitable guess α of the optimal solution value, an optimal solution C of the input CCMC problem is not destroyed in the random contraction phase of Algorithm 3.1 (i.e., none of the contractions are applied to two vertices lying on different sides of C) with probability at least $1/\mathrm{poly}(|V|)$. The following lemma shows that with a slightly larger choice of the optimal solution value guess, this result extends to almost-minimum cuts.

Lemma 3.15. *Let $\mathcal{I}$ be a CCMC instance with optimal solution value denoted by* OPT*, let $\kappa \geqslant 1$, and let $\mathcal{F}$ be the family of s-t CCMC instances reached by Algorithm 3.1 on input $\mathcal{I}$ with optimal value guess $\alpha \geqslant \kappa \cdot \mathrm{OPT}$ and error parameter $\rho > 0$. Then, the probability that a feasible solution C of $\mathcal{I}$ with value at most $\kappa \cdot \mathrm{OPT}$ is also feasible for at least one of the instances in $\mathcal{F}$ is at least $1/\binom{|V|}{\lceil 4m/\rho \rceil}$.*

Proof. Fix a feasible solution C of $\mathcal{I}$ with $w(\delta(C)) \leqslant \kappa \cdot \mathrm{OPT}$. Let $m_i \in \mathbb{Z}_{\geq 0}$ and $s_i \in \mathbb{Z}_{\geq 0}$ denote the modulus and the size of $V_{\not\equiv 0}$, respectively, when the i^{th} contraction step is performed. By assumption, $\alpha \geqslant \kappa \cdot \mathrm{OPT}$, hence Theorem 3.4 with $c = 2m_i/\rho$ guarantees that

$$\Pr[\text{contraction } i \text{ is bad w.r.t. } C] \leqslant \frac{4m_i}{\rho \cdot |V_{\not\equiv 0}|} \leqslant \frac{k_i}{s_i} \ ,$$

where we define $k_i \coloneqq \left\lceil \frac{4m_i}{\rho} \right\rceil$. Following the very same reasoning as in the proof of Theorem 3.8, we get

$$\Pr\left[\begin{array}{c}\text{no random contraction is bad w.r.t. } C \\ \text{throughout a full run of Algorithm 3.1 on } \mathcal{I}\end{array}\right] \geqslant \frac{1}{\binom{|V|}{k_1}} = \frac{1}{\binom{|V|}{\lceil 4m/\rho \rceil}} \ .$$

Thus, with the above probability, C is still feasible once Algorithm 3.1 reaches the if-block. In this case, among all s-t CCMC instances reached by Algorithm 3.1 in the if-block, the cut C is feasible for at least the one instance reached when choosing $S = V_{\not\equiv 0} \cap C$. This concludes the proof. □

Besides preserving almost-minimum cuts with inverse polynomial probability, the s-t CCMC instances reached by Algorithm 3.1 have structured vertex multiplicities γ as stated by Lemma 3.16 below. This structure directly reflects the enumeration step performed in the if-block of Algorithm 3.1, where the choice of a suitable subset $S \subseteq V_{\not\equiv 0}$ guarantees that the remaining congruency constraint (note that at this stage, the modulus may have reduced to a divisor m_0 of m) is satisfied for every feasible solution of the resulting s-t cut problem. Showing that we must have $m_0 > 1$ under a mild assumption on the algorithm parameters α and ρ, we obtain the following.

Lemma 3.16. *Let $\mathcal{I} = (G, w, \gamma, m, r)$ be a CCMC instance such that $m > 1$ and denote its optimal value by* OPT. *Let $\mathcal{I}' = (G', w', \gamma', m, r, s, t)$ be an s-t CCMC instance reached by Algorithm 3.1 in a call on $\mathcal{I}$ with error parameter $\rho > 0$ and optimal solution guess $\alpha \geqslant 0$ such that $\rho\alpha <$ OPT. Then, there exists a divisor m_0 of m with $m_0 > 1$ such that $\gamma(v) \equiv 0 \pmod{m_0}$ for all vertices v of G' with $v \notin \{s, t\}$.*

Proof. When processing a CCMC instance $\mathcal{I}$, Algorithm 3.1 starts with repeatedly doing random contractions and reduction steps, with each of the latter ones issuing a recursive call to Algorithm 3.1 using a modulus that is a divisor of the input modulus m. When there are only few vertices in $V_{\not\equiv 0}$ left, an enumeration over subsets of $V_{\not\equiv 0}$ is performed. Let m_0 be the modulus used when the if-block is started, and let $S \subseteq V_{\not\equiv 0}$ be the subset used in the enumeration step to reach $\mathcal{I}'$. If $S \in \{\emptyset, V_{\not\equiv 0}\}$, then the set $V_{\not\equiv 0}$ is contracted and used in $\mathcal{I}'$ as vertex s or t. In the other case, S and $V_{\not\equiv 0} \setminus S$ get contracted to the vertices s and t. Thus, in both cases, a vertex v of the contracted graph different from s and t lies in $V \setminus V_{\not\equiv 0}$, so by definition of $V_{\not\equiv 0}$, it satisfies $\gamma(v) \equiv 0 \pmod{m_0}$.

Consequently, it remains to prove that $m_0 > 1$. To this end, assume $m_0 = 1$ and consider the reduction step that leads to the recursive call of Algorithm 3.1 with modulus 1. This reduction step can only be performed if there is a reduction family $\mathcal{R}(\beta, q)$ for $\beta = \frac{\rho\alpha}{m}$ and some $q \in [m-1]$ with $\gcd(m, q) = 1$, where the latter condition comes from the assumption that the modulus is reduced to 1. By definition of a reduction family, we have $|\mathcal{R}(\beta, q)| = 2m - 1$, i.e., $\mathcal{R}(\beta, q) = \{R_1, \ldots, R_{2m-1}\}$, and every set R_i contains one vertex u_i with $\gamma(u_i) \equiv q \pmod{m}$, while all other vertices have γ-value $0 \pmod{m}$. Furthermore, the vertices $u_1, \ldots, u_{2m-1}$ are all distinct. Let $k \in [m]$ be such that $qk \equiv r \pmod{m}$ (such a k exists as $\gcd(m, q) = 1$), and let $C \coloneqq R_1 \cup \ldots \cup R_k$. Observe that C is feasible for $\mathcal{I}$, as $\emptyset \neq C \subsetneq V$ because $u_1 \in C$ and $u_{2m-1} \notin C$, and $\gamma(C) = \sum_{i=1}^{k} \gamma(u_i) \equiv k \cdot q \equiv r \pmod{m}$. But

$$w(\delta(C)) \leqslant \sum_{i \in [k]} w(\delta(R_i)) \leqslant k \cdot \beta \leqslant \rho\alpha < \text{OPT} \ ,$$

contradicting that OPT is the optimal solution value of $\mathcal{I}$. Thus indeed, we must have $m_0 > 1$. □

We remark that the assumptions of Lemma 3.16 imply OPT > 0, which is inevitable. Indeed, consider an instance only consisting of isolated vertices with $\gamma(v) \equiv 1 \pmod{m}$ and congruency constraint $\gamma(C) \equiv 1 \pmod{m}$. As there are no edges at all, no contractions can be applied, and an efficient enumeration is not possible either, leaving only a reduction to modulus 1—making an argument as in the previous proof impossible. This also explains why we have to assume that the optimal solution value is nonzero in Theorem 3.12.

Furthermore, note that in the proof of Lemma 3.16, we show that the modulus m_0 that is used when reaching the enumeration phase satisfies $m_0 > 1$ if $\rho\alpha <$ OPT. Equivalently, Algorithm 3.1 never reduces to global minimum cut instances for input moduli $m > 1$. In particular, no reduction steps are performed at all if m is a prime, hence in this case and under the given condition on ρ and α, we see that Algorithm 3.1 is exact.

Finally, it is easy to observe that the structured CCMC instances as specified in Lemma 3.16 can be transformed to equivalent s-t CCMC instances with strictly smaller modulus.

Lemma 3.17. *Let $\mathcal{I} = (G, w, \gamma, m, r, s, t)$ be an s-t CCMC instance on a graph $G = (V, E)$ such that there exists a divisor m_0 of m satisfying $\gamma(v) \equiv 0 \pmod{m_0}$ for all vertices $v \in V \setminus \{s, t\}$. Then, we can efficiently obtain an s-t CCMC instance $\mathcal{I}'$ on the same edge-weighted graph G with modulus m/m_0 such that a cut $C \subseteq V$ is feasible for $\mathcal{I}$ if and only if C is feasible for $\mathcal{I}'$.*

Proof. If $\mathcal{I}$ is infeasible, there is nothing to show (and feasibility can be checked efficiently). In the other case, we must have $\gamma(s) \equiv r \pmod{m_0}$, hence $r' \coloneqq \frac{r - (\gamma(s) \bmod m_0)}{m_0} \in \mathbb{Z}$, where for $v \in V$, $(\gamma(v) \bmod m_0)$ denotes the smallest non-negative integer k such that $\gamma(v) \equiv k \pmod{m_0}$. Moreover, let $m' \coloneqq \frac{m}{m_0} \in \mathbb{Z}$, and define $\gamma' \colon V \to \mathbb{Z}_{\geqslant 0}$ by

$$\gamma'(v) = \frac{\gamma(v) - (\gamma(v) \bmod m_0)}{m_0}$$

for all $v \in V$. We claim that $\mathcal{I}' = (G, w, \gamma', m', r', s, t)$ has the desired properties.

To see this, let $C \subseteq V$ be a feasible solution for $\mathcal{I}$. Then $\gamma(C) \equiv r \pmod{m}$, and thus by definition of γ' and as $\gamma(v) \equiv 0 \pmod{m_0}$ for all $v \in C \setminus \{s\}$,

$$\begin{aligned} m_0 \cdot \gamma'(C) &= \gamma(C) - (\gamma(s) \bmod m_0) \\ &\equiv r - (\gamma(s) \bmod m_0) \\ &\equiv m_0 \cdot r' \qquad\qquad \pmod{m} \ , \end{aligned}$$

and thus, after division by m_0, $\gamma'(C) \equiv r' \pmod{m'}$, so C is feasible for $\mathcal{I}'$. For the other direction, let $C \subseteq V$ be feasible for $\mathcal{I}'$, i.e., $\gamma'(C) \equiv r' \pmod{m'}$. After multiplication by m_0, the latter gives $m_0 \cdot \gamma'(C) \equiv r - (\gamma(s) \bmod m_0) \pmod{m}$, hence

$$\gamma(C) = m_0 \cdot \gamma'(C) + (\gamma(s) \bmod m_0) \equiv r \pmod{m} \ ,$$

so C is feasible for $\mathcal{I}$. Finally, observe that $\mathcal{I}'$ can obviously be obtained from $\mathcal{I}$ efficiently. □

From the above findings, we can now complete a proof of Theorem 3.12.

Proof of Theorem 3.12. Let $\mathcal{I}$ be the given CCMC instance. For all polynomially many values

$$\alpha \in \left\{ \kappa \cdot 2^j \cdot w_{\min} \,\middle|\, 0 \leqslant j \leqslant \lceil \log_2(w_{\text{tot}}/w_{\min}) \rceil \right\} \ ,$$

we run Algorithm 3.1 on $\mathcal{I}$ with $\rho = \frac{1}{2\kappa}$ for $\binom{|V|}{\lceil 8\kappa m \rceil} \log |V|$ times independently, obtaining families $\mathcal{F}_i^\alpha$ of *s-t* CCMC instances reached by the algorithm with guess α in the i^{th} run. By Observation 3.14, any cut C that is a solution to one of the problems in $\bigcup_{\alpha,i} \mathcal{F}_i^\alpha$ is a solution to $\mathcal{I}$, as well, and the solutions have the same value.

For the other direction, fix a solution C of $\mathcal{I}$ with value at most $\kappa \cdot \mathrm{OPT}$, and consider the families $\mathcal{F}_i^\alpha$ obtained from a run of Algorithm 3.1 with $\alpha \in [\kappa \cdot \mathrm{OPT}, 2\kappa \cdot \mathrm{OPT})$. By Lemma 3.15, for each of these $\binom{|V|}{\lceil 8\kappa m \rceil} \log |V|$ many families $\mathcal{F}_i^\alpha$, the following is true: With probability $1/\binom{|V|}{\lceil 8\kappa m \rceil}$, it contains an *s-t* CCMC instance for which C is feasible. Consequently, with probability at least

$$1 - \left(1 - \frac{1}{\binom{|V|}{\lceil 8\kappa m \rceil}}\right)^{\binom{|V|}{\lceil 8\kappa m \rceil} \cdot \log |V|} \geqslant 1 - \mathrm{e}^{-\log |V|} = 1 - \frac{1}{|V|} \ ,$$

the cut C is feasible for at least one instance in $\bigcup_i \mathcal{F}_i^\alpha$. As by Observation 3.13, every family $\mathcal{F}_i^\alpha$ has polynomial size, and because we generated polynomially many such families, $\mathcal{F} \coloneqq \bigcup_{i,\alpha} \mathcal{F}_i^\alpha$ has polynomial size. Thus, the *s-t* CCMC instances in $\mathcal{F}$ have all properties stated in Theorem 3.12 except for the fact that their modulus is still m. But, by Lemma 3.16 (note that $\rho\alpha < \mathrm{OPT}$ as $\alpha < 2\kappa \cdot \mathrm{OPT}$), the instances in $\mathcal{F}$ satisfy the assumptions of Lemma 3.17, and thus can be transformed to equivalent instances on the same edge-weighted graphs with moduli that are divisors of m and strictly smaller than m, as desired. □

3.5 Weaker contraction distributions from standard splitting techniques

The proof of Theorem 3.4 given in Section 3.3 is built on an algorithmic version of the splitting-off theorem by Lovász (Theorem 3.10), which allows for reducing a graph $G = (V, E)$ while preserving the sizes of minimum cuts separating the sets $\{q\}$ and $Q \setminus \{q\}$ for a fixed $Q \subseteq V$ and all $q \in Q$. This splitting-off version is much less known and studied than the most common variant which preserves all pairwise connectivities among vertices in a subset $Q \subseteq V$. In this section, we show that these better-known splitting-off versions, for which strongly polynomial algorithms are already known, also allow for constructing contraction distributions for our purposes that are only slightly weaker than those obtained in Section 3.3.

Let us start by stating one of the above-mentioned standard versions of splitting-off results, namely a theorem of Frank [Fra92]. We write $\mu_{G,w}(s,t)$ instead of $\mu_{G,w}(\{s\},\{t\})$ for the value of a minimum cut separating distinct vertices s and t.

Theorem 3.18 (Frank [Fra92])**.**[9] *Let $G = (V, E)$ be a graph with edge weights $w\colon E \to \mathbb{R}_{\geqslant 0}$, and let $Q \subseteq V$. Then there is a strongly polynomial time algorithm to obtain a graph $H = (Q, F)$ and edge weights $w_H\colon F \to \mathbb{R}_{\geqslant 0}$ satisfying*

(i) $\mu_{G,w}(s,t) = \mu_{H,w_H}(s,t)$ for all $s, t \in Q$ with $s \neq t$, and
(ii) $w_H(\delta_H(C \cap Q)) \leqslant w(\delta_G(C))$ for all $C \subseteq V$.

For the rest of this section, let us fix a CCMC instance $\mathcal{I} = (G, w, \gamma, m, r)$ with graph $G = (V, E)$, and let $H = (V_{\neq 0}, F)$ with edge weights $w_H\colon F \to \mathbb{R}_{\geqslant 0}$ be a graph obtained by applying Theorem 3.18 to (G, w) with $Q = V_{\neq 0}$. Moreover, let $\mathcal{D}$ be the distribution over vertex pairs $\{u, v\} \subseteq V$ given by choosing $\{u, v\} \in F$ with probability proportional to $w_H(\{u, v\})$. For this distribution $\mathcal{D}$, we show the following theorem, which is a weaker version of Theorem 3.4.

Theorem 3.19. *Let $\alpha \geqslant 0$ and $c > 0$ with $\sum_{v \in V_{\neq 0}} \nu(\{v\}) > \frac{4\alpha}{c} \cdot |V_{\neq 0}|$. Then, for any feasible solution C of the instance $\mathcal{I}$ with $w(\delta(C)) \leqslant \alpha$, the distribution $\mathcal{D}$ satisfies $\Pr_{\{u,v\}\sim\mathcal{D}}\big[|\{u, v\} \cap C| = 1\big] \leqslant c/|V_{\neq 0}|$.*

Note that compared to Theorem 3.4, the assumption in Theorem 3.19 is stronger by a factor of 2. This implies that the analogue of Algorithm 3.1 which contracts vertices based on the distribution $\mathcal{D}$ obtained in this section can perform contraction steps only if stronger assumptions are satisfied, and thus has to fall back on reduction or enumeration steps earlier. This leads to an increase in running time.

The proof of Theorem 3.19 is similar to the one of Theorem 3.4. There, we could exploit that by construction, $w_H(F) = \frac{1}{2}\sum_{v \in V_{\neq 0}} \nu(\{v\})$. This is no longer true in the current alternative setting, but we can instead use the following bound.

Theorem 3.20. *We have $w_H(F) \geqslant \frac{1}{4}\sum_{v \in V_{\neq 0}} \nu(\{v\})$.*

This indeed implies Theorem 3.19 immediately.

Proof of Theorem 3.19. If C is a solution of $\mathcal{I}$ with $w(\delta(C)) \leqslant \alpha$, then by the choice of $\mathcal{D}$,

$$\Pr_{\{u,v\}\sim\mathcal{D}}\big[|\{u, v\} \cap C| = 1\big] = \frac{w_H(\delta_H(C \cap V_{\neq 0}))}{w_H(F)} \leqslant \frac{4 \cdot w(\delta_G(C))}{\sum_{v \in V_{\neq 0}} \nu(\{v\})} \leqslant \frac{c}{|V_{\neq 0}|} \ ,$$

where the first inequality is due to Theorem 3.18 (ii) and Theorem 3.20, and the second one follows from $w(\delta_G(C)) \leqslant \alpha$ and the assumption that $\sum_{v \in V_{\neq 0}} \nu(\{v\}) > \frac{4\alpha}{c} \cdot |V_{\neq 0}|$. □

It thus remains to show Theorem 3.20. To this end, we use the notion of a Gomory-Hu tree [GH61] (see also [KV18; Sch03] for two excellent exhibitions of the topic). More precisely, we consider a Gomory-Hu tree $T = (V_{\neq 0}, L)$ for $V_{\neq 0}$ in G. This is a spanning tree over $V_{\neq 0}$, where the edges $L \subseteq \binom{V_{\neq 0}}{2}$ of the spanning tree are not necessarily edges of G. Moreover, the edges L of T have weights $w_T\colon L \to \mathbb{R}_{\geqslant 0}$, such that

$$w_T(\{s, t\}) = \mu_{G,w}(s, t) = \nu(C_{s,t}) \quad \forall \{s, t\} \in L \ , \tag{3.8}$$

where $C_{s,t} \subseteq V_{\neq 0}$ are all vertices of the graph $(V_{\neq 0}, L \setminus \{s, t\})$ in the connected component that contains s.[10] To prove Theorem 3.20, the next two lemmas relate both the w_H-weight of F and the values $\nu(\{v\})$, respectively, to weights on the Gomory-Hu tree T, which then allows us to compare them.

Lemma 3.21. *For all $v \in V_{\neq 0}$, we have $\nu(\{v\}) \leqslant w_T(\delta_T(v))$.*

[9] Frank shows how to get (i). However, (ii) is immediate from the fact that Frank's algorithm performs classical splitting-off operations. More precisely, the method repeatedly considers a pair of edges $\{w, v\}, \{w, u\}$ sharing one endpoint w, and reduces their weights by some $\varepsilon > 0$, while increasing the weight of $\{v, u\}$ by ε (if needed, a new edge $\{v, u\}$ is introduced). Clearly, this way of modifying weights will never increase the value of any cut.

[10] A more classical notion of Gomory-Hu trees considers a spanning tree over *all* vertices of G. However, the generalized version we need, with a tree only over a subset of the vertices, can be readily derived from the more classical version, and often follows as a byproduct when building classical Gomory-Hu trees (see proof of Theorem 15.14 in [Sch03] for a formal proof).

Proof. Let $k \coloneqq |\delta_T(v)|$, and let $t_1, \dots, t_k \in V_{\not\equiv 0}$ be the neighbors of v in T. The desired result holds due to

$$\nu(\{v\}) \leqslant \sum_{i=1}^{k} \nu(C_{v,t_i}) = \sum_{i=1}^{k} w_T(\{v, t_i\}) = w_T(\delta_T(v)) \ ,$$

where the inequality holds because a cut in G that separates v from $V_{\not\equiv 0} \setminus \{v\}$ can be obtained by removing, for each $i \in [k]$, the minimum cut in G that separates C_{v,t_i} from $V_{\not\equiv 0} \setminus C_{v,t_i}$; moreover, the first equality follows from (3.8). □

Lemma 3.22. *We have* $w_T(T) \leqslant 2w_H(F)$.

Proof. We start by showing that

$$w_H(\delta_H(v)) \geqslant w_T(f) \quad \forall f \in \delta_T(v) \ . \tag{3.9}$$

The above holds because for any $f = \{u, v\} \in \delta_T(v)$, we have

$$w_T(f) = \mu_{G,w}(u, v) = \mu_{H,w_H}(u, v) \leqslant w_H(\delta_H(v)) \ ,$$

where the first equality follows from (3.8), the second one from Theorem 3.18 (i), and the inequality holds because $w_H(\delta_H(v))$ is the value of the singleton cut $\{v\}$ in H, which is a v-u cut, and $\mu_{H,w_H}(u, v)$ is the value of the smallest v-u cut in H.

Finally, to show the lemma, we choose an arbitrary vertex $r \in V_{\not\equiv 0}$, and direct all edges L of $T = (V_{\not\equiv 0}, L)$ away from r, to obtain an r-arborescence. This arborescence can be interpreted as a bijection between L and $V_{\not\equiv 0} \setminus \{r\}$, where an edge $f \in L$ gets assigned to the vertex in $V_{\not\equiv 0}$ to which it points to. Now for each edge $\{u, v\} \in L$, we have $w_T(f) \leq w_H(\delta_H(v))$ by (3.9), where v is the vertex to which f points to. Hence, by summing over all edges in T, we obtain the first inequality in the following relation, which proves the statement:

$$w_T(L) \leqslant \sum_{v \in V_{\not\equiv 0} \setminus \{r\}} w_H(\delta_H(v)) \leqslant \sum_{v \in V_{\not\equiv 0}} w_H(\delta_H(v)) = 2w_H(F) \ ,$$

where the equality is the classical relation that the sum of weighted degrees is equal to twice the total weight. □

With the above two statements at hand, Theorem 3.20 is a straightforward consequence.

Proof of Theorem 3.20. We have

$$w_H(F) \geqslant \frac{1}{2} w_T(T) = \frac{1}{4} \sum_{v \in V_{\not\equiv 0}} w_T(\delta_T(v)) \geqslant \frac{1}{4} \sum_{v \in V_{\not\equiv 0}} \nu(\{v\}) \ ,$$

where the first inequality follows from Lemma 3.22, the equality holds because the sum of weighted degrees is twice the total weight, and the second inequality holds due to Lemma 3.21. □

Chapter 4

A new dynamic programming approach for spanning trees with chain constraints and beyond

4.1 Introduction

Given a graph $G = (V, E)$ and edge costs $c\colon E \to \mathbb{R}_{\geqslant 0}$, the problem of finding a minimum cost spanning tree (MST) in G with respect to c is one of the most classical network design problems. A variety of applications in areas like chip design, vehicle routing, and telecommunication networks triggered interest in constrained spanning tree problems. Moreover, such problems are regularly used as building blocks in the design of approximation algorithms. In particular, many approaches used in recent progress on the Traveling Salesman Problem (TSP) and its path version have a constrained spanning tree problem as a key component.

The arguably most classical example of a constrained spanning tree problem is the minimum bounded degree spanning tree problem (MBDST). Here, the goal is to find a spanning tree $T \subseteq E$ in G of minimum cost subject to T satisfying a degree constraint $|T \cap \delta(v)| \leqslant d(v)$ at every vertex v, where $d\colon V \to \mathbb{Z}_{>0}$ are given degree bounds. Already just finding a feasible solution for MBDST can easily be seen to be NP-hard, even in the special case where $d(v) = 2$ for all v, as this captures the Hamiltonian path problem. This is typical for most constrained spanning tree problems. The focus has therefore been on approximation algorithms that allow for a slight violation of the additional constraints. This led to algorithms with various trade-offs between cost and constraint violation. After a series of papers with progress on the approximation guarantees (see [KR00; KR03; Goe06; CRRT09a; CRRT09b] and references therein), an essentially best possible approximation algorithm for MBDST was given by Singh and Lau [SL07]. Using iterative relaxation, they return a spanning tree violating each degree constraint by at most 1 unit, and of cost no more than that of an optimal solution not violating the degree constraints. Bansal, Khandekar, and Nagarajan [BKN09] presented an elegant generalization of this result to upper bounds on the number of edges picked in a family of arbitrary edge sets $E_1, \dots, E_k \subseteq E$. More precisely, they show that a spanning tree violating each constraint by at most $\max_{e \in E} |\{i \in [k] \mid e \in E_i\}| - 1$ and with cost no more than that of an optimal solution can be found. If each edge is only contained in a constant number of constraints, this still leads to a constraint violation by only an additive constant. Whereas iterative relaxation is undoubtedly a very strong tool to find constrained spanning trees, it is difficult to obtain constraint violations of at most a constant (either additively or multiplicatively) through this technique when edges can be in a super-constant number of constraints (see [Zen12] for one rare example of this type).

However, constrained spanning tree problems appearing in the design of approximation algorithms, especially within problems related to TSP, are often of this type. (When referring to TSP and its variants, we always assume that the involved lengths are metric.) For example, Asadpour, Goemans, Madry, Oveis Gharan, and Saberi [AGMOS10] established a beautiful connection between the asymmetric TSP and so-called thin trees, which are trees with constraints on all cut sets. More precisely, if there is a constant c such that for any $k \in \mathbb{Z}_{>0}$, one can efficiently find in any k-edge-connected graph $G = (V, E)$ a spanning tree $T \subseteq E$

with $|T \cap \delta(S)| \leqslant {}^{c}/_{k} \cdot |\delta(S)|$ for all $S \subseteq V$, then this can be transformed into an $\mathcal{O}(1)$-approximation for asymmetric TSP. Such trees are sometimes referred to as constantly-thin trees. The existence of a weaker version of constantly-thin trees was conjectured by Goddyn [God04] and remains open. We highlight that very recently, Svensson, Tarnawski, and Végh [STV18] obtained a 5500-approximation for asymmetric TSP through different techniques, with the approximation factor later improved to $22 + \varepsilon$ by Traub and Vygen [TV20]. Still, different ideas might be needed to obtain considerably smaller factors in the region of those that are known for the symmetric version. Finding constantly-thin spanning trees may be one path to advance on this question. Moreover, especially for Path TSP, where the task is to find a shortest Hamiltonian s-t path in a complete graph with metric lengths, finding spanning trees with various additional constraints/properties has been crucial in recent progress [AKS15; Seb13; Vyg16; GV16; SZ16; TV18; Zen19]. Interestingly, the type of tree properties considered for Path TSP are often on the edges contained in a family of s-t cuts that form a chain.[1] More generally, the shortest connected T-join problem, which generalizes both Path TSP and classical TSP, naturally leads to a laminar family of cuts to be considered [CFG15].[2] The appearance of cut families with laminar or chain structure in this context stems from the use of combinatorial uncrossing arguments, which are ubiquitous in the context of TSP, and is thus not surprising. Clearly, when constraints are imposed on the edges in a family of cuts that are laminar, or even just a chain, then edges can appear in a large number of constraints.

The arguably most canonical constrained spanning tree problem with constraints on a laminar family of cuts is when there are upper bounds on the number of edges in each cut. This setting was considered by Bansal, Khandekar, Könemann, Nagarajan, and Peis [BKKNP13], who designed an iterative relaxation approach for returning a spanning tree violating each constraint by at most $\mathcal{O}(\log |V|)$ units and being of cost no more than the cost of an optimal solution not violating the constraints. As later shown by Olver and Zenklusen [OZ18], this is almost optimal because an additive violation of ${}^{c \log |V|}/_{\log \log |V|}$ units, for some constant c, cannot be achieved unless $\mathrm{P} = \mathrm{NP}$. It remains open whether $\mathcal{O}(1)$-multiplicative violations are possible.

In summary, constrained spanning tree problems where edges can appear in a large number of constraints are still badly understood, and new approaches and techniques are needed.

The goal of this chapter is to introduce a versatile dynamic programming type approach to deal with a variety of constraint types on laminar cut families of small width, with applications to chain-constrained spanning trees, Path TSP and beyond. Dynamic programming did not play a crucial role in the above-mentioned problems until a very recent breakthrough result by Traub and Vygen [TV18] in the context of Path TSP, which inspired this work. A key new technical ingredient in our approach is to introduce a generalized form of dynamic programming, where the value of a table entry does not only depend on the values of previous table entries, as it is usually the case, but also on a fixed representative solution saved together with each table entry. This leads to the peculiar situation that it is hard to define upfront the solution set over which our dynamic program optimizes. However, we can show that it optimizes over a relaxation of the problems we are interested in, and returns solutions with well-defined properties to be exploited later on, which is all we need. For chain-constrained problems, our dynamic program can be leveraged to return a fractional point in the spanning tree polytope, which can then be rounded to an actual spanning tree. We show that good spanning trees can be obtained by using negatively correlated rounding procedures together with an alteration procedure that may be of independent interest, and which we therefore present in a more general context.

[1] Throughout this chapter, a *cut* of a vertex set V is a nonempty set $S \subsetneq V$. An edge e *lies in* a cut S if $e \in \delta(S)$.

[2] For some even cardinality vertex set $T \subseteq V$ in a graph $G = (V, E)$, a T-join is an edge set $U \subseteq E$ such that the vertices of odd degree in the subgraph (V, U) are precisely T. Moreover, in the shortest connected T-join problem, one is allowed to choose as U a multiset of edges in E.

4.1.1 Our results

Here, we provide a summary of the results that we obtain by combining our dynamic programming approach with various other techniques. We start with a natural special case of laminarly constrained spanning trees that has been studied previously, namely the *minimum chain-constrained spanning tree* problem (MCCST), where upper bounds are imposed on the number of edges that can be chosen in a family of cuts that form a chain. Opposed to previous results, we also allow for lower bounds on the number of edges in the cuts, which can be handled with our methods without additional complications.

Minimum Chain-Constrained Spanning Tree Problem (MCCST): Let $G = (V, E)$ be a graph with edge costs $c\colon E \to \mathbb{R}_{\geqslant 0}$, and let $\emptyset \subsetneq S_1 \subsetneq S_2 \subsetneq \ldots \subsetneq S_k \subsetneq V$ and $a_1, \ldots, a_k, b_1, \ldots, b_k \in \mathbb{Z}_{>0}$. Find a spanning tree $T \subseteq E$ minimizing $c(T) \coloneqq \sum_{e \in T} c(e)$ among all trees satisfying

$$a_i \leqslant |T \cap \delta(S_i)| \leqslant b_i \quad \text{for all } i \in [k] \coloneqq \{1, \ldots, k\}.$$

For $\alpha, \beta \geqslant 1$, we say that an algorithm returning a spanning tree T is an (α, β)-*approximation* for MCCST, if $\frac{1}{\beta} \cdot a_i \leqslant |T \cap \delta(S_i)| \leqslant \beta \cdot b_i$ for all $i \in [k]$, and $c(T) \leqslant \alpha \cdot c(\mathrm{OPT})$, where OPT is a spanning tree of minimum cost among all spanning trees not violating the chain constraints. For MCCST without lower bounds, i.e., $a_1 = \ldots = a_k = 0$, Linhares and Swamy [LS16] recently presented an efficient $(\frac{\lambda}{\lambda - 1}, 9\lambda)$-approximation for any $\lambda > 1$ by extending a prior approach of Olver and Zenklusen [OZ18] that did not handle costs. Our main result is a quasi-polynomial algorithm for MCCST (with lower bounds) with essentially best possible guarantees.

Theorem 4.1. *For every $\varepsilon > 0$, there is a randomized $(1, 1 + \varepsilon)$-approximation algorithm for MCCST with running time $|V|^{\mathcal{O}(\log |V| / \varepsilon^2)}$.*

The approximation guarantee is essentially best possible in the sense that finding a tree that fulfills all chain constraints is NP-hard as shown in [OZ18]. Our randomized algorithm returns a $(1, 1 + \varepsilon)$-approximation with high probability, and can also be transformed into a Las Vegas algorithm. Moreover, Theorem 4.1 gives hopes that such best possible guarantees may be achievable with an efficient procedure. We prove the theorem through a combination of our new dynamic programming approach, which we will introduce in this context, together with negatively correlated rounding and an alteration step to improve the value of the final solution. As we discuss in Section 4.4, in this context, the alteration step we use could also be replaced by a technique introduced by Linhares and Swamy [LS16].

It turns out that with some modifications, the technical insights outlined above are enough to push our results beyond pure chain constraints towards the more general problem of *minimum laminarly constrained spanning trees* (MLCST), which is defined as follows.

Minimum Laminarly Constrained Spanning Tree Problem (MLCST): Let $G = (V, E)$ be a graph with edge costs $c\colon E \to \mathbb{R}_{\geqslant 0}$, let $\mathcal{L} \subseteq 2^V \setminus \{\emptyset, V\}$ be a laminar family, and $a_S, b_S \in \mathbb{Z}_{>0}$ for $S \in \mathcal{L}$. Find a spanning tree $T \subseteq E$ minimizing $c(T) \coloneqq \sum_{e \in T} c(e)$ among all trees satisfying

$$a_S \leqslant |T \cap \delta(S)| \leqslant b_S \quad \text{for all } S \in \mathcal{L}.$$

Note that if $\mathcal{L}$ contains precisely all singletons, then the above problem setting reduces to the minimum bounded degree spanning tree problem (MBDST) mentioned in the introduction. From a structural point of view, constraint types in the special cases MCCST and MBDST are "orthogonal" in the sense that the role of $\mathcal{L}$ is taken by a chain in one case and by an antichain in the other. Currently, there is no efficient approach covering both cases. For a step towards MLCST using our dynamic programming framework, we parametrize laminar families by their *width*, which is the smallest integer k such that the laminar family does not contain any $k + 1$ disjoint sets. Using this notion, we can generalize Theorem 4.1 to obtain the following result.

Theorem 4.2. *For every $\varepsilon > 0$, there is a randomized $(1, 1+\varepsilon)$-approximation algorithm for MLCST with running time $|V|^{\mathcal{O}(k \log |V|/\varepsilon^2)}$, where k is the width of the laminar family $\mathcal{L}$.*

Observe that the running time in the above result depends exponentially on the width of the laminar family. For width k up to the order $\mathcal{O}(\log |V|)$, we thus still achieve quasi-polynomial running time. Unfortunately, the exponential dependence on k seems to be intrinsic to our approach, and most likely, new ideas are required to overcome this.

Our dynamic programming approach is very versatile in terms of constraint types that can be handled. To highlight this fact, we show how it can be employed to replicate a recent result of Traub and Vygen [TV18] in the context of Path TSP.

Shortest Hamiltonian s-t Path Problem (Path TSP): Let $G = (V, E)$ be a complete graph with metric edge lengths $\ell\colon E \to \mathbb{R}_{\geqslant 0}$, and let $s, t \in V$ be two distinct vertices. Find a path $P \subseteq E$ minimizing $\ell(P) \coloneqq \sum_{e \in P} \ell(e)$ among all Hamiltonian s-t paths in G.

Leveraging our dynamic programming ideas, we obtain the following theorem.

Theorem 4.3. *For every $\varepsilon > 0$, there is a $(1.5 + \varepsilon)$-approximation algorithm for Path TSP with running time $|V|^{\mathcal{O}(1/\varepsilon)}$.*

The above result matches the algorithm by Traub and Vygen [TV18] in terms of approximation ratios, but provides a weaker running time guarantee, as Traub and Vygen are able to achieve an exponential dependence on $\log(1/\varepsilon)$. Our approach, on the other hand, avoids recursive calls to a dynamic program as used in [TV18]. Both results almost match the currently best known approximation guarantee that is just below 1.5 and can be obtained by combining the result of Karlin, Klein, and Gharan [KKG21], who provide an approximation factor of $3/2 - \delta$ for some $\delta > 10^{-36}$ for TSP, and the result of Traub, Vygen, and Zenklusen [TVZ20] that reduces Path TSP to TSP at an arbitrarily small loss in the approximation factor. While being rather involved, this result shows that the approximation factor of 1.5 (which was achieved with a very neat approach by Zenklusen [Zen19]) can be surpassed.

To obtain Theorem 4.3, our dynamic programming approach can be used to obtain a good spanning tree T for a Christofides-type algorithm, which starts with a spanning tree T and does parity correction in a second step by adding further edges. As observed by An, Kleinberg, and Shmoys [AKS15], it turns out that when following Wolsey's analysis [Wol80] of Christofides algorithm, then the number of edges of T in a chain of cuts crucially impacts the approximation guarantee. This naturally leads to a constrained spanning tree problem for finding T. As for MCCST, the constraints appearing in this problem are on a chain of cuts. However, they are not simply upper and lower bounds—ideally, we would like to impose parity constraints, requiring that each cut is crossed by an odd number of edges. Unfortunately, this leads to NP-hard problems, and we thus use a proxy for parity constraints that is good enough for finding short s-t paths. Again, we hope that this way of dealing with parity constraints, which naturally appear in the context of TSP, may find broader applications.

While techniques from MCCST have implications in Path TSP, their generalization to MLCST can be used in the following generalization of Path TSP.

Metric Shortest Connected T-Join Problem (MSCTJ): Let $G = (V, E)$ be a complete graph with metric edge lengths $\ell\colon E \to \mathbb{R}_{\geqslant 0}$, and let $T \subseteq V$ be nonempty with $|T|$ even. Find a T-join $J \subseteq E$ minimizing $\ell(J) \coloneqq \sum_{e \in J} \ell(e)$ among all T-joins J such that (V, J) is connected.

Contrary to Path TSP, here we obtain a constrained spanning tree problem with constraints not only on a chain of cuts, but on a laminar family of width at most $|T| - 1$. These can be handled similarly as in MLCST, giving the following result.

Theorem 4.4. *For every $\varepsilon > 0$, there is a $(1.5+\varepsilon)$-approximation algorithm for the metric shortest connected T-join problem with running time $|V|^{\mathcal{O}(|T|/\varepsilon)}$.*

We remark that both the algorithms by Traub and Vygen [TV18] and Zenklusen [Zen19] can be generalized to MSCTJ in a similar way. Hence, for constant $|T| > 4$, all three approaches imply an efficient method improving on the previously best 1.6-approximation by Sebő [Seb13], with the generalization of the approach of Zenklusen giving the currently best known guarantee of 1.5. Nevertheless, we expand on our approach here in order to highlight another immediate implication of our new techniques.

4.1.2 Organization of the chapter

We start by introducing our techniques in the context of MCCST. Section 4.2 clearly outlines what we want to achieve with our dynamic program, and why this implies Theorem 4.1 together with negatively correlated rounding procedures and the solution alteration technique mentioned ealier. Section 4.3 then provides a thorough discussion of the key aspects of our dynamic programming technique. Section 4.4 contains additional details on the local alteration approach that we use to obtain a unicriteria approximation for MCCST, and shows a further application of this technique to turn bicriteria approximations into unicriteria ones. In Section 4.5, we discuss in detail why the natural generalization of our techniques to laminar constraint families fails, and how these difficulties can be overcome to obtain results for MLCST and a proof of Theorem 4.2. Section 4.6 presents some implications of our new technique in Path TSP and its generalization, the MSCTJ problem, leading to Theorems 4.3 and 4.4. Section 4.A discusses why, for MCCST, the natural LP relaxation is not strong enough to obtain results with guarantees as in Theorem 4.1, thus further motivating the use of a dynamic programming approach to strengthen the relaxation. Finally, Section 4.B presents an example showing that a classical analysis of our DP, namely by backtracing an optimal solution, is impossible in the laminarly constrained setting.

4.2 Overview of our approach for MCCST

The first step of our approach for MCCST relies on finding a solution to a suitable polyhedral relaxation. The canonical relaxation, which was also used in prior results on chain-constrained trees [LS16; OZ18], enhances the spanning tree polytope P_{ST} with the cut constraints. We recall that P_{ST} is the convex hull of all characteristic vectors of spanning trees in $G = (V, E)$, and, by a seminal result of Edmonds [Edm71], can be described by

$$P_{\mathrm{ST}} \coloneqq \left\{ x \in \mathbb{R}^E_{\geqslant 0} \;\middle|\; \begin{array}{ll} x(E) = |V| - 1 & \\ x(E[S]) \leqslant |S| - 1 & \forall S \subsetneq V,\ |S| \geqslant 2 \end{array} \right\} ,$$

where $E[S] \subseteq E$ are all edges with both endpoints in S. The polytope $Q \subseteq \mathbb{R}^E$ below describes the natural relaxation of MCCST:

$$Q \coloneqq \{x \in P_{\mathrm{ST}} \,|\, a_i \leqslant x(\delta(S_i)) \leqslant b_i \ \forall i \in [k]\} \quad .$$

Unfortunately, solutions to the relaxation $\min\{c^\top x \mid x \in Q\}$ are too weak for our purposes. In particular, there are instances where there exists a solution $y \in Q$ fulfilling the chain-constraints, even though any spanning tree must violate at least one chain constraint by a factor of at least 2. (We provide such an example in Section 4.A.) Hence, when comparing any integral solution to y, it will be impossible to stay within a factor of $1 + \varepsilon$ regarding the violation of constraints—but this is precisely what we want to achieve. This also shows a hard limit for prior approaches, which are all based on Q.

We therefore aim for a stronger relaxation. It turns out that the reason why Q can be a bad relaxation is the potential existence of small bounds a_i, b_i. Indeed, assume that all a_i, b_i for $i \in [k]$ were at least $c \cdot \log k$ for a sufficiently large constant c (depending on ε). Then one could first find an optimal solution x^* to $\min\{c^\top x \mid x \in Q\}$, and then round x^* to a spanning tree by using one of several negatively correlated

rounding procedure developed within the last 10 years (see [AGMOS10; CVZ10]), which lead to Chernoff-type concentration bounds. The theorem below summarizes a simplified form of the properties obtained by those procedures.[3]

Theorem 4.5 (see [AGMOS10; CVZ10])**.** *Let $y \in P_{\mathrm{ST}}$. There exists an efficient randomized rounding scheme for rounding y to a random spanning tree T in G such that:*

(i) $\Pr[e \in T] = y_e$ *for all* $e \in E$*, and*
(ii) For any $\lambda > 0$ *and* $U \subseteq E$*, we have*

$$\Pr\left[(1-\lambda)y(U) \leqslant |T \cap U| \leqslant (1+\lambda)y(U)\right] \geqslant 1 - 2e^{-y(U)\lambda^2/3} \ .$$

Consider applying Theorem 4.5 to x^* to obtain a spanning tree T. By choosing $U = \delta(S_i)$ in the above theorem for any $i \in [k]$, one obtains that $|T \cap \delta(S_i)|$ is within a $(1 \pm \varepsilon)$-factor of $x^*(\delta(S_i))$ with probability $1 - k^{\Omega(1)}$ if $x^*(\delta(S_i)) \geqslant c \cdot \log k$. Moreover, $x^* \in Q$ implies $a_i \leqslant x^*(\delta(S_i)) \leqslant b_i$ for $i \in [k]$. Hence, a union bound over all chain constraints shows that T is unlikely to violate any chain constraint by a large factor.

Motivated by this observation, we design a dynamic programming approach to find points $y \in Q$ of small cost that, for each $i \in [k]$, are either integral on the edges $\delta(S_i)$, or have a large value $y(\delta(S_i))$. To formalize this idea, we introduce the notion of τ-integral solutions.

Definition 4.6 (τ-integral)**.** *For $\tau \in \mathbb{Z}_{\geqslant 0}$, we say that a point $y \in \mathbb{R}^E$ is* τ-integral *(with respect to the cuts $S_1, \ldots, S_k$), if for each $i \in [k]$, either*

(i) $y(\delta(S_i)) \leqslant \tau$ *and y is integral on the edges in $\delta(S_i)$, or*
(ii) $y(\delta(S_i)) \geqslant \tau + 1$.

We call the cuts S_i satisfying (i) and (ii), respectively, the y-small *and* y-large *cuts.*

Clearly, any integral point is τ-integral for any $\tau \in \mathbb{Z}_{\geqslant 0}$. The key implication of our dynamic programming approach in the context of MCCST is the following.

Theorem 4.7. *For any $\tau \in \mathbb{Z}_{\geqslant 0}$, there is an algorithm that returns in $|V|^{\mathcal{O}(\tau)}$ time a τ-integral point $y \in Q$ with $c^\top y \leqslant c(\mathrm{OPT})$, where OPT is an optimal solution to MCCST.*

Not surprisingly, to obtain Theorem 4.7 we want our dynamic program to guess edges in the cuts that will later be y-small. However, this simple high-level plan comes with some important technical hurdles. In particular, even if we knew the edges used in some cut $\delta(S_i)$, completing the two parts of the spanning tree on the left-hand side of the cut (on the vertices S_i) and on its right-hand side (on $V \setminus S_i$), respectively, are two highly dependent subprobems. Interestingly, it is not easy to separate them into independent ones by guessing further structure, like the connectedness on each side, without creating NP-hard subproblems. We expand on these, and further issues, in Section 4.3, and show how one can address them. A key difference between classical dynamic programs and our approach is that our propagation step requires a fractional solution of a previous subproblem, and not just a small fingerprint of previously obtained solutions.

The issue of small cuts is now resolved by Theorem 4.7 by setting $\tau = \Theta(k)$ and rounding a τ-integral point $y \in Q$ using a randomized rounding procedure with the guarantees stated in Theorem 4.5: Because y is integral on y-small cuts, the rounding procedure will return a tree T such that χ^T coincides with y on all y-small cuts, because it is marginal-preserving (property (i) in Theorem 4.5).

One last technical hurdle to overcome to obtain a $(1, 1+\varepsilon)$-approximation for MCCST is that the properties of a negatively correlated rounding procedure, as stated in Theorem 4.5, are not enough to get a spanning tree that both (i) violates chain constraints at most slightly, and (ii) has cost no more than $c^\top y$. Indeed, typical applications of such rounding procedures only lead to $(1+\varepsilon)$-approximations in terms of the objective

[3] More generally, randomized rounding procedures with these properties can be obtained for any matroid base polytope and Chernoff-type concentration holds for any linear function with small non-negative coefficients (see [CVZ10]).

(see [CVZ10; CVZ09] for examples). We show that this loss in the objective is avoidable in MCCST, and other settings, by using a simple alteration step that modifies the obtained spanning tree by swapping one edge.

Theorem 4.8. *Let $y \in P_{\mathrm{ST}}$ and $c \in \mathbb{R}^E$. Let T be a random spanning tree in $G = (V, E)$ drawn from a distribution satisfying $\Pr[e \in T] = y(e)$ for all $e \in E$. Let $\overline{T}$ be a spanning tree minimizing $c(U)$ among all spanning trees U whose symmetric difference $U \triangle T := (U \setminus T) \cup (T \setminus U)$ with T satisfies $|U \triangle T| \leqslant 2$ and such that $y(e) \in (0, 1)$ for $e \in U \triangle T$. Then*

$$\Pr\left[c(\overline{T}) \leqslant c^\top y\right] \geqslant (|V| - 1)^{-1} \ .$$

In Section 4.4, we show that Theorem 4.8 holds even in a much more general context and has implications outside MCCST. For the specific setting of MCCST, we observe in Section 4.4 that also a method introduced by Linhares and Swamy [LS16] can be adapted to avoid the $(1 + \varepsilon)$-factor loss in the objective.

We can now put together the above ingredients to obtain our quasi-polynomial $(1, 1 + \varepsilon)$-approximation for MCCST, stated as Algorithm 4.1 below.

Algorithm 4.1: Quasi-polynomial $(1, 1 + \varepsilon)$-approximation for MCCST

1. Let $\tau := \lfloor {}^{96 \log(2|V|)}/_{\varepsilon^2} \rfloor$, and use Theorem 4.7 to find a τ-integral point $y \in Q$ with $c^\top y \leqslant c(\mathrm{OPT})$.
2. Let $\ell := \lceil 2|V| \log |V| \rceil$, and randomly round y with a rounding procedure as guaranteed by Theorem 4.5, ℓ times independently, to obtain spanning trees $T_1, \ldots, T_\ell$.
3. For each $j \in [\ell]$, find a minimum cost spanning tree $\overline{T}_j$ among all spanning trees T with $|T \triangle T_j| \leqslant 2$ and such that $y(e) \in (0, 1)$ for all $e \in T \triangle T_j$.
4. Among all $\overline{T}_j$ for $j \in [\ell]$ with $\frac{a_i}{1+\varepsilon} \leqslant |\overline{T}_j \cap \delta(S_i)| \leqslant (1 + \varepsilon) b_i$ for all $i \in [k]$, return one of smallest cost.

We now show that the above results—in particular Theorem 4.7, which follows from our dynamic program, and Theorem 4.8—imply that Algorithm 4.1 is a quasi-polynomial $(1, 1 + \varepsilon)$-approximation for MCCST.

Proof of Theorem 4.1. We will show that with probability at least $1 - {}^1/_{|V|}$, there is one spanning tree $\overline{T}_j$ among the trees $\overline{T}_1, \ldots, \overline{T}_\ell$ computed by Algorithm 4.1 that satisfies both

(i) $\frac{1}{1+\varepsilon} \cdot y(\delta(S_i)) \leqslant |\overline{T}_j \cap \delta(S_i)| \leqslant (1 + \varepsilon) \cdot y(\delta(S_i))$ for all $i \in [k]$, and
(ii) $c(\overline{T}_j) \leqslant c^\top y$,

which indeed implies that the returned solution is $(1, 1 + \varepsilon)$-approximate because $a_i \leqslant y(\delta(S_i)) \leqslant b_i$ for all $i \in [k]$ due to $y \in Q$, and y satisfies $c^\top y \leqslant c(\mathrm{OPT})$ as guaranteed by step 1 of the algorithm.

We first analyze a single random spanning tree among the spanning trees $T_1, \ldots, T_\ell$ determined in step 2 of the algorithm. We denote by T such a spanning tree that was obtained by randomly rounding y with a randomized rounding procedure as guaranteed by Theorem 4.5. Observe that because the rounding is marginal-preserving, χ^T coincides with y on any edge $e \in E$ with $y(e) \in \{0, 1\}$. As y is τ-integral, all edges within y-small cuts are of this type and T thus fulfills all chain constraints corresponding to y-small cuts. Together with Chernoff-type concentration bounds guaranteed by Theorem 4.5, applied with $\lambda = {}^\varepsilon/_4$ and using $y(\delta(S_i)) \geqslant \tau + 1$ for y-large cuts, we have

$$\Pr\left[\left(1 - \frac{\varepsilon}{4}\right) y(\delta(S_i)) \leqslant |T \cap \delta(S_i)| \leqslant \left(1 + \frac{\varepsilon}{4}\right) y(\delta(S_i))\right] \geqslant 1 - \frac{1}{2|V|^2} \quad \forall i \in [k] \ .$$

Now let $\overline{T}$ be a spanning tree of minimum cost among all spanning trees $U \subseteq E$ with $|U \triangle T| \leqslant 2$ and $y(e) \in (0, 1)$ for $e \in U \triangle T$. The cost of this spanning tree has the same distribution as the cost of the spanning trees $\overline{T}_1, \ldots, \overline{T}_\ell$ computed in step 3 of Algorithm 4.1. By Theorem 4.8, we have

$$\Pr\left[c(\overline{T}) \leqslant c^\top y\right] \geqslant \frac{1}{|V| - 1} \ .$$

Using a union bound over the k events described in (4.2) and the one described in (4.2), we obtain that T and $\overline{T}$ simultaneously fulfill

(a) $\left(1-\frac{\varepsilon}{4}\right) y(\delta(S_i)) \leqslant |T \cap \delta(S_i)| \leqslant \left(1+\frac{\varepsilon}{4}\right) y(\delta(S_i))$ for all $i \in [k]$, and
(b) $c(\overline{T}) \leqslant c^\top y$,

with probability at least

$$1 - \left(k \cdot \frac{1}{2|V|^2} + \left(1 - \frac{1}{|V|-1}\right)\right) \geqslant \frac{1}{2|V|} \ ,$$

where we used $k \leqslant |V|$ in the above inequality. Next, we show that property (a) above implies

$$\left(1-\frac{\varepsilon}{2}\right) y(\delta(S_i)) \leqslant |\overline{T} \cap \delta(S_i)| \leqslant \left(1+\frac{\varepsilon}{2}\right) y(\delta(S_i)) \quad \forall i \in [k] \ ,$$

which in turn implies $\frac{1}{1+\varepsilon} \cdot y(\delta(S_i)) \leqslant |\overline{T} \cap \delta(S_i)| \leqslant (1+\varepsilon) \cdot y(\delta(S_i))$ for all $i \in [k]$, providing the property that we seek as highlighted in (4.2). To see that (4.2) holds for any $i \in [k]$ that corresponds to a y-small cut, notice that for such i we have $\overline{T} \cap \delta(S_i) = T \cap \delta(S_i)$, as $\overline{T}$ and T only differ on edges on which y has a fractional value, and, due to τ-integraliy of y, small cuts do not contain such edges. Hence, consider $i \in [k]$ with $y(\delta(S_i)) \geqslant \tau + 1$. Because $|T \triangle \overline{T}| \leq 2$, $\overline{T}$ is either the same as T or obtained from T by replacing one edge by a different one, so

$$|T \cap \delta(S_i)| - 1 \leqslant |\overline{T} \cap \delta(S_i)| \leqslant |T \cap \delta(S_i)| + 1 \ .$$

The relation (4.2) for y-large cuts now follows from the above inequality and (a):

$$|\overline{T} \cap \delta(S_i)| \geqslant \left(1-\frac{\varepsilon}{4}\right) \cdot y(\delta(S_i)) - 1 \geqslant \left(1-\frac{\varepsilon}{2}\right) \cdot y(\delta(S_i)) \ , \text{ and}$$
$$|\overline{T} \cap \delta(S_i)| \leqslant \left(1+\frac{\varepsilon}{4}\right) \cdot y(\delta(S_i)) + 1 \leqslant \left(1+\frac{\varepsilon}{2}\right) \cdot y(\delta(S_i)) \ ,$$

where the second inequality in each of the two above lines follows from $y(\delta(S_i)) \geqslant \tau + 1 \geqslant {}^{96 \log(2|V|)}/_{\varepsilon^2}$, because S_i is y-large.

In summary, the tree $\overline{T}$ satisfies the two desired properties highlighted in (4.2) with probability at least $(2|V|)^{-1}$. Because the algorithm computes $\ell = \lceil 2|V| \log |V| \rceil$ independent random trees $\overline{T}_1, \ldots, \overline{T}_j$ with the same distribution as $\overline{T}$, the probability that at least one of them fulfills the properties highlighted in (4.2) is at least

$$1 - \left(1 - \frac{1}{2|V|}\right)^{\ell} \geqslant 1 - e^{-\frac{\ell}{2|V|}} \geqslant 1 - \frac{1}{|V|} \ ,$$

as desired. Finally, the running time is dominated by the quasi-polynomial time dynamic programming approach used to find a cheap τ-integral point $y \in Q$ in step 1 of Algorithm 4.1. (All other steps of the algorithm can be performed efficiently.) By Theorem 4.7, we thus get a running time bound $|V|^{\mathcal{O}(\tau)} = |V|^{\mathcal{O}(\log |V|/\varepsilon^2)}$. □

4.3 The dynamic programming approach for MCCST

First observe that to prove Theorem 4.7, it suffices to consider $\tau \leqslant |V| - 1$, because any τ-integral point in $y \in Q$ for $\tau \geqslant |V| - 1$ is integral as $Q \subseteq P_{\mathrm{ST}}$, and the y-value on any cut is at most $|V| - 1$ for any point in P_{ST}. Thus, any $\tau \geqslant |V|$ can be replaced by $\tau = |V| - 1$, so we assume $\tau \leqslant |V| - 1$ in what follows.

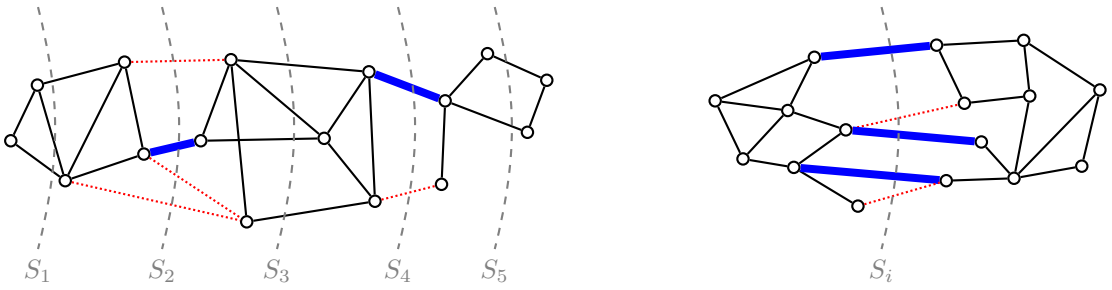
Our dynamic program to find a cheap τ-integral point in Q is inspired by recent dynamic programming approaches in the context of Path TSP [TV18; Zen19], but faces important new technical challenges that require novel conceptual insights. To highlight this point, let us first consider the significantly simpler special case of $\tau = 1$. The dynamic programming approaches for Path TSP are essentially algorithms for this case.[4]

[4] More precisely, dynamic programs for Path TSP are looking for points in the Held-Karp relaxation of Path TSP instead of the spanning tree polytope, but this is only a minor technical difference without significant impact on the dynamic program.

4.3.1 Brief overview to find cheap 1-integral solution following prior techniques

To gain intuition for this special case, which nicely allows for showcasing later on the added difficulty faced for general τ, assume that we knew upfront the small cuts with respect to an optimal solution $\mathrm{OPT} \subseteq E$, i.e., the cuts among $S_1, \dots, S_k$ in which OPT contains a single edge. Let $S_{i_1}, \dots, S_{i_\ell}$ for $1 \leqslant i_1 < \dots < i_\ell \leqslant k$ be these small cuts. For notational convenience, we set $S_{i_0} \coloneqq \emptyset$ and $S_{i_{\ell+1}} \coloneqq V$. Now consider the $\ell + 1$ induced subgraphs $G[S_{i_1}]$, $G[S_{i_2} \setminus S_{i_1}]$,..., $G[S_{i_\ell} \setminus S_{i_{\ell-1}}]$, $G[V \setminus S_{i_\ell}]$.[5] It is not hard to observe that the edges of OPT within each of these subgraphs must form a spanning tree in that subgraph. Moreover, for $j \in [\ell]$, the single edge $e_j \in \mathrm{OPT} \cap \delta(S_{i_j})$ must go from $S_{i_j} \setminus S_{i_{j-1}}$ to $S_{i_{j+1}} \setminus S_{i_j}$ for OPT to be a spanning tree (see Fig. 4.1a). If, moreover, we even knew the single edge $e_j \in \delta(S_{i_j}) \cap \mathrm{OPT}$ for each $j \in [\ell]$, then the problem of finding a *corresponding* cheapest 1-integral point $y \in P_{\mathrm{ST}}$—i.e., with y-small cuts $S_{i_1}, \dots, S_{i_\ell}$ and edges $e_1, \dots, e_\ell$ contained in them—decomposes into $\ell + 1$ independent linear programs, one within each of the above-mentioned induced subgraphs. More precisely, one has to find, for $j \in [\ell + 1]$, a cheapest point y^j in the spanning tree polytope of $G[S_{i_j} \setminus S_{i_{j-1}}]$ with lower bounds on each cut S_i with $S_{i_j} \subsetneq S_i \subsetneq S_{i_{j-1}}$ to make sure that y^j, together with the guessed edges in small cuts, has a load of at least 2 on these cuts.

The above observations now naturally lead to a dynamic programming approach that extends solutions from left to right, i.e., a 1-integral solution in some subgraph $G[S_i]$ for some i is extended to one on $G[S_j]$ for $j > i$. This way, one can use a dynamic program to optimize over all possibilities of small cuts and edges contained in them (see [TV18; Zen19] for more details of this approach in the context of Path TSP).



(a) Independent problems.

(b) Dependent problems.

Figure 4.1: **(a)** Here, the small cuts are $S_i = S_2$ and $S_i = S_4$, and we assume that only a single edge, drawn in thick and blue, crosses each small cut. Finding a τ-integral point $y \in P_{\mathrm{ST}}$ boils down to solving independent subproblems in $G[S_2]$, $G[S_4 \setminus S_2]$, and $G[V \setminus S_4]$. **(b)** Because more than one edge is in the small cut S_i, the problem does not decompose into independent subproblems on the left-hand side and right-hand side of S_i.

4.3.2 Toward general τ with connectivity patterns and resulting challenges

However, if $\tau \geqslant 2$, i.e., if there are two or more edges in small cuts, splitting the problem into independent ones along small cuts comes with significant additional challenges linked to obtaining connectivity and acyclicity globally from independent solutions of the subproblems (see Fig. 4.1b). One natural approach to try to address such challenges is to maintain more structure in the dynamic program, by for example also enumerating over potential *connectivity patterns* of edges in small cuts, i.e., ways of how the edges in a small cut could be connected on one or either side of the cut. For ease of presentation, consider a situation with a single small cut S_i, and a given selection of $t \leqslant \tau$ many edges $F \coloneqq \{\{u_j, v_j\} \mid j \in [t]\} \subseteq \delta(S_i)$ with $u_j \in S_i$ and $v_j \notin S_i$ for all $j \in [t]$. A *connectivity pattern* for the right-hand side of the cut is a partition $\mathcal{C}$ of $\{v_1, \dots, v_t\}$, where a set $C \in \mathcal{C}$ indicates that the vertices in C shall be connected in $G[V \setminus S_i]$. We call the triple $(S_i, F, \mathcal{C})$ a *connectivity triple* (see Fig. 4.2).

[5] For $W \subseteq V$, $G[W]$ is the subgraph of G induced by W.

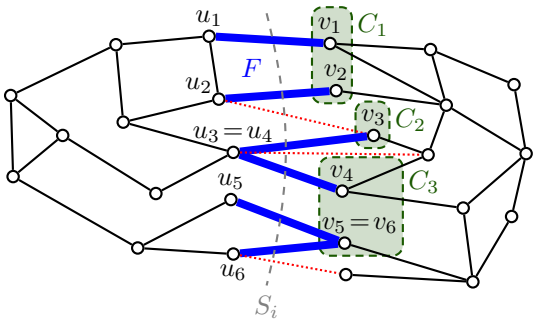


Figure 4.2: A connectivity triple $(S_i, F, \mathcal{C})$ with connectivity pattern $\mathcal{C} = \{C_1, C_2, C_3\}$.

Definition 4.9 (Compatibility with a connectivity triple)**.**

(i) A spanning tree $T \subseteq E$ is compatible *with the connectivity triple $(S_i, F, \mathcal{C})$ if $T \cap \delta(S_i) = F$ and the partition on $\{v_1, \dots, v_t\}$ induced by the connected components of $T \cap E[V \setminus S_i]$ equals $\mathcal{C}$.*

(ii) A set $R \subseteq \binom{V \setminus S_i}{2}$ is right-compatible *with $(S_i, F, \mathcal{C})$ if R is a forest and the partition on $\{v_1, \dots, v_t\}$ induced by the connected components of R equals $\mathcal{C}$.*

(iii) Let $U \subseteq E[S_i]$, and let $R \subseteq \binom{V \setminus S_i}{2}$ be right-compatible with $(S_i, F, \mathcal{C})$. Then U is left-compatible *with $(S_i, F, \mathcal{C})$ if $U \cup F \cup R$ is a spanning tree.*

(iv) Let $x \in \mathbb{R}^E$ with $\operatorname{supp}(x) \subseteq E[S_i]$, and let $R \subseteq \binom{V \setminus S_i}{2}$ be right-compatible with $(S_i, F, \mathcal{C})$. Then x is left-compatible *with $(S_i, F, \mathcal{C})$ if $x + \chi^F + \chi^R$ is in the spanning tree polytope of $(V, E \cup R)$.*

We highlight that right-compatible sets R are not required to be a subset of the edges of G, but can contain any pairs of vertices within $V \setminus S_i$. This makes sure that right-compatible sets exist for any connectivity triple, which simplifies the exposition. Moreover, one can observe that the above definitions of left-compatibility do not depend on which right-compatible set R is chosen, and are thus well-defined.

Knowing the correct connectivity triple, the desired separation into independent subproblems can actually be achieved. However, this comes at the cost that the subproblem on the side where we guessed the connectivity pattern becomes substantially harder than in the simple case $\tau = 1$. This is nicely highlighted by a simple connectivity pattern $\mathcal{C}$: Assume that all right endpoints $\{v_1, \dots, v_t\}$ of F are distinct, t is even, and let $\mathcal{C} = \{\{v_1, v_2\}, \{v_3, v_4\}, \dots, \{v_{t-1}, v_t\}\}$ be a grouping of the endpoints into pairs. To pinpoint the difficulties, consider the question of whether there exists a right-compatible edge set $U \subseteq E[V \setminus S_i]$. For this to be the case, U must be a forest with $t/2$ components, one for each pair v_j, v_{j+1} that connects that pair. Such a set U exists if and only if there are t vertex-disjoint paths in $G[V \setminus S_i]$, one between v_j and v_{j+1} for each $j \in \{1, 3, \dots, t-1\}$. Hence, just determining whether there exists a right-compatible solution is at least as difficult as the vertex-disjoint paths problem. Though this problem is efficiently solvable for a constant number of paths $t/2$, due to the seminal results by Robertson and Seymour in the context of the Graph Minor Project (see also [KKR12] for a faster procedure), the known techniques for disjoint paths are highly non-trivial, non-polyhedral, cannot handle costs, and, last but not least, the connectivity pattern $\mathcal{C}$ leading to the disjoint paths problem remains a very special case of connectivity patterns we have to deal with.

4.3.3 Efficiently extending subsolutions through relaxed connectivity requirements

To overcome this issue in our approach, we still enumerate over connectivity triples with connectivity patterns on the right-hand side of small cuts as described above, but will later not require that right-hand side solutions are right-compatible with it; they only have to properly complete an existing left-hand side solution. The left-hand side solutions, however, will be left-compatible with the guessed connectivity triples.

We start by observing a simple way to describe left-compatibility. Let $(S_i, F, \mathcal{C})$ be a connectivity triple. Then, left-compatible edge sets $U \subseteq E[S_i]$ are simply spanning trees in an auxiliary graph $G(S_i, F, \mathcal{C})$

which we obtain from $G[S_i]$ through the following operations: (i) add the edges F and their endpoints to $G[S_i]$, (ii) contract the vertex sets in $\mathcal{C}$, and (iii) contract the edges in F. There is a canonical one-to-one relation between the edges in $G(S_i, F, \mathcal{C})$ and $E[S_i]$, and we therefore treat them as the same edge set. By $P_{\mathrm{ST}}(S_i, F, \mathcal{C})$, we denote the spanning tree polytope of $G(S_i, F, \mathcal{C})$. Hence, $P_{\mathrm{ST}}(S_i, F, \mathcal{C})$ are all points that are left-compatible with $(S_i, F, \mathcal{C})$.

We are now ready to describe our dynamic programming approach. For ease of notation, we set $S_0 \coloneqq \emptyset$ and $S_{k+1} \coloneqq V$. Consider the set $\mathcal{K}$ of all connectivity triples $(S_i, F, \mathcal{C})$, where $i \in \{0, \dots, k+1\}$ and the crossing edges $F \subseteq \delta(S_i)$ satisfy $a_i \leqslant |F| \leqslant \min\{\tau, b_i\}$. For each $(S_i, F, \mathcal{C}) \in \mathcal{K}$ we determine via our dynamic program a point (partial solution) $y_{(S_i,F,\mathcal{C})} \in \mathbb{R}^{E[S_i]}$ with the following property.

Property 4.10.

(i) $y_{(S_i,F,\mathcal{C})} \in P_{\mathrm{ST}}(S_i, F, \mathcal{C})$.
(ii) $y_{(S_i,F,\mathcal{C})} + \chi^F$ *is* τ*-integral on* $S_1, \dots, S_{i-1}$.
(iii) $a_h \leqslant y_{(S_i,F,\mathcal{C})}(\delta(S_h)) + |F \cap \delta(S_h)| \leqslant b_h$ *for all* $h \in [i-1]$.
(iv) $c^\top y_{(S_i,F,\mathcal{C})}$ *is at most the cost of a cheapest edge set* $U \subseteq E[S_i]$ *such that* χ^U *fulfills (i), (ii), and (iii).*

For simplicity of notation, we consider all vectors defined on some subset $U \subseteq E$ of the edges, like $y_{(S_i,F,\mathcal{C})}$ above, to be vectors in $\mathbb{R}^E$, where all entries on which the vector was not defined are set to 0.[6] Clearly, if we can obtain such points in $|V|^{\mathcal{O}(\tau)}$ time, then we are done because $y \coloneqq y_{(V,\emptyset,\{\emptyset\})}$ is in Q because of (i) and (iii); y is a τ-integral point due to (ii); and y satisfies $c^\top y \leqslant c(\mathrm{OPT})$ due to (iv), as desired.

To construct the vectors $y_{(S_i,F,\mathcal{C})}$, we initialize $y_{(S_0,\emptyset,\{\emptyset\})}$ to the zero vector, and consider triples $(S_i, F, \mathcal{C}) \in \mathcal{K}$ in increasing order of i. Let $i \in [k+1]$, let $(S_i, \overline{F}, \overline{\mathcal{C}}) \in \mathcal{K}$, and assume that for each $(S_j, F, \mathcal{C}) \in \mathcal{K}$ with $j < i$, we already computed a vector $y_{(S_j,F,\mathcal{C})}$ satisfying Property 4.10. To compute a vector $y_{(S_i,\overline{F},\overline{\mathcal{C}})}$ satisfying Property 4.10, we consider all $(S_j, F, \mathcal{C}) \in \mathcal{K}$ with $j < i$ and for each such triple, we solve the following linear program, which finds a cheapest extension z of $y_{(S_j,F,\mathcal{C})}$ that is left-compatible with $(S_i, \overline{F}, \overline{\mathcal{C}})$:

$$
\begin{aligned}
\min \quad & c^\top z & & \\
& z \in P_{\mathrm{ST}}(S_i, \overline{F}, \overline{\mathcal{C}}) & & \\
\max\{\tau+1, a_h\} \leqslant\ & z(\delta(S_h)) + |\overline{F} \cap \delta(S_h)| \leqslant b_h & & \forall h \in \{j+1, \dots, i-1\} \\
& z(e) = y_{(S_j,F,\mathcal{C})}(e) & & \forall e \in E[S_j] \\
& z(e) = 1 & & \forall e \in F \\
& z(e) = 0 & & \forall e \in \delta(S_j) \setminus F \ .
\end{aligned}
\tag{exLP}
$$

Among all linear programs of type (exLP), i.e., one for each triple $(S_j, F, \mathcal{C})$ with $j < i$, we determine the one achieving the smallest optimal value and set $y_{(S_j,\overline{F},\overline{\mathcal{C}})}$ to be an optimal solution of that linear program.[7] This finishes the description of our dynamic program. The bottleneck of the running time is the repeated solving of linear programs of type (exLP). A simple bound on the number of such LPs that we solve is $|\mathcal{K}|^2$, and the running time of $|V|^{\mathcal{O}(\tau)}$ then follows from the following bound and the fact that we can solve (exLP) in strongly polynomial time through standard techniques. A more formal treatment, including a proof of the simple statement below, is given in Section 4.3.4.

Proposition 4.11. $|\mathcal{K}| = |V|^{\mathcal{O}(\tau)}$.

We now highlight a few key aspects of our dynamic program. First, even though (exLP) seeks to complete a prior solution $y_{(S_j,F,\mathcal{C})}$ to one for the triple $(S_i, \overline{F}, \overline{C})$, we do not require the completion to be right-compatible

[6] This makes sure that expressions like $c^\top y_{(S,F,\mathcal{C})}$ or $y_{(S,F,\mathcal{C})}(\delta(S_h))$ are well-defined.

[7] We use the usual convention that if some linear program (exLP) is infeasible, then its objective value is interpreted as ∞ (and we will never use a solution to an infeasible linear program later on). Infeasibility occurs, for example, for choices of $(S_j, F, \mathcal{C})$ where F does not contain all edges of $\overline{F} \cap \delta(S_j)$.

with $(S_j, F, \mathcal{C})$.[8] This connectivity pattern is completely disregarded in (exLP) and was only used on the left-hand side of S_j to construct $y_{(S_j,F,\mathcal{C})}$. The key observation is that any integral solution that is compatible with both triples $(S_j, F, \mathcal{C})$ and $(S_i, \overline{F}, \overline{\mathcal{C}})$, and has only large cuts between S_i and S_j, provides a legal way to complete $y_{(S_j,F,\mathcal{C})}$ as formally described by the following statement.

Lemma 4.12. *Let $R \subseteq \binom{V \setminus S_i}{2}$ be right-compatible with $(S_i, \overline{F}, \overline{\mathcal{C}})$. Then for any $U \subseteq E[S_i]$ such that $T := U \cup \overline{F} \cup R$ is a spanning tree compatible with both $(S_j, F, \mathcal{C})$ and $(S_i, \overline{F}, \overline{\mathcal{C}})$, and $|T \cap \delta(S_h)| \in [\max\{\tau + 1, a_h\}, b_h]$ for all $h \in \{j+1, \dots, i-1\}$, the following vector is a feasible solution to (exLP):*

$$z := y_{(S_j,F,\mathcal{C})} + \chi^{F \setminus \overline{F}} + \chi^{U \cap E[S_i \setminus S_j]} \ .$$

The above lemma is crucial to make sure that our dynamic program remains a relaxation of the original MCCST problem, even though it requires to complement a specific previously computed solution $y_{(S_j,F,\mathcal{C})}$. Simply speaking, Lemma 4.12 guarantees that the increment in cost when extending the solution $y_{(S_j,F,\mathcal{C})}$ up to the cut S_i through our dynamic program is no more than the best integral extension that realizes both connectivity triples at S_j and S_i. This makes sure that the solutions we compute fulfill point (iv) of Property 4.10. Intuitively, the reason why Lemma 4.12 holds is the following: No matter what precise point $y_{(S_j,F,\mathcal{C})}$ we computed, as long as it is left-compatible with $(S_j, F, \mathcal{C})$, it can be completed by any edge set that is right-compatible with $(S_j, F, \mathcal{C})$ to obtain a point in P_{ST}.

Also note that a crucial difference between our dynamic program and the classical way of using dynamic programming approaches is that we need an *explicit* solution $y_{(S_j,F,\mathcal{C})}$ in our propagation/extension step. Only knowing the connectivity triple $(S_j, F, \mathcal{C}) \in \mathcal{K}$ and the value of a best point $y_{(S_j,F,\mathcal{C})}$ for that triple would not be enough. To highlight this contrast, consider for example a classical dynamic programming approach for the (integer) knapsack problem (see, e.g., [KV18, Section 17.2]). Here, the dynamic program computes for every possible cost the smallest total weight of items realizing that cost. (Sometimes, the dynamic program is presented with the roles of costs and weights exchanged, in which case, for every possible weight, the dynamic program computes a minimum cost solution of that weight.) In the propagation step of this classical dynamic program, to extend existing partial solutions of smaller cost to partial solutions of larger cost, one only needs to know the weight and cost of previously computed partial solutions, but not the exact partial solution. Partial solutions are sometimes saved in classical dynamic programs to quickly retrieve a solution through backtracking, once the whole dynamic programming table is filled. However, to just determine the optimal value of a solution, and in particular to perform the propagation steps in the dynamic program, partial solutions are not used.

4.3.4 Details of the dynamic programming approach for MCCST

In this section, we complete the proofs that are missing to formally ensure that the dynamic programming approach described above achieves the guarantees claimed by Theorem 4.7. Recall that the dynamic program initializes $y_{(S_0,\emptyset,\emptyset)} = 0$, and propagates to points $y_{(S_i,F,\mathcal{C})} \in \mathbb{R}^E_{\geqslant 0}$ for all $(S_i, F, \mathcal{C}) \in \mathcal{K}$ using Algorithm 4.2.

Algorithm 4.2: Propagation to $y_{(S_i,\overline{F},\overline{\mathcal{C}})}$ from all $y_{(S_j,F,\mathcal{C})}$ with $j < i$.

1. For all $(S_j, F, \mathcal{C}) \in \mathcal{K}$ with $j < i$, solve the linear program (exLP) and obtain an optimal solution y.
2. Among all solutions y found in step 1, let $y_{(S_i,\overline{F},\overline{\mathcal{C}})}$ be one minimizing $c^\top y$. Return $y_{(S_i,\overline{F},\overline{\mathcal{C}})}$.

We start by proving Lemma 4.12, which essentially shows that the extension found by our dynamic program when solving a linear program of the form (exLP) has cost no more than the best integral extension that is compatible with the connectivity triples on both sides of the extension.

[8] Notice that we did not formally define right-compatibility for fractional points because we do not need it, but a natural extension would be to say that it is a convex combination of right-compatible integral solutions.

Proof of Lemma 4.12. To obtain feasibility of z for (exLP), it is easy to see that the inequality constraints that are stated explicitly in (exLP) follow immediately by definition of z. Thus, it remains to prove that $z \in P_{\mathrm{ST}}(S_i, \overline{F}, \overline{\mathcal{C}})$, i.e., that z is left-compatible with $(S_i, \overline{F}, \overline{\mathcal{C}})$.

By assumption, the tree T is compatible with $(S_j, F, \mathcal{C})$, hence $T \cap \binom{V \setminus S_j}{2}$ is right-compatible with $(S_j, F, \mathcal{C})$. Moreover, $y_{(S_j,F,\mathcal{C})}$ is left-compatible with $(S_j, F, \mathcal{C})$. Combining these observations, we see that $y_{(S_j,F,\mathcal{C})} + \chi^F + \chi^{T \cap \binom{V \setminus S_j}{2}}$ is in the spanning tree polytope of $(V, E \cup R)$. By partitioning $T \cap \binom{V \setminus S_j}{2}$ into $T \cap E[S_i \setminus S_j]$, $\overline{F} \setminus F$ and R, we get that, equivalently, $y_{(S_j,F,\mathcal{C})} + \chi^F + \chi^{T \cap E[V \setminus S_j]} + \chi^{\overline{F} \setminus F} + \chi^R$ is in the spanning tree polytope of $(V, E \cup R)$, which implies that $y_{(S_j,F,\mathcal{C})} + \chi^{F \setminus \overline{F}} + \chi^{T \cap E[S_i \setminus S_j]}$ is left-compatible with $(S_i, \overline{F}, \overline{\mathcal{C}})$. As $T \cap E[S_i \setminus S_j] = U \cap E[S_i \setminus S_j]$, the Lemma follows. □

With the bound on the incremental cost of an extension from Lemma 4.12, we can show that propagating along Algorithm 4.2, we maintain Property 4.10.

Lemma 4.13. *Let $i \in [k+1]$ and $(S_i, \overline{F}, \overline{\mathcal{C}}) \in \mathcal{K}$. Assume that for all $(S_j, F, \mathcal{C}) \in \mathcal{K}$ with $j < i$, we are given points $y_{(S_j,F,\mathcal{C})}$ satisfying Property 4.10. Then $y_{(S_i,\overline{F},\overline{\mathcal{C}})}$ obtained by Algorithm 4.2 also has Property 4.10.*

Proof. Note that any solution of (exLP) satisfies points (i), (ii) and (iii) of Property 4.10 with respect to $(S_i, \overline{F}, \overline{\mathcal{C}})$. Point (i) follows directly from the corresponding constraint in (exLP), while (ii) and (iii) are implied by the constraints in the linear program for cuts S_h with $h \geqslant j$, and follow for cuts S_h with $h < j$ from the fact that $y_{(S_j,F,\mathcal{C})}$ has Property 4.10, by assumption.

To see that point (iv) holds, it is enough to see that for any edge set $U \subseteq E[S_i]$ fulfilling (i), (ii) and (iii), there is a connectivity triple $(S_i, F, \mathcal{C})$ such that the corresponding solution y of the linear program (exLP) satisfies $c^\top y \leqslant c(U)$. To see this, fix such an edge set U, which is by definition left-compatible with $(S_i, \overline{F}, \overline{\mathcal{C}})$. Thus, for any $R \subseteq \binom{V \setminus S_i}{2}$ that is right-compatible with $(S_i, \overline{F}, \overline{\mathcal{C}})$, the edge set $T \coloneqq U \cup \overline{F} \cup R$ is a spanning tree in $(V, E \cup R)$ compatible with $(S_i, \overline{F}, \overline{\mathcal{C}})$. Let $j < i$ be maximal such that S_j is a T-small cut. Let $F = T \cap \delta(S_j)$ and let $\mathcal{C}$ be the connectivity pattern such that T is compatible with $(S_j, F, \mathcal{C})$.

We claim that $(S_j, F, \mathcal{C})$ is the connectivity triple that we are looking for, i.e., if y is an optimal solution of (exLP) when extending $y_{(S_j,F,\mathcal{C})}$ to $(S_i, \overline{F}, \overline{\mathcal{C}})$, then $c^\top y \leqslant c(U)$. Thereto, observe that by definition, $T = U \cup \overline{F} \cup R$ is a spanning tree in $(V, E \cup R)$ compatible with both (S_j, F, C) and $(S_i, \overline{F}, \overline{\mathcal{C}})$, and by the choice of j, we also have $|T \cap \delta(S_h)| \in [\max\{\tau + 1, a_h\}, b_h]$ for all $h \in \{j+1, \ldots, i-1\}$. Thus, R and U satisfy the assumptions of Lemma 4.12, and we get that $z \coloneqq y_{(S_j,F,\mathcal{C})} + \chi^{F \setminus \overline{F}} + \chi^{U \cap E[S_i \setminus S_j]}$ is a feasible solution to (exLP). But y is an optimal solution of the same linear program, hence

$$c^\top y \leqslant c^\top y_{(S_j,F,\mathcal{C})} + c(\chi^{F \setminus \overline{F}}) + c(U \cap E[S_i \setminus S_j]) \ .$$

By assumption, $y_{(S_j,F,\mathcal{C})}$ satisfies Property 4.10, and applying point (iv) with the edge set $U \cap E[S_j]$, we get $c^\top y_{(S_j,F,\mathcal{C})} \leqslant c(U \cap E[S_j])$. Combining this with the above, we obtain the desired inequality

$$c^\top y \leqslant c(U \cap E[S_j]) + c(\chi^{F \setminus \overline{F}}) + c(U \cap E[S_i \setminus S_j]) = c(U) \ . \qquad \square$$

The final step before wrapping up and proving Theorem 4.7 is to show the bound on $|\mathcal{K}|$ from Proposition 4.11.

Proof of Proposition 4.11. The set $\mathcal{K}$ consists of all connectivity triples $(S_i, F, \mathcal{C})$, where $i \in \{0, 1, \ldots, k+1\}$, $F \subseteq \delta(S_i)$ has size at most τ, and $\mathcal{C}$ is a corresponding connectivity pattern. In order to build such a triple, there are at most $|V| + 1$ options for choosing a set S_i (one for each $i \in \{0, \ldots, k+1\}$, and we have $k \leqslant |V| - 1$). Furthermore, note that $\delta(S_i)$ can contain at most $\mathcal{O}(|V|^2)$ many edges of G, hence there are at most $|V|^{\mathcal{O}(\tau)}$ many choices for a subset of size at most τ. Finally, $\mathcal{C}$ is a partition of the at most τ many endpoints of the edges in F that do not lie inside S_i, and the number of such partitions can be bounded by $|V|^{\mathcal{O}(\tau)}$ as well, where we use $\tau \leqslant |V| - 1$. In total, we thus get $|\mathcal{K}| \leqslant |V|^{\mathcal{O}(\tau)}$. □

Finally, we are ready to prove Theorem 4.7.

Proof of Theorem 4.7. We calculate points $y_{(S_i,F,\mathcal{C})}$ for all $(S_i, F, \mathcal{C}) \in \mathcal{K}$ by initializing $y_{(\emptyset,\emptyset,\{\emptyset\})} = 0$ and calculating $y_{(S_i,F,\mathcal{C})}$ in increasing order of i using the propagation step described in Algorithm 4.2. Note that $y_{(\emptyset,\emptyset,\{\emptyset\})}$ has Property 4.10, and hence, by an inductive application of Lemma 4.13, we obtain that all points $y_{(S_i,F,\mathcal{C})}$ for $(S_i, F, \mathcal{C}) \in \mathcal{K}$ have Property 4.10. In particular, Property 4.10 for $y_{(V,\emptyset,\{\emptyset\})}$ immediately implies that this point has the properties claimed by Theorem 4.7. Note that the guarantee on $c^\top y_{(V,\emptyset,\{\emptyset\})}$ follows from the fact that a cheapest edge set U such that χ^U fulfils points (i), (ii) and (iii) of Property 4.10 with respect to $(S_i, F, \mathcal{C}) = (V, \emptyset, \{\emptyset\})$ is in fact an optimal solution.

For the running time bound, observe that the dominating operation of our dynamic programming procedure is repeatedly solving linear programs of type (exLP). The total number of such linear programs that we solve is bounded from above by $|\mathcal{K}|^2$, and therefore, the running time of $|V|^{\mathcal{O}(\tau)}$ follows from Proposition 4.11 and the observation that linear programs of the type (exLP) can be solved in strongly polynomial time. The latter can be achieved by using a compact extended formulation for the spanning tree polytope with small coefficients in the constraint matrix (one can, for example, use the one by Martin [Mar91], which has coefficients that are bounded by 1 in absolute value), and then applying the framework of Tardos [Tar86]. □

4.4 Local correction steps for rounding procedures in $\{0, 1\}$-polytopes

In this section, we discuss details on the proof of Theorem 4.8, used to avoid a $(1 + \varepsilon)$-factor loss in the objective value. We present this result separately because it may be of independent interest, as it applies to a broad class of problem settings. At the end of this section, in Section 4.4.3, we briefly discuss how, for MCCST, an alternative approach introduced by Linhares and Swamy [LS16] also allows for avoiding a loss in the objective value.

We show Theorem 4.8 by proving a more general statement for $\{0, 1\}$-polytopes, based on polyhedral neighborhoods. We therefore start with some basic polyhedral terminology.[9] $\{0, 1\}$-polytopes are a representation of *set systems* $(E, \mathcal{F})$, where E is a finite ground set and $\mathcal{F} \subseteq 2^E$. One can think of $\mathcal{F}$ as the feasible sets of some combinatorial problem over E. For example, E may be the edge set of a graph and $\mathcal{F}$ the family of all spanning trees. The *combinatorial polytope* $P_\mathcal{F}$ that corresponds to $\mathcal{F}$ is defined by

$$P_\mathcal{F} := \operatorname{conv}(\{\chi^F \mid F \in \mathcal{F}\}) \ ,$$

where conv denotes the convex hull. Hence, if $\mathcal{F}$ are all spanning trees of a graph, $P_\mathcal{F}$ is the spanning tree polytope. For $F_1, F_2 \in \mathcal{F}$ and $q \in \mathbb{Z}_{\geqslant 0}$, we say that F_2 is in the q-neighborhood of F_1 on $P_\mathcal{F}$ if one can reach the vertex χ^{F_2} of $P_\mathcal{F}$ from the vertex χ^{F_1} by successively traversing at most q edges of $P_\mathcal{F}$. Notice that this natural notion extends the way we modify T to obtain $\overline{T}$ in Theorem 4.8: Indeed, this follows from the well-known property that two spanning trees $T, \overline{T}$ in G—or, more generally, any two bases $T, \overline{T}$ of a matroid—have the property that $\chi^T, \chi^{\overline{T}}$ are adjacent in $P_\mathcal{F}$ if and only if $|T \triangle \overline{T}| = 2$ (see, e.g., [Sch03, Volume B]). Furthermore, for any $y \in P_\mathcal{F}$, we denote by $P_{\mathcal{F}_y} \subseteq P_\mathcal{F}$ the minimal face of $P_\mathcal{F}$ that contains y. Additionally, $\mathcal{F}_y \subseteq \mathcal{F}$ denotes all sets $F \in \mathcal{F}$ such that $\chi^F \in P_{\mathcal{F}_y}$. (Note that these definitions of $\mathcal{F}_y$ and $P_{\mathcal{F}_y}$ are consistent with (4.4) in the sense that $P_{\mathcal{F}_y}$ is indeed the combinatorial polytope of the family $\mathcal{F}_y$.)

A key quantity in our derivations is the cardinality $\rho(\mathcal{F})$ of a largest size set in $\mathcal{F}$:

$$\rho(\mathcal{F}) := \max\{|F| \mid F \in \mathcal{F}\} \ .$$

Typically, when having a set system $(E, \mathcal{F})$ where all sets have the same cardinality, i.e., $|F| = \rho(\mathcal{F})$ for all $F \in \mathcal{F}$, we can obtain slightly stronger results later. We call such set systems *equal-cardinality* systems. Note that the family of spanning trees (or bases of any matroid) is an equal-cardinality system. We prove the following generalization of Theorem 4.8.

[9] We refer the interested reader to [Sch03, Volume A] for more information on polyhedral combinatorics.

Theorem 4.14. *Let $(E,\mathcal{F})$ be a set system, let $y \in P_{\mathcal{F}}$, $c \in \mathbb{R}^E$, $q \in \mathbb{Z}_{\geqslant 1}$, and let T be a random set in $\mathcal{F}$ drawn from a distribution satisfying $\Pr[e \in T] = y_e$ for all $e \in E$. Let $\overline{T} \in \mathcal{F}$ be a set minimizing $c(U)$ among all $U \in \mathcal{F}$ in the q-neighborhood of T on $P_{\mathcal{F}_y}$. Then*

$$\Pr\left[c(\overline{T}) \leqslant c^\top y\right] \geqslant \frac{q}{2\rho(\mathcal{F})} \ .$$

Moreover, if $(E,\mathcal{F})$ is an equal-cardinality system, then

$$\Pr\left[c(\overline{T}) \leqslant c^\top y\right] \geqslant \frac{q}{\rho(\mathcal{F})} \ .$$

First observe that Theorem 4.14 indeed implies Theorem 4.8.

Proof of Theorem 4.8. We set E to be the edges of $G=(V,E)$, $\mathcal{F} \subseteq 2^E$ to be all spanning trees in G, and $q=1$. Clearly, in this case we have $\rho(\mathcal{F}) = |V|-1$ and $|F| = \rho(\mathcal{F})$ for all $F \in \mathcal{F}$, because every spanning tree has precisely $|V|-1$ edges. Hence spanning trees form an equal-cardinality system. By Theorem 4.14 we thus obtain

$$\Pr[c(W) \leqslant c^\top y] \geqslant \frac{1}{|V|-1} \ ,$$

where $W \in \mathcal{F}$ is a set minimizing $c(U)$ among all $U \subseteq \mathcal{F}$ in the 1-neighborhood of T on $P_{\mathcal{F}_y}$.

For the above to imply Theorem 4.8, it suffices to show that any $U \subseteq \mathcal{F}$ in the 1-neighborhood of T on $P_{\mathcal{F}_y}$ fulfills $y(e) \in (0,1)$ for all $e \in U \Delta T$. Because $P_{\mathcal{F}} \subseteq [0,1]^E$—i.e., non-negativity constraints and constraints of type $x(e) \leqslant 1$ are valid for $P_{\mathcal{F}}$—all points on $P_{\mathcal{F}_y}$ coincide with y on the edges where y is integral, i.e.,

$$P_{\mathcal{F}_y} \subseteq \left\{x \in [0,1]^E \;\middle|\; x(e) = y(e)\ \forall e \in E \text{ with } y(e) \in \{0,1\}\right\} \ .$$

This implies that any $F \subseteq \mathcal{F}_y$ fulfills $y(e) \in (0,1)$ for all $e \in F \mathbin{\triangle} T$. Hence, this also holds for any $U \subseteq \mathcal{F}$ in the 1-neighborhood of T on $P_{\mathcal{F}_y}$, as desired, and finishes the proof. □

4.4.1 Proof of Theorem 4.14

We show Theorem 4.14 in several steps. We first derive a bound on the cost of a well-chosen set A in the 1-neighborhood of another fixed set F. The following lemma formalizes this statement. The cost improvement is measured with respect to the cost of some target set Q, which will later be chosen to be a set in $\mathcal{F}$ of smallest cost.

Lemma 4.15. *Let $(E,\mathcal{F})$ be a set system. Let $c \in \mathbb{R}^E$, and $F, Q \in \mathcal{F}$ with $F \neq Q$. Then there exists a neighbor $A \in \mathcal{F}$ of F on $P_{\mathcal{F}}$ satisfying*

(i) $c(A) - c(Q) \leqslant \left(1 - \frac{1}{|Q \mathbin{\triangle} F|}\right) \cdot (c(F) - c(Q))$, *and*
(ii) $|Q \mathbin{\triangle} A| \leqslant |Q \mathbin{\triangle} F| - 1$.

Moreover, if $(E,\mathcal{F})$ is an equal-cardinality system, then the above properties can be strengthened to

(i') $c(A) - c(Q) \leqslant \left(1 - \frac{2}{|Q \mathbin{\triangle} F|}\right) \cdot (c(F) - c(Q))$, *and*
(ii') $|Q \mathbin{\triangle} A| \leqslant |Q \mathbin{\triangle} F| - 2$.

Proof. Consider the vertex χ^F of $P_{\mathcal{F}}$, and the family $\{A_1, \ldots, A_\ell\} \in \mathcal{F}$ of all neighboring sets in $\mathcal{F}$ on $P_{\mathcal{F}}$. We start with a basic polyhedral property, namely that the cone with apex χ^F spanned by all edges of $P_{\mathcal{F}}$ incident with χ^F contains the whole polytope, i.e.,

$$P_{\mathcal{F}} \subseteq \chi^F + \operatorname{cone}\left(\left\{\chi^{A_i} - \chi^F \;\middle|\; i \in [\ell]\right\}\right) \ .$$

In particular, this implies that there exist coefficients $\lambda_i \geqslant 0$ for $i \in [\ell]$ such that

$$\chi^Q = \chi^F + \sum_{i=1}^{\ell} \lambda_i \cdot \left(\chi^{A_i} - \chi^F\right) \quad .$$

We are only interested in strictly positive coefficients. Let k be the number of strictly positive coefficients, and assume, by renumbering the indices, that these are the coefficients $\lambda_1, \ldots, \lambda_k$. Hence,

$$\chi^Q = \chi^F + \sum_{i=1}^{k} \lambda_i \cdot \left(\chi^{A_i} - \chi^F\right) \quad ,$$

and $\lambda_i > 0$ for all $i \in [k]$. Let $\lambda \coloneqq \sum_{i=1}^{k} \lambda_i$.

Claim. We have $\lambda \leqslant |Q \mathbin{\triangle} F|$. Moreover, if $(E, \mathcal{F})$ is an equal-cardinality system, then $\lambda \leqslant \frac{1}{2}|Q \mathbin{\triangle} F|$.

Proof of claim. We have

$$\begin{aligned}
|Q \Delta F| &= \|\chi^Q - \chi^F\|_1 \\
&= \left\| \sum_{i=1}^{k} \lambda_i \cdot \left(\chi^{A_i} - \chi^F\right) \right\|_1 \\
&= \left\| \sum_{i=1}^{k} \lambda_i \cdot \left(\chi^{A_i \setminus F} - \chi^{F \setminus A_i}\right) \right\|_1 \\
&= \sum_{i=1}^{k} \lambda_i \cdot \left\| \chi^{A_i \setminus F} - \chi^{F \setminus A_i} \right\|_1 \\
&= \sum_{i=1}^{k} \lambda_i \cdot |A_i \mathbin{\triangle} F| \quad , \qquad (4.1)
\end{aligned}$$

where the second equality follows from (4.4.1), and the forth one from the fact that all vectors $\chi^{A_i \setminus F} - \chi^{F \setminus A_i}$ in the sum are non-positive on entries corresponding to F and non-negative on all other entries. Now, because A_i are neighbors of F on $P_{\mathcal{F}}$, we have $A_i \neq F$ for $i \in [k]$, and hence $|A_i \mathbin{\triangle} F| \geqslant 1$. This implies, together with (4.1), the first statement of the claim. Moreover, if $(E, \mathcal{F})$ is an equal-cardinality system, then $A_i \neq F$ implies $|A_i \mathbin{\triangle} F| \geqslant 2$, which leads to the strengthened statement of the claim for equal-cardinality systems. □

Taking the scalar product of c with both sides of (4.4.1), and rearranging terms, we get

$$c(F) - c(Q) = \sum_{i=1}^{k} \lambda_i \cdot (c(F) - c(A_i)) \quad .$$

Using an averaging argument, there exists an index $j \in [k]$ such that

$$\frac{1}{\lambda} \cdot (c(F) - c(Q)) \leqslant c(F) - c(A_j) \quad ,$$

which is equivalent to

$$c(A_j) - c(Q) \leqslant \left(1 - \frac{1}{\lambda}\right) (c(F) - c(Q)) \quad .$$

We will show that $A \coloneqq A_j$ fulfills the statement of the lemma. First observe that (4.4.1) together with the claim implies that A fulfills properties (i) and (i'), respectively. To show (ii) and (ii'), we show that any A_i for $i \in [k]$ fulfills $|Q \triangle A_i| < |Q \triangle F|$. This indeed implies both point (ii) and (ii'), because when dealing with equal-cardinality systems, the symmetric difference between any two sets in the system has even cardinality. Hence, it remains to show $|Q \triangle A_i| < |Q \triangle F|$.

We start by observing that equation (4.4.1) implies $A_i \subseteq Q \cup F$. Indeed, if there were any $e \in A_i$ with $e \notin Q \cup F$, then this would lead to a strictly positive entry for e on the right-hand side of (4.4.1), whereas χ^Q, which appears on the left-hand side of (4.4.1), has a 0-entry at e. Analogously, we can derive that $Q \cap F \subseteq A_i$, because if there was $e \in (Q \cap F) \setminus A_i$, then this would imply that the right-hand side of (4.4.1) has as its entry at e a value strictly less than 1, contradicting that the left-hand side has a value of 1 at entry e. In summary, we have $A_i \subseteq Q \cup F$ and $Q \cap F \subseteq A_i$. However, among all sets satisfying these properties, the set F is the unique set that maximizes the symmetric difference with Q. Because $A_i \neq F$ we thus have $|Q \triangle A_i| < |Q \triangle F|$, as desired, which finishes the proof of Lemma 4.15. □

Lemma 4.15 is a statement about finding good sets in the 1-neighborhood of any set F. By repeatedly applying the lemma, we obtain the following generalization for q-neighborhoods.

Lemma 4.16. *Let $(E, \mathcal{F})$ be a set system. Let $c \in \mathbb{R}^E$, and let $F, Q \in \mathcal{F}$ with $F \neq Q$. Then, for any $q \in \{1, \dots, |Q \triangle F|\}$, there exists a set $A \in \mathcal{F}$ in the q-neighborhood of F on $P_{\mathcal{F}}$ satisfying*

(i) $c(A) - c(Q) \leqslant \left(1 - \frac{q}{|Q \triangle F|}\right) \cdot (c(F) - c(Q))$, *and*
(ii) $|Q \triangle A| \leqslant |Q \triangle F| - q$.

Moreover, if $(E, \mathcal{F})$ is an equal-cardinality system, then we obtain the following strengthening. For any $q \in \{1, \dots, \frac{1}{2}|Q \triangle F|\}$, there exists a set $A \in \mathcal{F}$ in the q-neighborhood of F on $P_{\mathcal{F}}$ satisfying

(i') $c(A) - c(Q) \leqslant \left(1 - \frac{2q}{|Q \triangle F|}\right) \cdot (c(F) - c(Q))$, *and*
(ii') $|Q \triangle A| \leqslant |Q \triangle F| - 2q$.

Proof. We prove the lemma by induction on q. For $q = 1$ the statement holds due to Lemma 4.15. Now assume $q > 1$, and we show the inductive step for the case where $(E, \mathcal{F})$ is not necessarily an equal-cardinality system. The extension to equal-cardinality systems is analogous. By the inductive hypothesis, there is a set $\overline{A} \in \mathcal{F}$ in the $(q-1)$-neighborhood of F on $P_{\mathcal{F}}$ satisfying

(a) $c(\overline{A}) - c(Q) \leqslant \left(1 - \frac{q-1}{|Q \triangle F|}\right) \cdot (c(F) - c(Q))$, and
(b) $|Q \triangle \overline{A}| \leqslant |Q \triangle F| - (q-1)$.

Moreover, applying Lemma 4.15 to $F = \overline{A}$, we obtain that there is a set $A \in \mathcal{F}$ in the 1-neighborhood of $\overline{A}$ in $P_{\mathcal{F}}$—and hence, A is a q-neighbor of F in $P_{\mathcal{F}}$—such that

(c) $c(A) - c(Q) \leqslant \left(1 - \frac{1}{|Q \triangle \overline{A}|}\right) \cdot (c(\overline{A}) - c(Q))$, and
(d) $|Q \triangle A| \leqslant |Q \triangle \overline{A}| - 1$.

The fact that A fulfills point (ii) is now an immediate consequence of (b) and (d). Moreover, we have

$$\begin{aligned} c(A) - c(Q) &\leqslant \left(1 - \frac{1}{|Q \triangle \overline{A}|}\right) \cdot \left(1 - \frac{q-1}{|Q \triangle F|}\right) \cdot (c(F) - c(Q)) \\ &\leqslant \left(1 - \frac{1}{|Q \triangle F| - (q-1)}\right) \cdot \left(1 - \frac{q-1}{|Q \triangle F|}\right) \cdot (c(F) - c(Q)) \\ &= \left(1 - \frac{q}{|Q \triangle F|}\right) \cdot (c(F) - c(Q)) \quad , \end{aligned}$$

where the first inequality follows from (c) and (a), and the second one from (b). Hence, this shows that A also fulfills (i) and finishes the proof. □

Our next lemma, Lemma 4.17, shows that if the value of a set $T \in \mathcal{F}$ is not significantly larger than $c^\top y$, then there is a good solution in its neighborhood. Afterwards, in Lemma 4.18, we provide a lower bound for the probability of this happening if T has a distribution with marginals given by y, which is the setting of Theorem 4.14.

Lemma 4.17. *Let $(E, \mathcal{F})$ be a set system, let $y \in P_\mathcal{F}$, $\mu \geqslant 1$, and $\eta = \min\{c(F) \mid F \in \mathcal{F}_y\}$. Then for any $T \in \mathcal{F}_y$ with*

$$(\mu - 1) \cdot (c(T) - \eta) \leqslant \mu \cdot (c^\top y - \eta) \ ,$$

there is a set $U \in \mathcal{F}_y$ in the $\lceil 2\rho(\mathcal{F}_y)/\mu \rceil$-neighborhood of T on $P_{\mathcal{F}_y}$ with $c(U) \leqslant c^\top y$. Moreover, if $(E, \mathcal{F}_y)$ is an equal-cardinality system, then such a set $U \in \mathcal{F}$ even exists in the $\lceil \rho(\mathcal{F}_y)/\mu \rceil$-neighborhood of T on $P_{\mathcal{F}_y}$.

Proof. The statement trivially holds for $\mu = 1$. Hence, assume $\mu > 1$. Let $q := \lceil \rho(\mathcal{F}_y)/\mu \rceil$ if $(E, \mathcal{F})$ is an equal-cardinality system, and $q := \lceil 2\rho(\mathcal{F}_y)/\mu \rceil$ otherwise. Furthermore, we define

$$Q \in \operatorname{argmin} \{c(F) \mid F \in \mathcal{F}_y\} \ ,$$

and hence, $c(Q) = \eta$. By Lemma 4.16, there is a set $U \in \mathcal{F}_y$ in the q-neighborhood of T on $P_{\mathcal{F}_y}$ satisfying

$$\begin{aligned} c(U) - \eta &\leqslant \left(1 - \frac{2\rho(\mathcal{F}_y)}{|Q \mathbin{\triangle} T| \cdot \mu}\right) \cdot (c(T) - \eta) \\ &\leqslant \left(1 - \frac{1}{\mu}\right) \cdot (c(T) - \eta) \\ &\leqslant \left(1 - \frac{1}{\mu}\right) \cdot \frac{\mu}{\mu - 1} (c^\top y - \eta) \\ &= c^\top y - \eta \ , \end{aligned}$$

where the second inequality follows from $2\rho(\mathcal{F}_y) \geqslant |Q \mathbin{\triangle} T|$, and the third one from the inequality given in the statement of Lemma 4.17. Hence, U fulfills the properties required by Lemma 4.17, which finishes the proof. □

Lemma 4.18. *Let $(E, \mathcal{F})$ be a set system, let $y \in P_\mathcal{F}$, $\mu \geqslant 1$, and $\eta = \min\{c(F) \mid F \in \mathcal{F}_y\}$. Moreover, let T be a random set in $\mathcal{F}_y$ drawn from a distribution that satisfies $\Pr[e \in T] = y_e$ for all $e \in E$. Then*

$$\Pr\left[(\mu - 1) \cdot (c(T) - \eta) \leqslant \mu \cdot (c^\top y - \eta)\right] \geqslant \frac{1}{\mu} \ .$$

Proof. First observe that $c(T) - \eta$ is a non-negative random variable with expected value $c^\top y - \eta$, as $\Pr[e \in T] = y_e$ for all $e \in E$. If $c^\top y = \eta$, then the statement holds because $c(T) - \eta$ has expectation zero and is non-negative; thus, it is 0 with probability 1. Assume from now on $c^\top y > \eta$. Then, the lemma is a consequence of Markov's inequality, which implies

$$\Pr\left[c(T) - \eta \geqslant \frac{\mu}{\mu - 1} \cdot (c^\top y - \eta)\right] \leqslant \frac{\mu - 1}{\mu}.$$

Writing $\frac{\mu-1}{\mu} = 1 - \frac{1}{\mu}$, we see that this implies the statement of the lemma. □

Finally, combining Lemma 4.17 and Lemma 4.18, Theorem 4.14 now readily follows.

Proof of Theorem 4.14. By choosing $\mu = 2\rho(\mathcal{F})/q$ in both Lemma 4.17 and Lemma 4.18 we immediately obtain the first part of Theorem 4.14. The bound for the case of equal cardinality set systems is obtained by setting $\mu = \rho(\mathcal{F})/q$ in both Lemma 4.17 and Lemma 4.18. □

4.4.2 Further applications of alteration technique

The presented alteration technique is a rather general approach that is not tightly linked to the MCCST setting where we applied it. It may thus be of independent interest. In particular, it can be used to avoid a multiplicative loss in the objective in several contexts where (randomized) rounding approaches are used. We briefly mention one such application. The following packing result was shown in [CVZ09].

Theorem 4.19 (Theorem 6.2 in [CVZ09])**.** *Let P be the base polytope of a matroid on ground set N, let $A \in [0,1]^{m \times N}$, and let $b \in \mathbb{R}^N$. Then there is a $(1+\varepsilon, \mathcal{O}(\log m / \log\log m))$-bicriteria approximation for the problem*

$$\min \left\{ c^\top x \;\middle|\; x \in \{0,1\}^N,\ x \in P,\ Ax \leqslant b \right\} \quad ,$$

where the first guarantee is w.r.t. the cost of the solution and the second one w.r.t. the overflow on the packing constraints.

The above approximation was obtained by rounding a point $y \in P$ that fulfills $c^\top y \leqslant c(\mathrm{OPT})$ through a negatively correlated rounding procedure. Such a procedure preserves marginals, and hence, falls into the setting of our Theorem 4.14, which allows for avoiding the loss in the objective, at an additive $+1$ cost in the second objective, which is negligible. Through this alteration, one obtains a unicriteria $(1, \mathcal{O}(\log m / \log\log m))$-approximation.

4.4.3 Alternative approach to avoid $1+\varepsilon$ loss via techniques of Linhares and Swamy

We briefly want to highlight that a recently introduced approach of Linhares and Swamy [LS16] also allows for avoiding a $(1+\varepsilon)$-factor loss in the objective. More precisely, they introduced a Lagrangian relaxation based approach to reduce certain bicriteria weighted packing problems to bicriteria unweighted packing problems. Within this framework, they also show how it can be modified to avoid losses in the objective value under some conditions. For this they need an LP-based rounding procedure with certain properties.

In the following, we focus on the specific problem of MCCST to expand further on this approach and how it can be made to work in this context. For simplicity, consider an MCCST problem with only upper bounds on the chain constraints, which falls within the setting of packing constraints considered in [LS16]. Let $y \in Q$ be a fractional point as computed in the first step of Algorithm 4.1. The point y can be interpreted as an optimal LP solution to a linear program on the minimal face of the spanning tree polytope on which y lies, together with upper bounds on the chain constraints of large y-value. This allows for interpreting y as an LP solution as required by the framework of Linhares and Swamy. Additionally, the framework needs a rounding procedure that both (i) rounds y to a spanning tree T on the same minimal face of the spanning tree polytope on which y lies, and (ii) the spanning tree T needs to satisfy that $|T \cap \delta(S_i)|$ is within a $(1 \pm \epsilon)$-factor of b_i for each chain constraint corresponding to a set S_i for which y is tight, i.e., $y(\delta(S_i)) = b_i$. Notice that it is important that the rounding does not just return a tree almost fulfilling the chain constraints, but we also need that y-tight chain constraints remain nearly-tight after rounding. Our alteration step does not require such a property, but has other requirements. Hence, the two techniques are not strictly comparable.

The negatively correlated rounding procedure that we employ fulfills both requirements stated above. It always rounds to a spanning tree on the same minimal face, because it is marginal-preserving. Moreover, equation (4.2) shows that the load on chain constraints does not change much.

Finally, we want to mention that the framework in [LS16] can also be adjusted to deal with lower bounds in our context of MCCST.

4.5 Extension to MLCST

The key ingredient of the approximation algorithm for MCCST presented in the previous sections is obtaining a τ-integral point y that is feasible for the linear relaxation of the problem. Ideally, we would like to find

such a point in the more general case of MLCST as well, and then apply Theorems 4.5 and 4.8 for rounding and local corrections, as before.

Unfortunately, analyzing the natural generalization of our dynamic programming approach, where we determine partial solutions $y_{(S,F,\mathcal{C})}$ for all connectivity triples $(S, F, \mathcal{C})$ with $S \in \mathcal{L}$ by continuously extending previously obtained solutions, comes with obstacles even if the laminar family $\mathcal{L}$ has constant width. To highlight some aspects thereof, consider the instance given in Fig. 4.3, where the laminar family $\mathcal{L}$ consists of precisely two sets S_1 and S_2, and edge costs c are such that edges in $E[S_1] \cup E[S_2]$ have cost 1, and all other edges have cost 0.

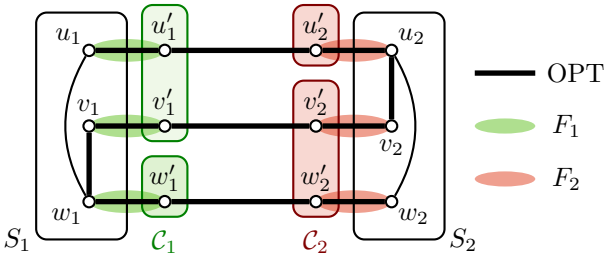


Figure 4.3: Connectivity triples $(S_1, F_1, \mathcal{C}_1)$ and $(S_2, F_2, \mathcal{C}_2)$ induced by an optimal solution OPT.

It is easy to see that every optimal solution has cost 2, as for example the one indicated in bold. Note that in our dynamic program, we would first construct solutions $y_{(S_i,F,\mathcal{C})}$ compatible with the pattern $(S_i, F, \mathcal{S})$ for all such connectivity patterns, and then try to extend every combination of solutions on S_1 and S_2 to a global solution. Typically, such dynamic programming approaches are analyzed by backtracing an optimal solution. In our example, consider the optimal solution given in Fig. 4.3. This solution induces the connectivity triples $(S_1, F_1, \mathcal{C}_1)$ and $(S_2, F_2, \mathcal{C}_2)$ on S_1 and S_2, respectively, where $F_i = \{\{u_i, u_i'\}, \{v_i, v_i'\}, \{w_i, w_i'\}\}$ for $i \in \{1, 2\}$, $\mathcal{C}_1 = \{\{u_1', v_1'\}, \{w_1'\}\}$ and $\mathcal{C}_2 = \{\{u_2'\}, \{v_2', w_2'\}\}$, as indicated. Ideally, we would like that the cheapest common extension of $y_{(S_1,F_1,\mathcal{C}_1)}$ and $y_{(S_2,F_2,\mathcal{C}_2)}$ has cost at most the cost of an optimal solution. However, note that we could potentially have

$$y_{(S_1,F_1,\mathcal{C}_1)} = \chi^{\{\{u_1,w_1\}\}} \quad \text{and} \quad y_{(S_2,F_2,\mathcal{C}_2)} = \chi^{\{\{u_2,w_2\}\}},$$

in which case we can easily see that there does not even exist a feasible solution that restricts to $y_{(S_1,F_1,\mathcal{C}_1)}$ and $y_{(S_2,F_2,\mathcal{C}_2)}$ on S_1 and S_2, respectively.

Note that in the case of MCCST, where we only had to consider extensions from a single partial solution on a smaller set, the analysis outlined above was enough to obtain our result (see the proof of Lemma 4.12). In particular, it was enough to know that the dynamic program considered building a solution along the small cuts and connectivity triples induced by an optimal solution (even though these are not known upfront). For MLCST, we deviate from this typical analysis and exploit that the DP considers all potential connectivity triples on the small cuts induced by an optimal solution. Let us illustrate this in the above example. We claim that there exist connectivity patterns $\mathcal{C}_1'$ and $\mathcal{C}_2'$ (potentially different from the patterns $\mathcal{C}_1$ and $\mathcal{C}_2$ induced by the optimal solution) such that the best common extension of $y_{(S_1,F_1,\mathcal{C}_1')}$ and $y_{(S_2,F_2,\mathcal{C}_2')}$ has cost at most 2. To see this, we proceed iteratively, starting with $\mathcal{C}_1' = \mathcal{C}_1$. Note that if on $E[S_1]$, we replace the edges of the optimal solution by those of $y_{(S_1,F_1,\mathcal{C}_1)}$, the new point is a feasible solution and, by definition of $y_{(S_1,F_1,\mathcal{C}_1)}$, the total cost does not increase. Observe that the new set of edges induces a different connectivity pattern on S_2 than OPT did, and let this pattern be $\mathcal{C}_2'$ (see Fig. 4.4). Now replacing the optimal solution on $E[S_2]$ by $y_{(S_2,F_2,\mathcal{C}_2')}$, which is $\chi^{\{\{u_2,v_2\}\}}$ in our example, we again see that feasibility is guaranteed, and the total cost does again not increase. To finish the argument, note that we just constructed a common extension of the two partial solutions $y_{(S_1,F_1,\mathcal{C}_1')}$ and $y_{(S_2,F_2,\mathcal{C}_2')}$ of cost no more than the cost of the optimal solution—thus, the best extension will be of cost at most OPT, proving the desired guarantee.

The above idea of iteratively defining suitable connectivity patterns can be generalized to an arbitrary number of sibling sets $S_1, \ldots, S_w$, and is crucial in the analysis of the propagation step of our dynamic

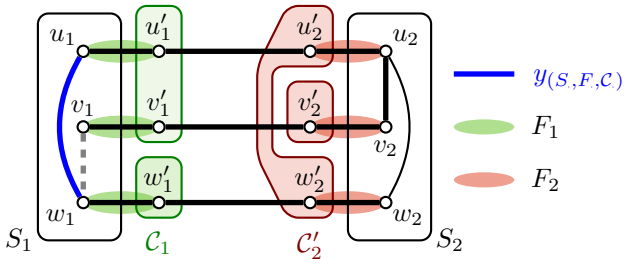


Figure 4.4: Patching $y_{(S,F,\mathcal{C})}$ induces $\mathcal{C}'_2$.

program. For now, we want to highlight that replacing parts of an optimal solution by a previously obtained partial solution and then inducing new connectivity patterns on other parts requires the partial solutions to be integral, as there is no notion of induced connectivity patterns for fractional solutions. In our extension steps, however, we find common extensions of partial solutions through a linear program similar to (exLP), which will in general not be integral. For this reason, we apply the rounding and local correction methods presented in Theorems 4.5 and 4.8 after every single extension step, giving integral solutions at every stage and thus allowing for inducing connectivity patterns iteratively as defined above.

The example from Fig. 4.3 discussed earlier might seem contrived due to the fact that the highlighted problems could be avoided by breaking ties in the right way, for example by choosing different optimal partial solutions $y_{(S_1,F_1,\mathcal{C}_1)}$ and $y_{(S_2,F_2,\mathcal{C}_2)}$. Even though this is the case here, there exist more complex examples without any tie-breaking options that exhibit the same issue. One such example is presented in Section 4.B.

It is important to also recall that Theorems 4.5 and 4.8 provide guarantees on constraint violation and cost of the integral solution obtained through rounding only with certain probabilities. For a more concise analysis, our algorithm will at each step apply the rounding and local correction operations repeatedly until—in expected polynomial running time—an integral solution with the desired properties is found. More formally, we obtain a Las Vegas algorithm with the following guarantees.

Theorem 4.20. *For every $\varepsilon > 0$, there is a $(1, 1+\varepsilon)$-approximation algorithm for MLCST with expected running time $|V|^{\mathcal{O}(k \log |V|/\varepsilon^2)}$, where k is the width of the laminar family $\mathcal{L}$.*

Note that any Las Vegas algorithm can easily be transformed into a randomized approximation algorithm, i.e, with deterministic polynomial running time and where the returned solution has the desired properties with high probability: By Markov's inequality, the probability that the running time of a single run of the algorithm guaranteed by Theorem 4.20 is less that twice the expected running time is at least $1/2$. Consequently, among $\log_2 |V|$ many independent runs, the probability that at least one run succeeds is at least $1 - 1/|V|$. Thus, Theorem 4.2 stated in the introduction is implied by Theorem 4.20.

In the following two sections, we present the modifications of our dynamic programming approach for the laminar case in detail, and we expand on the ideas highlighted above, leading to a proof of Theorem 4.20.

4.5.1 Dynamic programming in the laminar case

For a formal description of the dynamic programming algorithm for MLCST, we stick to the notation defined for MCCST, now of course considering the laminar family $\mathcal{L}$ instead of the chain $S_1 \subsetneq \ldots \subsetneq S_k$. We denote by $\mathcal{K}$ the set of all connectivity triples $(S, F, \mathcal{C})$ where $S \in \mathcal{L} \cup \{\emptyset, V\}$, with the crossing edges $F \subseteq \delta(S)$ satisfying $a_S \leqslant |F| \leqslant \min\{\tau, b_S\}$ for $S \in \mathcal{L}$, where the parameter τ is the maximal number of edges that the dynamic program "guesses" in small cuts. For MLCST, we will again choose $\tau = \mathcal{O}(\log |V|/\varepsilon^2)$.

For every $(S, F, \mathcal{C}) \in \mathcal{K}$, we use our dynamic program to determine a partial solution $T_{(S,F,\mathcal{C})} \subseteq E[S]$ with the following properties. For $S \in \mathcal{L}$, we denote $\mathcal{L}_S := \{S' \in \mathcal{L} \mid S' \subsetneq S\}$.

Property 4.21.

(i) $T_{(S,F,\mathcal{C})}$ *is a spanning tree of* $G(S,F,\mathcal{C})$.
(ii) $(1-\varepsilon)\cdot a_{S'} \leqslant |(T_{(S,F,\mathcal{C})} \cup F) \cap \delta(S')| \leqslant (1+\varepsilon)\cdot b_{S'}$ *for all* $S' \in \mathcal{L}_S$.
(iii) $c(T_{(S,F,\mathcal{C})})$ *is at most the cost of a cheapest edge set* $U \subseteq E[S]$ *that forms a spanning tree of* $G(S,F,\mathcal{C})$ *and satisfies* $a_{S'} \leqslant |(U \cup F) \cap \delta(S')| \leqslant b_{S'}$ *for all* $S' \in \mathcal{L}_S$.

It is clear that if in expected running time $|V|^{\mathcal{O}(k \log |V|/\varepsilon^2)}$, we can obtain trees $T_{(S,F,\mathcal{C})}$ with the above properties, then Theorem 4.20 follows because the tree $T := T_{(V,\emptyset,\emptyset)}$ has precisely the desired properties: (i) and (ii) state that T is a spanning tree of G violating the cut constraints only by a factor of $(1 \pm \varepsilon)$, and by (iii), $c(T)$ is at most the cost of an optimal solution.

To compute all $T_{(S,F,\mathcal{C})}$, we initialize $T_{(\emptyset,\emptyset,\{\emptyset\})} = \emptyset$, and then propagate to $T_{(S,F,\mathcal{C})}$ for all $(S,F,\mathcal{C}) \in \mathcal{K}$ in an order such that for all $S, S' \in \mathcal{L}$, if $S' \subsetneq S$, then $T_{(S',F,\mathcal{C})}$ is computed before $T_{(S,F,\mathcal{C})}$. A new tree $T_{(S,F,\mathcal{C})}$ is obtained in two steps: First, for all choices of connectivity triples $(S_1,F_1,\mathcal{C}_1),\dots,(S_w,F_w,\mathcal{C}_w)$ such that $S_1,\dots,S_w \in \mathcal{L}_S$ and $S_1,\dots,S_w$ have pairwise empty intersections, we extend the partial solutions $T_{(S_i,F_i,\mathcal{C}_i)}$ for $i \in [w]$ to a solution T on $(S,F,\mathcal{C})$. In a second step, we find the best solution among all extensions obtained this way, and keep it as $T_{(S,F,\mathcal{C})}$. The full propagation procedure is summarized in Algorithm 4.4, and details of a single extension step are described in Algorithm 4.3.

Algorithm 4.3: Extending partial solutions $T_{(S_1,F_1,\mathcal{C}_1)},\dots,T_{(S_w,F_w,\mathcal{C}_w)}$ to $(S,F,\mathcal{C})$.

---

1. Let $\tau := \lfloor 96\log(2|V|)/\varepsilon^2 \rfloor$, and let y be a minimizer of

$$\begin{aligned} \min \quad & c^\top y && && \text{(lamExLP)}\\ & y \in P_{\mathrm{ST}}(S_i,F,\mathcal{C}) && \\ \max\{\tau+1, a_{S'}\} \leqslant\ & y(\delta(S')) + |F \cap \delta(S')| \leqslant b_{S'} && \forall S' \in \mathcal{L}\colon \exists i \in [w] \text{ with } S_i \subsetneq S' \subsetneq S \\ & y(e) = \chi^{T_{(S_i,F_i,\mathcal{C}_i)}}(e) && \forall e \in E[S_i],\ \forall i \in [w] \\ & y(e) = \chi^{F_i}(e) && \forall e \in \delta(S_i) \setminus F,\ \forall i \in [w]\ . \end{aligned}$$

2. Randomly round y with a rounding procedure as guaranteed by Theorem 4.5 to obtain a spanning tree T_0 of $G(S,F,\mathcal{C})$.
3. Find a minimum cost spanning tree T among all spanning trees of $G(S,F,\mathcal{C})$ with $|T \triangle T_0| \leqslant 2$ and such that $y(e) \in (0,1)$ for all $e \in T \triangle T_0$.
4. If T satisfies
$$(1-\varepsilon)\cdot a_{S'} \leqslant |(T_{(S,F,\mathcal{C})} \cup F) \cap \delta(S')| \leqslant (1+\varepsilon)\cdot b_{S'}$$
for all $S' \in \mathcal{L}_S$, and $c(T) \leqslant c^\top y$, output T. Else, repeat from step 2.

Let us expand on the nature of the extension step given in Algorithm 4.3. The purpose of the linear program (lamExLP) is to find a common extension y of the trees $T_{(S_i,F_i,\mathcal{C}_i)}$ for $i \in [w]$ that uses precisely the edges F_i in the cuts $\delta(S_i)$, and is left-compatible with $(S,F,\mathcal{C})$. Note that the last condition appears in (lamExLP) as the constraint $y \in P_{\mathrm{ST}}(S,F,\mathcal{C})$. Additionally, we require that the partial solution y together with the edges in F satisfy the cut constraints on cuts $S' \in \mathcal{L}$ with $S_i \subsetneq S' \subsetneq S$, with a load of at least $\tau+1$ on all those cuts. Recall that the latter comes from the idea of finding τ-integral fractional solutions (which is what we need to control cut sizes in the rounding procedure) and letting the dynamic program try all combinations of maximal small cuts $S_1,\dots,S_w \subsetneq S$.

Once a fractional extension y is found, we use a rounding procedure as guaranteed by Theorem 4.5 to round it to an integral solution, namely a spanning tree T_0. Note that by definition, y coincides with $T_{(S_i,F_i,\mathcal{C}_i)}$ on $E[S_i]$ for all $i \in [w]$. As the rounding scheme is marginal-preserving (property (i) in Theorem 4.5), it follows that T_0 will coincide with these partial solutions as well, thus inheriting their properties. On $E[S \setminus \bigcup_{i=1}^w S_i]$, property (ii) will make sure that all cut constraints are satisfied up to small multiplicative

errors. In step 3, we exploit the exchange steps described in Theorem 4.8 to regain potential loss in the objective that may have occurred in step 2 compared to $c^\top y$. Both step 2 and step 3 can fail with certain probabilities, hence we repeatedly apply them until an extension with the properties listed in step 4 is found.

We remark that the linear program (lamExLP) might be infeasible for several reasons (impossibility of completing the edge sets $T_{(S_i,F_i,\mathcal{C}_i)}$ to a point in $P_{\mathrm{ST}}(S,F,\mathcal{C})$, infeasibility of the lower bound constraints on cuts, inconsistencies among the edge sets F_i, loops generated by edges in F_i, etc.). In such a case, we interpret the cost of a common integral extension T to be ∞, which avoids using such extensions later on.

Algorithm 4.4: Propagation to $T_{(S,F,\mathcal{C})}$ from all partial solutions $T_{(S',F',\mathcal{C}')}$ with $S' \subsetneq S$.

1. For every choice of connectivity triples $(S_i,F_i,\mathcal{C}_i) \in \mathcal{K}'$ for $i \in [w]$ where $S_1,\dots,S_w \in \mathcal{L}$ are strict subsets of S with pairwise empty intersections, apply Algorithm 4.3 to extend the trees $T_{(S_1,F_1,\mathcal{C}_1)},\dots,T_{(S_w,F_w,\mathcal{C}_w)}$ to $(S,F,\mathcal{C})$. Let $\mathcal{T}$ be the set of all trees obtained this way.
2. Return $T_{(S,F,\mathcal{C})} \in \operatorname{argmin}_{T\in\mathcal{T}} c(T)$.

Algorithm 4.4 considers all potential candidates for $T_{(S,F,\mathcal{C})}$ that were obtained through extension steps, and returns the one of minimum cost.

4.5.2 Analyzing the DP

We first show that the trees $T_{(S,F,\mathcal{C})}$ computed by Algorithm 4.3 satisfy Property 4.21. Formally, we prove this statement by induction. Obviously, the point $T_{(\emptyset,\emptyset,\{\emptyset\})}$ has all desired properties. The induction step to complete the proof is captured by the following lemma.

Lemma 4.22. *Let $(S,F,\mathcal{C}) \in \mathcal{K}$ with $S \neq \emptyset$. Assume that for all $(S',F',\mathcal{C}') \in \mathcal{K}$ with $S' \subsetneq S$, we are given $T_{(S',F',\mathcal{C}')}$ satisfying Property 4.21, and let $T_{(S,F,\mathcal{C})}$ be obtained from Algorithm 4.4. Then $T_{(S,F,\mathcal{C})}$ satisfies Property 4.21, as well.*

Proof. Let $\mathcal{T}$ be defined as in Algorithm 4.4, namely the set of all $T \subseteq E[S]$ that were obtained trough Algorithm 4.3. We already remarked earlier that any such T is a spanning tree of the corresponding graph $G(S,F,\mathcal{C})$, hence satisfying point (i) in Property 4.21. For point (ii), we prove that every $T \in \mathcal{T}$ satisfies

$$(1-\varepsilon)\cdot a_{S'} \leqslant |(T\cup F)\cap\delta(S')| \leqslant (1+\varepsilon)\cdot b_{S'}$$

for all $S' \in \mathcal{L}_S$. Indeed, for cuts S' with $S' \subsetneq S_i$ for some $i \in [w]$, we have $(T\cup F)\cap\delta(S') = (T_{(S_i,F_i,\mathcal{C}_i)}\cup F_i)\cap\delta(S_i)$, hence (4.5.2) follows from the assumption that $T_{(S_i,F_i,\mathcal{C}_i)}$ has Property 4.21. If $S' = S_i$ for some $i \in [w]$, then $(T\cup F)\cap\delta(S') = F_i$, and we have $a_{S'} \leqslant |F_i| \leqslant b_{S'}$ by definition of the connectivity pattern $(S_i,F_i,\mathcal{C}_i)$. Finally, if $S_i \subsetneq S' \subsetneq S$, then (4.5.2) is guaranteed by step 4 in Algorithm 4.3.

To see that point (iii) of Property 4.21 holds, fix an edge set $U \subseteq E[S]$ that forms a spanning tree of $G(S,F,\mathcal{C})$ and satisfies $a_{S'} \leqslant |(U\cup F)\cap\delta(S')| \leqslant b_{S'}$ for all $S' \in \mathcal{L}$ with $S' \subsetneq S$. We have to show that $c(T_{(S,F,\mathcal{C})}) \leqslant c(U)$. As a first step, consider any $T \in \mathcal{T}$ and let y be the solution of (lamExLP) that was used to obtain T. By step 4 in Algorithm 4.3, we have $c(T) \leqslant c^\top y$. It is thus enough to see that one of the linear programs (lamExLP) considered while propagating to $(S,F,\mathcal{C})$ has a solution y with $c(U) \geqslant c^\top y$.

To this end, let $R \subseteq \binom{V\setminus S}{2}$ be any set of edges that is right-compatible with $(S,F,\mathcal{C})$. Then, $T \coloneqq U\cup F\cup R$ is in the spanning tree polytope of $(V, E\cup R)$. Let $S_1,\dots,S_w \subsetneq S$ be the maximal χ^T-small cuts in $\mathcal{L}_S$, and let $F_i = T\cap\delta(S_i)$. We define connectivity patterns $\mathcal{C}_1,\dots,\mathcal{C}_w$ such that $(S_i,F_i,\mathcal{C}_i) \in \mathcal{K}$ iteratively as follows, where $T_0 \coloneqq T$ and $i \in [w]$:

$$\text{Let } \mathcal{C}_i \text{ s.t. } T_{i-1} \text{ is compatible with } (S_i,F_i,\mathcal{C}_i), \text{ and let } T_i \coloneqq \big(T_{i-1}\setminus E[S_i]\big)\cup T_{(S_i,F_i,\mathcal{C}_i)}. \tag{4.2}$$

First of all, observe that all edge sets T_i are indeed spanning trees, making the above operation well-defined. To see this, we proceed inductively and assume that T_{i-1} is a spanning tree. Compatibility of

T_{i-1} with $(S_i, F_i, \mathcal{C}_i)$ implies that $T_{i-1} \cap E[V \setminus S_i]$ is right-compatible with $(S_i, F_i, \mathcal{C}_i)$, while $T_{(S_i,F_i,\mathcal{C}_i)}$ is left-compatible with $(S_i, F_i, \mathcal{C}_i)$, by assumption. Thus, $(T_{i-1} \cap E[V \setminus S_i]) \cup F_i \cup T_{(S_i,F_i,\mathcal{C}_i)} = (T_{i-1} \setminus E[S_i]) \cup T_{(S_i,F_i,\mathcal{C}_i)} = T_i$ is indeed a spanning tree. We claim that the construction in (4.2) leads to a tree T_w with the properties

(a) $c(T) \geqslant c(T_w)$, and

(b) $T_w \cap E[S]$ is feasible for (lamExLP) when extending from $T_{(S_1,F_1,\mathcal{C}_1)}, \ldots, T_{(S_w,F_w,\mathcal{C}_w)}$ to $(S, F, \mathcal{C})$.

These two properties are enough to conclude. As T and T_w are identical outside of $E[S]$, (a) implies that

$$c(U) = c(T \cap E[S]) \geqslant c(T_w \cap E[S]) \ .$$

Moreover, (b) implies that $c(T_w \cap E[S]) \geqslant c^\top y$, which together with the previous inequality gives the desired $c(U) \geqslant c^\top y$.

To see property (a), we show that for all $i \in [w]$, we have $c(T_{i-1}) \geqslant c(T_i)$. By definition of T_i, the latter is equivalent to $c(T_{i-1} \cap E[S_i]) \geqslant c(T_i \cap E[S_i])$. But by construction, $T_{i-1} \cap E[S_i] = U \cap E[S_i]$, and $T_i \cap E[S_i] = T_{(S_i,F_i,\mathcal{C}_i)}$. Note that $U \cap E[S_i]$ is an integral solution of the subproblem on $G(S_i, F_i, \mathcal{C}_i)$, and hence Property 4.21 (iii) for $T_{(S_i,F_i,\mathcal{C}_i)}$ implies $c(U \cap E[S_i]) \geqslant c(T_{(S_i,F_i,\mathcal{C}_i)})$, which is precisely what we need. Consequently, we have $c(T_{i-1}) \geqslant c(T_i)$ for all $i \in [w]$, and hence

$$c(T) = c(T_0) \geqslant c(T_1) \geqslant \ldots \geqslant c(T_w) \ ,$$

as desired. For property (b), we check that the constraints in (lamExLP) hold for $T_w \cap E[S]$, i.e., for all $S' \in \mathcal{L}$ such that there is an $i \in [w]$ with $S_i \subsetneq S' \subsetneq S$, we prove

$$\max\{\tau + 1, a_{S'}\} \leqslant |(T_w \cap E[S]) \cap \delta(S')| + |F \cap \delta(S')| \leqslant b_{S'} \ .$$

We have $|(T_w \cap E[S]) \cap \delta(S')| + |F \cap \delta(S')| = |T_w \cap \delta(S')| = |(U \cup F) \cap \delta(S')|$, where the last equality follows from the construction of T_w. Thus, the lower bound $a_{S'}$ and the upper bound $b_{S'}$ are implied by the assumption on U. Moreover, the lower bound $\tau + 1$ follows from the definition of $S_1, \ldots, S_w$ as the maximal $\chi^{U \cup F}$-small cuts, implying (4.5.2). Finally, by construction, T_w equals $T_{(S_i,F_i,\mathcal{C}_i)}$ and F_i on $E[S_i]$ and $\delta(S_i)$, respectively, for all $i \in [w]$. Consequently, $T_w \cap E[S]$ satisfies all constraints of (lamExLP). This finishes the proof of Lemma 4.22. □

It remains to analyze the expected running time of an extension step as described in Algorithm 4.3.

Lemma 4.23. *Algorithm 4.3 has expected running time* $|V|^{\mathcal{O}(1)}$.

Proof. First of all, note that every single step of Algorithm 4.3 can be implemented in running time $|V|^{\mathcal{O}(1)}$. In particular, linear programs of the type (lamExLP) can be solved in strongly polynomial time by using a compact extended formulation for the spanning tree polytope with small coefficients in the constraint matrix (one can, for example, use the one by Martin [Mar91], which has coefficients that are bounded by 1 in absolute value), and then applying the framework of Tardos [Tar86]. Consequently, the above lemma is reduced to proving a bound on the expected number of iterations that are needed to achieve the properties required in step 4.

Replicating the analysis in the proof of Theorem 4.1, we see that Theorems 4.5 and 4.8 imply that the tree T obtained in steps 2 and 3 of Algorithm 4.3 has the desired properties with probability at least $1/2|V|$. As all iterations are independent, the probability that we succeed precisely at iteration $j \in \mathbb{Z}_{>0}$, i.e., in time $j \cdot |V|^{\mathcal{O}(1)}$, is $(1 - 1/2|V|)^{j-1} \cdot 1/2|V|$. Consequently, the expected running time is

$$\sum_{j \geqslant 1} j \cdot |V|^{\mathcal{O}(1)} \cdot \left(1 - \frac{1}{2|V|}\right)^{j-1} \cdot \frac{1}{2|V|} = |V|^{\mathcal{O}(1)} \ ,$$

where we use that $\sum_{j \geqslant 1} j(1-x)^{j-1} = \frac{1}{x^2}$ for $x \in (0, 1)$. □

Together with the bound $|\mathcal{K}| = |V|^{\mathcal{O}(\tau)}$ from Proposition 4.11, we are finally ready to prove Theorem 4.20.

Proof of Theorem 4.20. We run a dynamic program that calculates trees $T_{(S,F,\mathcal{C})}$ for all $(S, F, \mathcal{C}) \in \mathcal{K}$ starting from the initialization $T_{(\emptyset,\emptyset,\{\emptyset\})} = \emptyset$, and using Algorithm 4.4 for propagation. Note that $T_{(\emptyset,\emptyset,\{\emptyset\})}$ satisfies Property 4.21, and hence by an inductive application of Lemma 4.22, all trees $T_{(S,F,\mathcal{C})}$ satisfy Property 4.21. In particular, $T_{(V,\emptyset,\{\emptyset\})}$ is thus a tree satisfying the guarantees of Theorem 4.20.

The running time is determined by the number of calls to Algorithm 4.3. For every triple $(S, F, \mathcal{C}) \in \mathcal{K}$, when calculating $T_{(S,F,\mathcal{C})}$, there is one call to Algorithm 4.3 for every possible choice of triples $(S_1, F_1, \mathcal{C}_1), \ldots, (S_w, F_w, \mathcal{C}_w) \in \mathcal{K}$ with $S_1, \ldots, S_w \in \mathcal{L}_S$ having pairwise empty intersections. Note that $\mathcal{L}$ has width k, so there are at most k sets with pairwise empty intersection, i.e., $w \leqslant k$. This implies that the number of calls to Algorithm 4.3 is bounded from above by $|\mathcal{K}|^{k+1}$. Consequently, the bound on the expected running time of $|V|^{\mathcal{O}(k\tau)}$ follows from combining Proposition 4.11 and Lemma 4.23. □

4.6 Implications in Path TSP and beyond

In this section, we present an application of our new dynamic programming technique to Path TSP and one natural generalization of it, namely the problem of finding connected T-joins. We remark that in order to keep notation unambiguous, we reserve the variable T for trees and use Q instead in Q-joins throughout the rest of this chapter.

Observe that the solution structure in the connected Q-join problem is more general than for Path TSP, as by definition, Q-joins are edge multisets, while a path can contain every edge at most once. However, it turns out that the connected Q-join problem always has optimal solutions with simpler structure, as the following theorem shows.

Theorem 4.24 (Cheriyan, Friggstad, Gao [CFG15])**.** *Let $G = (V, E)$ be a complete graph with metric edge lengths $\ell\colon E \to \mathbb{R}_{\geqslant 0}$, and let $Q \subseteq V$ be non-empty and of even cardinality. Given a connected Q-join J, a spanning tree T of G with $\ell(T) \leqslant \ell(J)$ that is a connected Q-join can be found efficiently.*

The proof of Theorem 4.24 exploits the assumption that the instance is metric, and shows that whenever a Q-join has parallel edges or cycles, shortcutting allows for a reduction. In particular, Theorem 4.24 shows that indeed, Path TSP is a special case of the connected Q-join problem, as for $Q = \{s, t\}$ with $s \neq t$, a spanning tree that is a connected Q-join is a Hamiltonian s-t path. Moreover, by Theorem 4.24, we also see that it is enough for an approximate solution to compare well to optimal spanning trees, which is a crucial observation for simplifying our analysis.

With the above said, we give a short recap of the approach by Christofides' for TSP and Path TSP [Chr76] in the context of the more general shortest connected Q-join problem. More precisely, we walk through a polyhedral analysis of Christofides' Algorithm due to Wolsey [Wol80]. These ideas form the basis of essentially all improvements in approximation algorithms for Path TSP over the last few years [AKS15; GV16; SZ16; TV18; Vyg16], culminating in a recent $3/2$-approximation algorithm for Path TSP by Zenklusen [Zen19] that was later beaten by combining the result of Karlin, Klein, and Gharan [KKG21], who provide an approximation factor of $3/2 - \delta$ for some $\delta > 10^{-36}$ for TSP, and the result of Traub, Vygen, and Zenklusen [TVZ20] that reduces Path TSP to TSP at an arbitrarily small loss in the approximation factor.

4.6.1 Christofides' algorithm and Wolsey's analysis

Christofides' algorithm builds on the observation that a solution has to satisfy two properties, namely connectivity and correct degree parities. Connectivity can be guaranteed by starting with a spanning tree T. The degree parities of T are wrong precisely at the vertices in $Q_T \coloneqq \operatorname{odd}(T) \mathbin{\triangle} Q$, which can be corrected by adding a Q_T-join J to T.[10] The multiset obtained by combining T and J can be shortcut to a solution

[10] By $\operatorname{odd}(T)$, we denote the set of odd-degree vertices in T.

of the problem, and approximation guarantees follow by choosing T and J such that $\ell(T)$ and $\ell(J)$ can be bounded in terms of $\ell(\mathrm{OPT})$, where OPT denotes an optimal solution of the problem.

While it is easy to observe that for a shortest spanning tree T, we have $\ell(T) \leqslant \ell(\mathrm{OPT})$, bounds on $\ell(J)$ for a shortest Q_T-join J can for example be obtained by exploiting polyhedral descriptions of Q-joins. In particular, the dominant of the Q-join polytope[11] is given by

$$P^{\uparrow}_{Q\text{-join}} \coloneqq \left\{ x \in \mathbb{R}^E_{\geqslant 0} \,\middle|\, x(\delta(C)) \geqslant 1 \,\forall\, Q\text{-cuts } C \subseteq V \right\} \ ,$$

where a *Q-cut* C is a subset of V with $|C \cap Q|$ odd (see [Sch03, Section 29]). By integrality of this polytope, in order to prove a bound of the form $\ell(J) \leqslant \gamma \cdot \ell(\mathrm{OPT})$ for a shortest Q_T-join J, it is sufficient to find a point $z \in P^{\uparrow}_{Q_T\text{-join}}$ with $\ell^\top z \leqslant \gamma \cdot \ell(\mathrm{OPT})$, and a $(1+\gamma)$-approximation follows.

In many approaches, an important role for finding a suitable point z is taken by a linear relaxation of the problem. For the connected Q-join problem, we use the formulation

$$\begin{aligned} \min \quad & \ell^\top x & \\ & x(\delta(C)) \geqslant 2 \quad \forall C \subsetneq V,\ C \neq \emptyset,\ |C \cap Q| \text{ even} \\ & x(\delta(C)) \geqslant 1 \quad \forall C \subseteq V,\ |C \cap Q| \text{ odd} \\ & x \in \mathbb{R}^E_{\geqslant 0} \ , \end{aligned} \tag{$\mathrm{LP}_{\mathrm{HK}}$}$$

which is an adaptation of the well-known Held-Karp relaxation for TSP.

If x^* is an optimal solution of ($\mathrm{LP}_{\mathrm{HK}}$), the point $z = {}^{x^*}\!/_2$ is a good candidate for a feasible point of $P^{\uparrow}_{Q_T\text{-join}}$. More precisely, observe that the constraint $z(\delta(C)) \geqslant 1$ is violated precisely for cuts C with $x^*(\delta(C)) < 2$. As $\delta(C) = \delta(V \setminus C)$, we can fix a vertex $q \in Q$, and it is enough to consider the cuts in the family

$$\mathcal{N} \coloneqq \{C \subseteq V \mid q \notin C,\ x^*(\delta(C)) < 2\} \ ,$$

the so-called *narrow* cuts of x^*. By the first constraint in ($\mathrm{LP}_{\mathrm{HK}}$), only cuts C with $|C \cap Q|$ odd can be narrow. Moreover, note that the constraints $z(\delta(C)) \geqslant 1$ only appear in the description of $P^{\uparrow}_{Q_T\text{-join}}$ if C is a Q_T-cut, but this is not necessarily the case for all $C \in \mathcal{N}$. If indeed, none of the narrow cuts are Q_T-cuts, we conclude that $z = {}^{x^*}\!/_2$ is feasible for $P^{\uparrow}_{Q_T\text{-join}}$. Using that $\ell^\top x^* \leqslant \ell(\mathrm{OPT})$, we get

$$\ell(T) + \ell(J) \leqslant {}^3\!/_2 \cdot \ell(\mathrm{OPT}) \ ,$$

and hence a ${}^3\!/_2$-approximation. In particular, if we could obtain an optimal solution x^* of ($\mathrm{LP}_{\mathrm{HK}}$) and a tree T^* with $\ell(T^*) \leqslant \ell(\mathrm{OPT})$ that has an odd number of edges in every narrow cut of x^*, we could achieve the above result. To see this, consider a narrow cut C, and observe that

$$\sum_{v \in C} \deg_{T^*}(v) = 2 \cdot |T^* \cap E[C]| + |T^* \cap \delta(C)| \ .$$

The assumption that $|T^* \cap \delta(C)|$ is odd implies that T^* has an odd number of odd-degree vertices in C. As $|C \cap Q|$ is odd, we conclude that $|C \cap Q_{T^*}| = |C \cap (\mathrm{odd}(T^*) \triangle Q)|$ is even, i.e., C is indeed not a Q_{T^*}-cut.

Ideally, we would thus like to find a spanning tree T^* that has an odd number of edges in each narrow cut $C \in \mathcal{N}$. Cheriyan, Friggstad, and Gao [CFG15] showed that the family $\mathcal{N}$ is in fact a laminar family. We additionally observe that the width of this family is at most $|Q| - 1$: Every cut $C \in \mathcal{N}$ has odd intersection with Q, i.e., it contains at least one element of $Q \setminus \{q\}$, so there can be at most $|Q| - 1$ many mutually disjoint sets in $\mathcal{N}$. However, even for a chain, one can see that it is NP-hard to decide whether there is a spanning

[11] The dominant of the Q-join polytope is the set of all points $x \in \mathbb{R}^E$ such that there is a convex combination $y = \sum_{i=1}^k \lambda_i \chi^{J_i}$ of characteristic vectors $\chi^{J_i} \in \{0,1\}^E$ of Q-joins J_i with $\lambda_i > 0$ for $i \in [k]$ such that $y \leqslant x$.

tree that is odd in each cut.[12] As an alternative, we can also use a tree T with slightly weaker properties, and instead modify the vector z to obtain a feasible point for $P^{\uparrow}_{Q_T\text{-join}}$. In particular, observe the following.

Observation 4.25. *Assume that we are given an optimal solution x^* of ($\mathrm{LP}_{\mathrm{HK}}$), a tree T and a point $y \in \mathbb{R}^E_{\geqslant 0}$ with the following properties.*

(i) For all narrow cuts C of x^*, either $|T \cap \delta(C)|$ is odd, or $y(\delta(C)) \geqslant 1/\varepsilon$.
(ii) $\ell^\top y \leqslant \ell(\mathrm{OPT})$, and $\ell(T) \leqslant \ell(\mathrm{OPT})$.

Then, shortcutting the combination of T and a shortest Q_T-join J gives a $(1.5+\varepsilon)$-approximate solution for the connected Q-join problem.

Proof. We claim that $z \coloneqq \frac{1}{2}(x^* + \varepsilon y) \in \mathbb{R}^E_{\geqslant 0}$ satisfies $z \in P^{\uparrow}_{T_Q\text{-join}}$. From the above discussion and the bounds in (ii), this immediately implies the observation. The constraints of the form $z(\delta(C)) \geqslant 1$ are obviously satisfied for non-narrow cuts C of x^*, and we saw that they do not appear in the description of $P^{\uparrow}_{Q_T\text{-join}}$ for narrow cuts C if $|T \cap \delta(C)|$ is odd. For the remaining narrow cuts C, property (i) implies $y(\delta(C)) \geqslant 1/\varepsilon$, and hence $z(\delta(C)) \geqslant \frac{1}{2}(1 + \varepsilon \cdot 1/\varepsilon) = 1$. □

The close relation of y and T that is required in Observation 4.25 motivates studying τ-odd solutions, which to some extent embrace properties of y and T.

Definition 4.26 (τ-odd)**.** *For $\tau \in \mathbb{Z}_{\geqslant 0}$ such that τ is odd, and a family $\mathcal{F}$ of cuts, we say that a point $y \in \mathbb{R}^E$ is* τ-odd *(with respect to $\mathcal{F}$), if for each $C \in \mathcal{F}$, either*

(i) $y(\delta(C)) \leqslant \tau$, y is integral on the edges in $\delta(C)$, and $y(\delta(C))$ is odd, or
(ii) $y(\delta(C)) \geqslant \tau + 2$.

We call the cuts C satisfying (i) and (ii), respectively, the y-small *and* y-large *cuts.*

Note that, for $\tau \approx 1/\varepsilon$, given a short τ-odd point $y \in P_{\mathrm{ST}}$ with respect to the narrow cuts $\mathcal{N}$ of an optimal solution x^* of ($\mathrm{LP}_{\mathrm{HK}}$), a tree with the properties needed in Observation 4.25 could be obtained as a minimum length spanning tree T such that χ^T coincides with y on the integral edges of y. This reduces the problem to finding short τ-odd points, which is where our dynamic programming approach can help.

4.6.2 Obtaining τ-odd points via our DP

We now discuss how our dynamic programming approach can be adjusted to compute τ-odd points. Note that τ-odd points are by definition very similar to τ-integral points: The extra constraint is essentially that a τ-odd point can only have an odd number of edges in small cuts. Our dynamic programming approach can easily handle this type of constraints, as edges in small cuts are always determined by the connectivity triples used. Thus, if we define the set

$$\mathcal{K}' \coloneqq \big\{(S,F,\mathcal{C}) \in \mathcal{K} \,\big|\, |F| \text{ odd}\big\} \cup \big\{(\emptyset,\emptyset,\{\emptyset\}),(V,\emptyset,\{\emptyset\})\big\}$$

and run the dynamic program presented in Section 4.3 with $\mathcal{K}'$ instead of $\mathcal{K}$ (and no lower or upper bounds on the cuts), we immediately obtain the following analogue of Theorem 4.7.

Theorem 4.27. *Let $\mathcal{C}$ be a chain of cuts. For any $\tau \in \mathbb{Z}_{\geqslant 0}$, there is an algorithm that returns in $|V|^{\mathcal{O}(\tau)}$ time a τ-odd point $y \in P_{\mathrm{ST}}$ with respect to $\mathcal{C}$ such that $c^\top y \leqslant c(T)$ for every spanning tree T that has an odd number of edges in every cut in $\mathcal{C}$.*

[12] NP-hardness can for example be derived by a reduction from the Hamiltonian s-t path problem. For this consider an arbitrary numbering of the vertices $V = \{v_1, \dots, v_n\}$ with $v_1 = s$ and $v_n = t$, and consider the complete chain $S_1, \dots, S_{n-1}$, where $S_i = \{v_1, \dots, v_i\}$ for $i \in [n-1]$. First, the cuts S_1 and S_{n-1} ensure that a spanning tree that is odd in each cut must have odd degree at v_1 and v_n. Moreover, for any $i \in \{2, \dots, n-1\}$, the vertex v_i must have even degree in the spanning tree due to the cut constraints on S_{i-1} and S_i. Because the degrees of a spanning tree on n vertices sum up to $2(n-1)$, this implies that v_1 and v_n must have degree 1, and all other vertices degree 2. However, a spanning tree with these properties is a Hamiltonian v_1-v_n path, and any Hamiltonian v_1-v_n path is such a spanning tree.

The above is precisely what we need for Path TSP: In this special case, we have $Q = \{s, t\}$, hence the family $\mathcal{N}$ of narrow cuts has width 1, i.e., it is a chain. This was also observed earlier by An, Kleinberg and Shmoys [AKS15] and has been crucial in many recent improvements for Path TSP. In our case, it guarantees applicability of Theorem 4.27, and allows the approximation algorithm for Path TSP stated as Algorithm 4.5, which we analyze to prove Theorem 4.3 below.

Algorithm 4.5: Polynomial $(3/2 + \varepsilon)$-approximation for Path TSP

1. Let x^* be an optimal solution of ($\mathrm{LP}_{\mathrm{HK}}$), and let $\mathcal{C}$ be the family of narrow cuts of x^* not containing t.
2. Let $\tau \in \{\lfloor 1/\varepsilon \rfloor, \lfloor 1/\varepsilon \rfloor + 1\}$ odd, and use the algorithm guaranteed by Theorem 4.27 to find a τ-odd point $y \in P_{\mathrm{ST}}$ with respect to $\mathcal{C}$.
3. Let T be the shortest spanning tree of G such that χ^T coincides with y on all integral edges of y.
4. Let J be a shortest Q_T-join in G, and return the shortcutted multiunion of J and T.

Proof of Theorem 4.3. We claim that the pair (y, T) generated in Algorithm 4.5 has the properties listed in Observation 4.25. To see that point (i) holds, first note that by τ-oddness of y with respect to $\mathcal{C}$, we have that for every narrow cut C of x^*, either $y(\delta(C)) \geqslant \tau + 2 \geqslant 1/\varepsilon$, or y is integral on $\delta(C)$ and $y(\delta(C))$ is odd. As χ^T coincides with y on all integral edges of y, the latter case in particular implies that χ^T coincides with y on $\delta(C)$, and consequently, $|T \cap \delta(C)|$ is odd. Hence, point (i) is satisfied. For point (ii), observe that by definition, $c(T) \leqslant c^\top y$, and Theorem 4.27 implies $c^\top y \leqslant c(\mathrm{OPT})$. Together, this yields $c(T) \leqslant c(\mathrm{OPT})$, as desired.

By Observation 4.25, it follows that Algorithm 4.5 is a $(1.5 + \varepsilon)$-approximation for Path TSP. For a running time guarantee, we observe that all steps in Algorithm 4.5 can be performed efficiently. For step 1, note that the minimum cut in G with respect to weights x^* has value at least 1 by the lower bounds in the relaxation ($\mathrm{LP}_{\mathrm{HK}}$), and enumerating all cuts with values that are within a factor of 2 from the minimum cut can be done in time $\mathcal{O}(|V|^4|E|)$ (see [NNI97]). Step 2 can be done in time $|V|^{\mathcal{O}(1/\varepsilon)}$ by Theorem 4.27. Finally, minimum length Q-joins can be obtained efficiently (see [EJ73], for example), with running times dominated by the second step. Thus, the running time of Algorithm 4.5 is $|V|^{\mathcal{O}(1/\varepsilon)}$. This completes the proof of Theorem 4.4. □

The above approach exploits that the τ-odd point y lies in the spanning tree polytope: This guarantees that a spanning tree T coinciding with y on integral edges can be found in step 3 of Algorithm 4.5. In the more general case of connected Q-joins with $|Q| > 2$, the narrow cuts $\mathcal{N}$ no longer form a chain, which causes problems in directly extending the propagation step of our dynamic programming approach, as we highlighted in the generalization from MCCST to MLCST. With minor modifications, ideas of this generalization also help for the connected Q-join problem. In fact, a simple alteration of the extension step can be used to obtain a pair (y, T) with the properties highlighted in the following theorem.

Theorem 4.28. *For any odd $\tau \in \mathbb{Z}_{\geqslant 0}$ and a laminar family $\mathcal{L}$ of width k, there is an algorithm that returns in time $|V|^{\mathcal{O}(k\tau)}$ a point $y \in \mathbb{R}^E_{\geqslant 0}$ and a spanning tree T of G with the following properties:*

(i) y is τ-odd with respect to $\mathcal{L}$.
(ii) χ^T coincides with y on all y-small cuts in $\mathcal{L}$.
(iii) $\ell(T) \leqslant \ell^\top y \leqslant \ell^\top x$ for every τ-odd point $x \in P_{\mathrm{ST}} \cap \mathbb{Z}^E_{\geqslant 0}$.

Observe that Theorem 4.28 allows for bounding the length of the τ-good points that we find from above by the length of τ-good spanning trees only, but by Theorem 4.24, this is sufficient for comparing to optimal solutions of the Q-join problem, once we prove that connected Q-joins that are spanning trees are indeed τ-good. From the above ingredients, we obtain our approximation algorithm for the connected Q-join problem, which is stated as Algorithm 4.6 below. It is this algorithm that we analyze for a proof of Theorem 4.4.

Algorithm 4.6: Polynomial $(3/2+\varepsilon)$-approximation for the shortest connected Q-join problem

1. Let x^* be an optimal solution of ($\mathrm{LP}_{\mathrm{HK}}$), and let $\mathcal{N}$ be the family of all narrow cuts of x^* not containing a fixed element $q \in Q$.
2. Let $\tau \in \{\lfloor 1/\varepsilon \rfloor, \lfloor 1/\varepsilon \rfloor + 1\}$ be odd, and use the algorithm guaranteed by Theorem 4.28 to find a pair (y, T) of a τ-odd point $y \in \mathbb{R}^E_{\geqslant 0}$ and a spanning tree T of G.
3. Let J be a shortest Q_T-join in G, and return the multiunion of J and T.

Proof of Theorem 4.4. As in the proof of Theorem 4.3, it is easy to see that the pair (y, T) returned by Algorithm 4.6 satisfies point (i) in Observation 4.25. To prove that point (ii) holds, we claim that for any spanning tree S that is a Q-join, χ^S is a τ-good solution of P_{ST} with respect to $\mathcal{N}$. By Theorem 4.28, this implies that $\ell^\top y \leqslant \ell(S)$ for all spanning trees S that are Q-joins. By Theorem 4.24, at least one such spanning tree is in fact an optimal solution to the Q-join problem, and thus $\ell^\top y \leqslant \ell(\mathrm{OPT})$ follows. Furthermore, Theorem 4.28 also implies $\ell(T) \leqslant \ell^\top y \leqslant \ell(\mathrm{OPT})$.

For concluding the approximation guarantee of $3/2+\varepsilon$ with the help of Observation 4.25, it is thus sufficient to prove the claim. Thereto, consider a spanning tree S that is a Q-join, and let $C \in \mathcal{N}$. We show that $\chi^S(\delta(C))$ is odd, which implies the claim due to integrality of χ^S. Double counting the edges in $S \cap E[C]$, we get

$$\chi^S(\delta(C)) = |S \cap \delta(C)| = \sum_{v \in C} \deg_S(v) - 2 \cdot |S \cap E[C]| \ .$$

Consequently, $\chi^S(\delta(C))$ has the same parity as the number of odd-degree vertices in C with respect to S. But as S is a Q-join, the latter number is equal to $|C \cap Q|$, and as we already observed previously, all $C \in \mathcal{N}$ satisfy that $|C \cap Q|$ is odd. This finishes the proof of the claim.

Finally, in order to obtain a running time bound, note that step 1 can be done in time $\mathcal{O}(|V|^4|E|)$ (see the proof of Theorem 4.3). The second step has running time $|V|^{\mathcal{O}(k/\varepsilon)}$ by Theorem 4.28, and minimum length Q-joins can be obtained efficiently (see [EJ73], for example), with running times dominated by the second step. Thus, the running time of Algorithm 4.6 is bounded by $|V|^{\mathcal{O}(|Q|/\varepsilon)}$. This completes the proof of Theorem 4.4. □

It thus remains to give a proof of Theorem 4.28, which we outline in the remainder of this section. We adopt the dynamic programming approach used for MLCST, where in order to make sure that the DP guesses an odd number of edges in small cuts, we use $\mathcal{K}'$ instead of $\mathcal{K}$ throughout the procedure. For every connectivity triple $(S, F, \mathcal{C}) \in \mathcal{K}'$, the dynamic programming approach will construct a pair $(y_{(S,F,\mathcal{C})}, T_{(S,F,\mathcal{C})}) \in \mathbb{R}^E_{\geqslant 0} \times 2^E$ with the following property.

Property 4.29.

(i) $\mathrm{supp}(y_{(S,F,\mathcal{C})}) \subseteq E[S]$, *and* $y_{(S,F,\mathcal{C})}$ *is* τ*-odd with respect to* $\mathcal{L}_S$.
(ii) $\chi^{T_{(S,F,\mathcal{C})}} \in P_{\mathrm{ST}}(S, F, \mathcal{C})$, *and* $\chi^{T_{(S,F,\mathcal{C})}}$ *coincides with* $y_{(S,F,\mathcal{C})}$ *on all* $y_{(S,F,\mathcal{C})}$*-small cuts in* $\mathcal{L}_S$.
(iii) For all $U \subseteq E[S]$ *such that* $\chi^U \in P_{\mathrm{ST}}(S, F, \mathcal{C})$ *and* $\chi^U + \chi^F$ *is* τ*-odd with respect to* $\mathcal{L}_S$, *we have* $\ell(T_{(S,F,\mathcal{C})}) \leqslant \ell^\top y_{(S,F,\mathcal{C})} \leqslant \ell(U)$.

It is clear that if we can construct such pairs for all triples $(S, F, \mathcal{C}) \in \mathcal{K}'$, then we are done, as $(y_{(V,\emptyset,\{\emptyset\})}, T_{(V,\emptyset,\{\emptyset\})})$ satisfies the assumptions of Observation 4.25. To maintain pairs (y, T) in the dynamic program, we replace the extension step (Algorithm 4.3) by the one presented in Algorithm 4.7 below.

Algorithm 4.7: Extending $(y_{(S_1,F_1,\mathcal{C}_1)}, T_{(S_1,F_1,\mathcal{C}_1)}), \dots, (y_{(S_w,F_w,\mathcal{C}_w)}, T_{(S_w,F_w,\mathcal{C}_w)})$ to $(S,F,\mathcal{C})$.

1. Let z be a minimizer of the linear program

$$\begin{aligned} \min \quad & \ell^\top z && && (\text{lamExLP}_2)\\ & z \in P_{\mathrm{ST}}(S,F,\mathcal{C}) && \\ z(\delta(S')) + |F \cap \delta(S')| &\geqslant \tau + 2 && \forall S \in \mathcal{L} \text{ such that } \exists i \in [w] \text{ with } S_i \subsetneq S' \subsetneq S \\ z(e) &= \chi^{T_{(S_i,F_i,\mathcal{C}_i)}}(e) && \forall e \in E[S_i],\ \forall i \in [w] \\ z(e) &= \chi^{F_i}(e) && \forall e \in \delta(S_i) \setminus F,\ \forall i \in [w]\ . \end{aligned}$$

2. Define $y \in \mathbb{R}^E_{\geqslant 0}$ by $y(e) = \begin{cases} y_{(S_i,F_i,\mathcal{C}_i)}(e) & \text{if } e \in E[S_i] \text{ for some } i \in [w], \\ z(e) & \text{else.} \end{cases}$
3. Let $T \subseteq E[S]$ be of minimum length $\ell(T)$ such that
   (i) $\chi^T \in P_{\mathrm{ST}}(S,F,\mathcal{C})$, (ii) $T \cap E[S_i] = T_{(S_i,F_i,\mathcal{C}_i)}\ \forall i \in [w]$, and (iii) $T \cap \delta(S_i) = F_i\ \forall i \in [w]$.
4. Output (y,T).

The idea of this modified extension step is to maintain both a fractional point and a spanning tree at every stage, where (as in the dynamic program for MLCST) extension steps using the linear program are always based on trees. A new fractional point is then obtained by combining the potentially fractional extension with the fractional solutions of the subproblems (see step 2). Opposed to the situation in MLCST, the application here only requires spanning trees that have an odd number of edges in small cuts of the corresponding fractional point. This is easily achieved by the construction of the trees in step 3. This construction guarantees that every pair (y,T) returned by Algorithm 4.7 satisfies the first two points of Property 4.29 with respect to the connectivity triple $(S,F,\mathcal{C})$, which we formally prove in Lemma 4.30.

For propagation, we use Algorithm 4.8, which is an analogon of Algorithm 4.4. It considers all potential candidates for $(y_{(S,F,\mathcal{C})}, T_{(S,F,\mathcal{C})})$ that can be obtained through extension steps, and the shortest pair (y,T) with respect to $\ell^\top y$ is returned.

Algorithm 4.8: Propagation to $(y_{(S,F,\mathcal{C})}, T_{(S,F,\mathcal{C})})$ from all $(y_{(S',F',\mathcal{C}')}, T_{(S',F',\mathcal{C}')})$ with $S' \subsetneq S$.

1. For every choice of triples $(S_i,F_i,\mathcal{C}_i) \in \mathcal{K}'$ for $i \in [w]$ where $S_1, \dots, S_w \subsetneq S$ have pairwise empty intersections, apply Algorithm 4.7 to extend $(y_{(S_1,F_1,\mathcal{C}_1)}, T_{(S_1,F_1,\mathcal{C}_1)}), \dots,$ $(y_{(S_w,F_w,\mathcal{C}_w)}, T_{(S_w,F_w,\mathcal{C}_w)})$ to $(S,F,\mathcal{C})$. Collect all pairs (y,T) obtained this way in the set $\mathcal{P}$.
2. Return $(y_{(S,F,\mathcal{C})}, T_{(S,F,\mathcal{C})}) \in \operatorname{argmin}_{(y,T)\in\mathcal{P}} \ell^\top y$.

We now show that the pairs (y,T) returned by Algorithm 4.8 have Property 4.29. Adapting the approach pursued in the case of MLCST (Lemma 4.22), we proceed by induction. Obviously, the pair $(y_{(\emptyset,\emptyset,\{\emptyset\})}, T_{(\emptyset,\emptyset,\{\emptyset\})}) = (0,\emptyset)$ satisfies Property 4.29, and the inductive step is given by the following lemma.

Lemma 4.30. *Let $(S,F,\mathcal{C}) \in \mathcal{K}'$ with $S \neq \emptyset$. Assume that for all $(S',F',\mathcal{C}') \in \mathcal{K}'$ with $S' \subsetneq S$, we are given $(y_{(S',F',\mathcal{C}')}, T_{(S',F',\mathcal{C}')})$ satisfying Property 4.29, and let $(y_{(S,F,\mathcal{C})}, T_{(S,F,\mathcal{C})})$ be obtained from Algorithm 4.8. Then $(y_{(S,F,\mathcal{C})}, T_{(S,F,\mathcal{C})})$ satisfies Property 4.29, as well.*

Proof. Let $\mathcal{P}$ be defined as in Algorithm 4.8. We already remarked above that all pairs $(y,T) \in \mathcal{P}$ satisfy point (i) and point (ii) of Property 4.29 with respect to the connectivity triple $(S,F,\mathcal{C})$. Indeed, $\operatorname{supp}(y_{(S,F,\mathcal{C})}) \subseteq E[S]$ holds by definition, as well as $\chi^{T_{(S,F,\mathcal{C})}} \in P_{\mathrm{ST}}(S,F,\mathcal{C})$. Additionally, the facts that $y_{(S,F,\mathcal{C})}$ is τ-odd with respect to $\mathcal{L}_S$ and that $\chi^{T_{(S,F,\mathcal{C})}}$ coincides with $y_{(S,F,\mathcal{C})}$ on all $y_{(S,F,\mathcal{C})}$-small cuts in $\mathcal{L}_S$ are true by definition for cuts $L \in \mathcal{L}_S$ with $S_i \subsetneq L$ for some $i \in [w]$; and they are implied by the assumptions on $(y_{(S_i,F_i,\mathcal{C}_i)}, T_{(S_i,F_i,\mathcal{C}_i)})$ through Property 4.29 for cuts $L \in \mathcal{L}_S$ with $L \subsetneq S_i$ for some $i \in [w]$.

Moreover, for any pair $(y,T) \in \mathcal{P}$ we also have $\ell(T) \leqslant \ell^\top y$: If z is the solution of the linear program (lamExLP$_2$) that was used to define y, then $\ell(T) \leqslant \ell^\top z$ because χ^T is in fact an optimal solution of the same linear program without the constraints $z(\delta(S)) \geqslant \tau+2$. As by assumption, $\ell(T_{(S_i,F_i,\mathcal{C}_i)}) \leqslant \ell^\top y_{(S_i,F_i,\mathcal{C}_i)}$, we further see that $\ell^\top z \leqslant \ell^\top y$, and hence $\ell(T) \leqslant \ell^\top y$, which is the first statement in point (iii).

Thus, it remains to prove that there exists a pair $(y,T) \in \mathcal{P}$ such that $\ell^\top y \leqslant \ell(U)$ for every set $U \subseteq E[S]$ such that $\chi^U \in P_{\mathrm{ST}}(S,F,\mathcal{C})$ and $\chi^U + \chi^F$ is τ-odd with respect to $\mathcal{L}_S$. Fix such a set U, and let $R \subseteq \binom{V\setminus S}{2}$ be any set of edges that is right-compatible with $(S,F,\mathcal{C})$. Then, $T \coloneqq U \cup F \cup R$ is in the spanning tree polytope of $(V, E \cup R)$. Let $S_1,\ldots,S_w \subsetneq S$ be the maximal χ^T-small cuts in $\mathcal{L}_S$, and let $F_i = T \cap \delta(S_i)$. We define connectivity patterns $\mathcal{C}_1,\ldots,\mathcal{C}_w$ such that $(S_i,F_i,\mathcal{C}_i) \in \mathcal{K}'$ iteratively as follows, where $T_0 \coloneqq T$ and $i \in [w]$.

$$\text{Let } \mathcal{C}_i \text{ s.t. } T_{i-1} \text{ is compatible with } (S_i,F_i,\mathcal{C}_i), \text{ and let } T_i \coloneqq \big(T_{i-1} \setminus E[S_i]\big) \cup T_{(S_i,F_i,\mathcal{C}_i)}. \tag{4.3}$$

We claim that if extending $(y_{(S_1,F_1,\mathcal{C}_1)}, T_{(S_1,F_1,\mathcal{C}_1)}), \ldots, (y_{(S_w,F_w,\mathcal{C}_w)}, T_{(S_w,F_w,\mathcal{C}_w)})$ to $(S,F,\mathcal{C})$ using Algorithm 4.7 returns the pair (y,T), then $\ell^\top y \leqslant \ell(U)$. To this end, observe that if z is the solution of the linear program (lamExLP$_2$) used in this call to Algorithm 4.7, then we can write

$$\ell^\top y = \sum_{i\in[w]} \ell^\top y_{(S_i,F_i,\mathcal{C}_i)} + \ell^\top z - \sum_{i\in[w]} \ell(T_{(S_i,F_i,\mathcal{C}_i)}) \ . \tag{4.4}$$

We will bound the right-hand side by $\ell(U)$. Thereto, we claim that by the construction in (4.3), we have $\ell^\top y_{(S_i,F_i,\mathcal{C}_i)} \leqslant \ell(U \cap E[S_i])$. This follows from invoking Property 4.29 (iii) for the pair $(y_{(S_i,F_i,\mathcal{C}_i)}, T_{(S_i,F_i,\mathcal{C}_i)})$ (which is satisfied by assumption) with the edge set $U \cap E[S_i]$. To this end, we have to show that (a) $\chi^{U\cap E[S_i]} \in P_{\mathrm{ST}}(S_i,F_i,\mathcal{C}_i)$, and (b) $\chi^{U\cap E[S_i]} + \chi^{F_i}$ is τ-odd with respect to $\mathcal{L}_{S_i}$. Indeed, (a) follows from the fact that T_{i-1} is compatible with $(S_i,F_i,\mathcal{C}_i)$, hence $\chi^{T_{i-1}\cap E[S_i]} \in P_{\mathrm{ST}}(S_i,F_i,\mathcal{C}_i)$, and $T_{i-1} \cap E[S_i] = U \cap E[S_i]$ by construction; (b) follows from $\chi^U + \chi^F$ being τ-odd with respect to $\mathcal{L}_S$, as $S_i \subsetneq S$. Furthermore, note that T_w is compatible with $(S,F,\mathcal{C})$, hence $T_w \cap E[S]$ is left-compatible with $(S,F,\mathcal{C})$, i.e., $\chi^{T_w\cap E[S]} \in P_{\mathrm{ST}}(S,F,\mathcal{C})$. Moreover, we can write

$$T_w \cap E[S] = \Big(U \setminus \textstyle\bigcup_{i\in[w]} E[S_i]\Big) \cup \bigcup_{i\in[w]} T_{(S_i,F_i,\mathcal{C}_i)} \ ,$$

and from this form and the fact that $S_1,\ldots,S_w$ are maximal χ^T-small cuts in $\mathcal{L}_S$, it can be seen that $T_w \cap E[S]$ is an integral solution of (lamExLP$_2$). As z is an optimal fractional solution of the same linear program, we have $\ell^\top z \leqslant \ell(T_w \cap E[S])$. Applying the inequalities that we just obtained to (4.4), we get

$$\begin{aligned}
\ell^\top y &\leqslant \sum_{i\in[w]} \ell(U \cap E[S_i]) + \ell(T_w \cap E[S]) - \sum_{i\in[w]} \ell(T_{(S_i,F_i,\mathcal{C}_i)}) \\
&= \sum_{i\in[w]} \ell(U \cap E[S_i]) + \ell\Big(U \setminus \textstyle\bigcup_{i\in[w]} E[S_i]\Big) \\
&= \ell(U) \ ,
\end{aligned}$$

as desired. □

In order to bound the running time of our dynamic program, we need an upper bound on the number of connectivity triples in $\mathcal{K}'$, but this is easily obtained from the upper bound $|\mathcal{K}| \leqslant |V|^{\mathcal{O}(\tau)}$ in Proposition 4.11: We have $\mathcal{K}' \subseteq \mathcal{K}$, and thus also $|\mathcal{K}'| \leqslant |V|^{\mathcal{O}(\tau)}$. With these ingredients, we are finally ready to formally prove Theorem 4.28.

Proof of Theorem 4.28. We run a dynamic program that calculates pairs $(y_{(S,F,\mathcal{C})}, T_{(S,F,\mathcal{C})})$ for all $(S,F,\mathcal{C}) \in \mathcal{K}'$ starting with the initialization $(y_{(\emptyset,\emptyset,\{\emptyset\})}, T_{(\emptyset,\emptyset,\{\emptyset\})}) = (0,\emptyset)$, and using Algorithm 4.8 for propagation in an order such that $(y_{(S',F',\mathcal{C}')}, T_{(S',F',\mathcal{C}')})$ is computed before $(y_{(S,F,\mathcal{C})}, T_{(S,F,\mathcal{C})})$ if $S' \subsetneq S$. Note that

$(y_{(\emptyset,\emptyset,\{\emptyset\})}, T_{(\emptyset,\emptyset,\{\emptyset\})})$ satisfies Property 4.29, and hence by an inductive application of Lemma 4.30, all pairs $(y_{(S,F,\mathcal{C})}, T_{(S,F,\mathcal{C})})$ satisfy Property 4.29. In particular, $(y_{(V,\emptyset,\{\emptyset\})}, T_{(V,\emptyset,\{\emptyset\})})$ is thus a pair satisfying the guarantees of Theorem 4.28.

In terms of running time, the dominating operation is repeatedly solving linear programs of the type (lamExLP$_2$). The total number of linear programs that we have to solve during this procedure is bounded from above by $|\mathcal{K}'|^{k+1}$, and the running time of $|V|^{\mathcal{O}(k\tau)}$ thus follows from $\mathcal{K}' \subseteq \mathcal{K}$ and Proposition 4.11, as remarked above, and the fact that linear programs of the type (lamExLP$_2$) can be solved in strongly polynomial time by using a compact extended formulation for the spanning tree polytope with small coefficients in the constraint matrix (one can, for example, use the one by Martin [Mar91], which has coefficients that are bounded by 1 in absolute value), and then applying the framework of Tardos [Tar86]. □

Appendix 4.A Weakness of the natural relaxation

In this section, we demonstrate that the natural relaxation of MCCST, which is given by

$$Q = \{x \in P_{\mathrm{ST}} \mid a_i \leqslant x(\delta(S_i)) \leqslant b_i \ \forall i \in [k]\} \quad ,$$

is too weak for allowing small bounds on constraint violation when comparing an integral solution to fractional solutions of the relaxation. More precisely, we show the following.

Theorem 4.31. *For every $\varepsilon > 0$, there is an instance of the MCCST problem such that there exists a point $y \in Q$, but for any tree T satisfying the chain constraints, there is a cut C in the chain such that*

$$|T \cap \delta(C)| \geqslant (2-\varepsilon) \cdot y(\delta(C)) \quad .$$

Proof. We construct a family of instances of the MCCST problem depending on a parameter $k \in \mathbb{Z}_{>0}$, where each instance has the properties listed in Theorem 4.31, but with a factor $2 - {}^{2}/_{(k+2)}$ instead of $2-\varepsilon$. This is obviously enough to conclude the theorem because for a fixed $\varepsilon > 0$ and large enough k, we obtain an instance with the desired properties.

For $k \in \mathbb{Z}_{>0}$, let H_k be the graph obtained as follows. Start with a path of length 2^{k-1} on vertices $v_0, v_1, \ldots, v_{2^{k-1}}$, and for all $i, j \in \{0, 1, \ldots, 2^{k-1}\}$ such that $j - i = 2^\ell$ for some $\ell \in \mathbb{Z}_{\geqslant 0}$, add a path of length 2 between v_i and v_j. More precisely, for any such i and j, we add a new vertex $w_{i,j}$ as well as edges $\{v_i, w_{ij}\}$ and $\{w_{ij}, v_j\}$. Additionally, we define cuts S_ℓ for $\ell \in [2^k]$ by

$$S_\ell \coloneqq \{v_i \mid 2i < \ell\} \cup \{w_{i,j} \mid i + j < \ell\} \quad .$$

Note that for $\ell_1 < \ell_2$, we have $S_{\ell_1} \subseteq S_{\ell_2}$, and thus the family $\mathcal{S}_k \coloneqq \{S_1, S_2, \ldots, S_{2^k}\}$ is a chain. An illustration of this construction for $k = 3$ is given in Fig. 4.5.

To complete the instances of the MCCST problem and to ensure feasibility, we can define $a_\ell = 0$ and $b_\ell = k + 1$ for all $\ell \in [2^k]$. Note that this is equivalent to not putting any constraints on the sizes of the given cuts. In fact our arguments are independent of the precise degree bounds (given feasibility).

We first observe that the corresponding relaxation Q has solutions with small weight on all cuts in $\mathcal{S}_k$. To this end, define $y \in \mathbb{R}^E_{\geqslant 0}$ by

$$y(e) = \begin{cases} 1 & \text{if } e = \{v_{i-1}, v_i\} \text{ for some } i \in [2^{k-1}], \\ {}^1/_2 & \text{else.} \end{cases}$$

Note that y is indeed a point in P_{ST}. This can be seen by writing $y = {}^{(x_1 + x_2)}/_2$, where $x_1, x_2 \in \mathbb{R}^E_{\geqslant 0}$ are incidence vectors of spanning trees given by

$$x_1(e) = \begin{cases} 1 & \text{if } e = \{v_{i-1}, v_i\} \text{ for some } i, \\ 1 & \text{if } e = \{v_i, w_{i,j}\} \text{ for some } i, j, \\ 0 & \text{if } e = \{w_{i,j}, v_j\} \text{ for some } i, j, \end{cases} \quad \text{and} \quad x_2(e) = \begin{cases} 1 & \text{if } e = \{v_{i-1}, v_i\} \text{ for some } i, \\ 0 & \text{if } e = \{v_i, w_{i,j}\} \text{ for some } i, j, \\ 1 & \text{if } e = \{w_{i,j}, v_j\} \text{ for some } i, j. \end{cases}$$

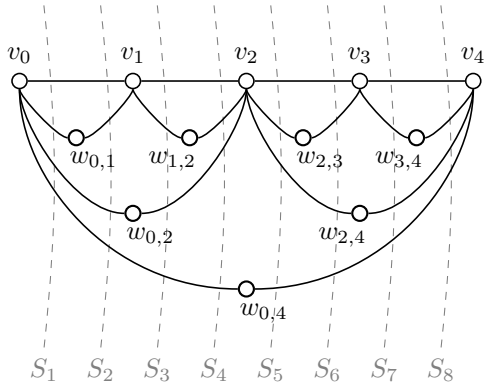


Figure 4.5: The graph H_k and the family $\mathcal{S}_k = \{S_1, \ldots, S_2\}$ of cuts for $k = 3$.

Moreover, we observe that for every $i \in [2^k]$, we have

$$y(\delta(S_i)) = 1 + \frac{k}{2} \ . \tag{4.5}$$

The above implies that indeed, $y \in Q$. Now, consider a spanning tree T of H_k. We claim that for every $k \in \mathbb{Z}_{>0}$, there exists $i \in [2^k]$ such that

$$|T \cap \delta(S_i)| \geqslant k \ . \tag{4.6}$$

Once we prove this, we can combine (4.5) and (4.6) to obtain that there exists $i \in [2^k]$ such that

$$|T \cap \delta(S_i)| \geqslant \frac{k}{1 + k/2} \cdot y(\delta(S_i)) = \left(2 - \frac{2}{k+2}\right) \cdot y(\delta(S_i)) \ .$$

By choosing k large enough such that $\varepsilon \geqslant 2/(k+2)$, we thus indeed obtain an instance satisfying the properties listed in Theorem 4.31. It remains to prove the claim. To this end, we show the following stronger lemma.

Lemma 4.32. *Let F be a subset of the edges of H_k such that any vertex $w_{i,j}$ of H_k is incident to at least one edge of F. Then there exists $i \in [2^k]$ such that $|F \cap \delta(S_i)| \geqslant k$.*

Proof of Lemma 4.32. We proceed by induction on k. The statement of the base case $k = 1$ is directly implied by the assumption on F. Indeed, at least one of the two edges $\{v_0, w_{0,1}\}$ and $\{w_{0,1}, v_1\}$ is in F, and correspondingly, at least one of $|F \cap \delta(S_1)| \geqslant 1$ or $|F \cap \delta(S_2)| \geqslant 1$ holds.

For the inductive step, let $k \geqslant 2$ and consider the graph $H_k = (V, E)$. By the assumption on F, at least one of the edges $\{v_0, w_{0,2^{k-1}}\}$ and $\{w_{0,2^{k-1}}, v_{2^{k-1}}\}$ is in F. By symmetry, we can assume without loss of generality that $\{v_0, w_{0,2^{k-1}}\} \in F$. Observe that the subgraph of H_k induced by the vertex set $V_0 := \{v_i \mid i \leqslant 2^{k-2}\} \cup \{w_{i,j} \mid i, j \leqslant 2^{k-2}\}$ is isomorphic to H_{k-1}, and $F \cap E[V_0]$ has an edge incident to every vertex $w_{i,j}$ of this copy of H_{k-1}. Hence, by induction, there exists $i \in [2^{k-1}]$ such that $|(F \cap E[V_0]) \cap \delta(S_i)| \geqslant k - 1$. As additionally, $\{v_0, w_{0,2^{k-1}}\} \in \delta(S_i)$, this implies $|F \cap \delta(S_i)| \geqslant k$. □

Finally, observe that by connectivity, every spanning tree T of H_k contains at least one edge incident to $w_{i,j}$, for all vertices $w_{i,j}$ of H_k. Consequently, Lemma 4.32 does indeed imply existence of $i \in [2^k]$ such that $|T \cap \delta(S_i)| \geqslant k$. This completes the proof of Theorem 4.31. □

Appendix 4.B Analyzing the DP by backtracing OPT fails in the general case

The aim of this section is to extend an example from Section 4.5 which showed that the analysis of our dynamic programming approach cannot be done in the classical way, i.e., by backtracing an optimal solution. While the issues in the example constructed in Section 4.5 can be fixed by breaking ties in the right way, we now present a slightly more involved instance (see Fig. 4.6) where the naive approach faces problems that cannot be avoided easily.

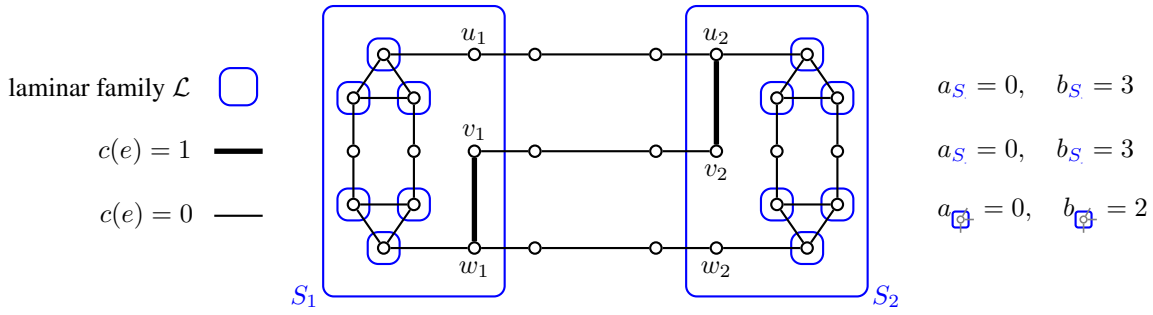
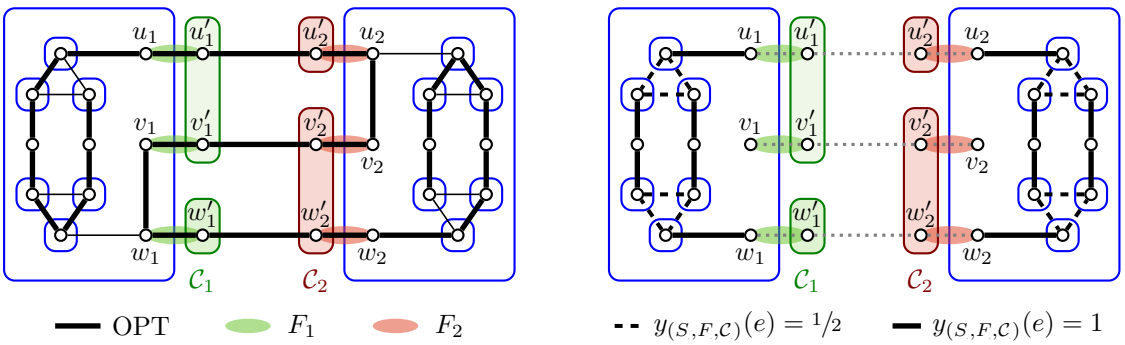


Figure 4.6: An instance of the MLCST problem: The laminar family $\mathcal{L}$ is given by the blue sets, with lower and upper bounds indicated on the right. Edge costs $c\colon E \to \mathbb{R}_{\geqslant 0}$ are 0 on all edges except for $c((v_1, w_1)) = c((u_2, v_2)) = 1$.

The problem instance in Fig. 4.6 is very similar to the instance discussed in Section 4.5 (Fig. 4.3). While the latter had edges (u_i, w_i) for $i \in \{1, 2\}$, the vertices u_i and w_i are connected by an auxiliary graph in the new instance. This auxiliary graph has the following two crucial properties:

(i) The auxiliary graph does not contain a spanning tree that satisfies the laminar constraints.
(ii) The spanning tree polytope of the auxiliary graph contains a point that satisfies the laminar constraints.

In other words, the two properties state that it is possible to "fractionally connect" u_i and w_i in the auxiliary graph, while an integral solution cannot connect u_i and w_i through the auxiliary graph. In particular, this implies that any feasible integral solution will use both edges (v_1, w_1) and (u_2, v_2), and thus have cost at least 2. One such integral solution is given in Fig. 4.7a. Observe that no matter how we choose an integral solution, the connectivity patterns induced on the sets S_1 and S_2 will always be $(S_1, F_1, \mathcal{C}_1)$ and $(S_2, F_2, \mathcal{C}_2)$, respectively, as indicated in Fig. 4.7a.



(a) An optimal integral solution with induced connectivity triples $(S_1, F_1, \mathcal{C}_1)$ and $(S_2, F_2, \mathcal{C}_2)$ on S_1 and S_2.

(b) Fractional solutions $y_{(S,F,\mathcal{C})} \in P_{\mathrm{ST}}(S_i, F_i, \mathcal{C}_i)$ that can not be completed to a global solution.

Figure 4.7: Optimal integral and partial fractional solutions connect vertices differently inside S_1 and S_2.

To analyze our DP approach by classical backtracing of an optimal solution, our goal would be to find partial solutions $y_{(S_i,F_i,\mathcal{C}_i)} \in P_{\mathrm{ST}}(S_i, F_i, \mathcal{C}_i)$ for $i \in \{1, 2\}$, and show that a common extension of these solutions has smaller value than the actual optimal solution we started with. In our example, however, property (ii) of the auxiliary graph allows for partial fractional solutions $y_{(S_i,F_i,\mathcal{C}_i)}$ that differ substantially from integral solutions in terms of connectivity. More precisely, the two fractional solutions $y_{(S_1,F_1,\mathcal{C}_1)}$ and $y_{(S_2,F_2,\mathcal{C}_2)}$ given in Fig. 4.7b are both of cost 0 (and hence optimal), but there does not exist a common extension that is feasible for the natural linear relaxation of our problem instance at all.

There is one last caveat that has to be addressed: Our dynamic program is designed to construct partial solutions inside all *small* cuts of the laminar family for any choice of edges in the small cuts and corresponding connectivity patterns, and it always extends previously found solutions. Above, the threshold τ for deciding whether a cut is small was implicitly assumed to be at least 3 so that both S_1 and S_2 are small cuts. In the particular example, this implies that all the other cuts in $\mathcal{L}$ (which are precisely the singleton cuts) would be small cuts, as well, forcing our dynamic program to first construct partial solutions in these small cuts and only then extend to S_1 and S_2. This would inevitably lead to integral partial solutions, hence we do need a setting where the singleton cuts are *large* cuts.

To achieve this, we introduce dummy edges that increase the number of edges in the singleton cuts of $\mathcal{L}$. More precisely, for any given threshold τ and every small singleton cut $\{x\} \in \mathcal{L}$, we can modify the problem instance as follows to turn $\{x\}$ into a large cut: Introduce new vertices $x_1, \dots, x_\tau$ and edges $\{x, x_i\}$ for $1 \leqslant i \leqslant \tau$, and increase the bounds $a_{\{x\}}$ and $b_{\{x\}}$ by τ. Feasible solutions of the old and the new instance are in one-to-one correspondence and can be transformed into one another by adding or removing all the edges $\{x, x_i\}$, which are obviously part of any feasible solution of the new instance.

To conclude, by introducing dummy edges in the graph given in Fig. 4.6 as described above, we obtain an instance where an analysis of our DP approach by backtracing an optimal solution fails inevitably, thus again emphasizing the importance of our novel approach.

Bibliography

[AEGOVW16] S. Artmann, F. Eisenbrand, C. Glanzer, T. Oertel, S. Vempala, and R. Weismantel. "A note on non-degenerate integer programs with small sub-determinants". In: *Operations Research Letters* 44.5 (2016), pp. 635–639.

[AF21] M. Aprile and S. Fiorini. "Regular Matroids Have Polynomial Extension Complexity". In: *Mathematics of Operations Research* (2021). To appear.

[AGMOS10] A. Asadpour, M. X. Goemans, A. Madry, S. Oveis Gharan, and A. Saberi. "An $O(\log n/\log\log n)$-approximation Algorithm for the Asymmetric Traveling Salesman Problem". In: *Proceedings of the 21st ACM-SIAM Symposium on Discrete Algorithms (SODA '10)*. Austin, 2010, pp. 379–389.

[AKS15] H.-C. An, R. Kleinberg, and D. B. Shmoys. "Improving Christofides' Algorithm for the s-t Path TSP". In: *Journal of the ACM* 62.5 (2015), 34:1–34:28.

[Art20] S. Artmann. "Optimization of bimodular integer programs and feasibility for three-modular base block IPs". PhD thesis. ETH Zurich, 2020.

[AWZ17] S. Artmann, R. Weismantel, and R. Zenklusen. "A Strongly Polynomial Algorithm for Bimodular Integer Linear Programming". In: *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC '17)*. Montreal, 2017, pp. 1206–1219.

[BC87] F. Barahona and M. Conforti. "A construction for binary matroids". In: *Discrete Mathematics* 66.3 (1987), pp. 213–218.

[BDEHN14] N. Bonifas, M. Di Summa, F. Eisenbrand, N. Haehnle, and M. Niemeier. "On Sub-determinants and the Diameter of Polyhedra". In: *Discrete Computational Geometry* 52.1 (2014), pp. 14. 102–115.

[Ben95] A. A. Benczúr. "A Representation of Cuts Within 6/5 Times the Edge Connectivity with Applications". In: *Proceedings of the 36th Annual IEEE Symposium on Foundations of Computer Science (FOCS '95)*. Milwaukee, 1995, pp. 92–102.

[BFMR14] A. A. Bock, Y. Faenza, C. Moldenhauer, and A. J. Ruiz-Vargas. "Solving the Stable Set Problem in Terms of the Odd Cycle Packing Number". In: *Proceedings of the 34th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS '14)*. New Delhi, 2014, pp. 187–198.

[BHKP08] A. Bhalgat, R. Hariharan, T. Kavitha, and D. Panigrahi. "Fast Edge Splitting and Edmonds' Arborescence Construction for Unweighted Graphs". In: *Proceedings of the 19th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '08)*. San Francisco, 2008, pp. 455–464.

[BK98] A. A. Benczúr and D. R. Karger. “Augmenting Undirected Edge Connectivity in $\mathcal{O}(n^2)$ Time”. In: *Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA ’98)*. San Francisco, 1998, pp. 500–509.

[BKKNP13] N. Bansal, R. Khandekar, J. Könemann, V. Nagarajan, and B. Peis. “On generalizations of network design problems with degree bounds”. In: *Mathematical Programming, Series A* 141 (2013), pp. 479–506.

[BKN09] N. Bansal, R. Khandekar, and V. Nagarajan. “Additive guarantees for degree-bounded directed network design”. In: *SIAM Journal on Computing* 39.4 (2009), pp. 1413–1431.

[BT97] D. Bertsimas and C. Teo. “The parsimonious property of cut covering problems and its applications”. In: *Operations Research Letters* 21.3 (1997), pp. 123–132.

[CFG15] J. Cheriyan, Z. Friggstad, and Z. Gao. “Approximating Minimum-Cost Connected T-Joins”. In: *Algorithmica* 72 (2015), pp. 126–147.

[CFHJW20] M. Conforti, S. Fiorini, T. Huynh, G. Joret, and S. Weltge. “The stable set problem in graphs with bounded genus and bounded odd cycle packing number”. In: *Proceedings of the 31st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA ’20)*. Salt Lake City, 2020, pp. 2896–2915.

[CFHW21] M. Conforti, S. Fiorini, T. Huynh, and S. Weltge. “Extended formulations for stable set polytopes of graphs without two disjoint odd cycles”. In: *Mathematical Programming* (2021). To appear.

[CGM92] P. M. Camerini, G. Galbiati, and F. Maffioli. “Random Pseudo-Polynomial Algorithms for Exact Matroid Problems”. In: *Journal of Algorithms* 13 (1992), pp. 258–273.

[Chr76] N. Christofides. *Worst-case analysis of a new heuristic for the Travelling Salesman Problem*. Technical Report 388. Graduate School of Industrial Administration, Carnegie Mellon University, 1976.

[Cla] Clay Mathematics Institute. *Millenium Problems: The P vs NP Problem*. https://www.claymath.org/millennium-problems/p-vs-np-problem, accessed September 07, 2021.

[CRRT09a] K. Chaudhuri, S. Rao, S. Riesenfeld, and K. Talwar. “A push-relabel approximation algorithm for approximating the minimum-degree MST problem and its generalization to matroids”. In: *Theoretical Computer Science* 410 (44 2009), pp. 4489–4503.

[CRRT09b] K. Chaudhuri, S. Rao, S. Riesenfeld, and K. Talwar. “What would Edmonds Do? Augmenting paths and witnesses for degree-bounded MSTs”. In: *Algorithmica* 55 (1 2009), pp. 157–189.

[CVZ09] C. Chekuri, J. Vondrák, and R. Zenklusen. *Dependent Randomized Rounding for Matroid Polytopes and Applications*. 2009. arXiv: 0909.4348 [cs.DS].

[CVZ10] C. Chekuri, J. Vondrák, and R. Zenklusen. “Dependent Randomized Rounding via Exchange Properties of Combinatorial Structures”. In: *51st Annual IEEE Symposium on Foundations of Computer Science (FOCS ’10)*. Las Vegas, 2010, pp. 575–584.

[CXY18] K. Chandrasekaran, C. Xu, and X. Yu. “Hypergraph k-Cut in Randomized Polynomial Time”. In: *Proceedings of the 29th Annual Annual ACM-SIAM Symposium on Discrete Algorithms (SODA ’18)*. New Orleans, 2018, pp. 1426–1438.

[DEFM15] M. Di Summa, F. Eisenbrand, Y. Faenza, and C. Moldenhauer. “On Largest Volume Simplices and Sub-determinants”. In: *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA ’15)*. San Diego, 2015, pp. 315–323.

[DK14] M. Dinitz and G. Kortsarz. “Matroid Secretary for Regular and Decomposable Matroids”. In: *SIAM Journal on Computing* 43.5 (2014), pp. 1807–1830.

[Edm71] J. Edmonds. “Matroids and the Greedy Algorithm”. In: *Mathematical Programming* 1.1 (1971), pp. 127–136.

[EJ73] J. Edmonds and E. L. Johnson. “Matching, Euler tours and the Chinese postman”. In: *Mathematical Programming* 5.1 (1973), pp. 88–124.

[EV17] F. Eisenbrand and S. Vempala. “Geometric random edge”. In: *Mathematical Programming* 164.1 (2017), pp. 325–339.

[FJWY21] S. Fiorini, G. Joret, S. Weltge, and Y. Yuditsky. *Integer programs with bounded subdeterminants and two nonzeros per row*. 2021. arXiv: 2106.05947 [math.CO].

[Fra92] A. Frank. “Augmenting graphs to meet edge-connectivity requirements”. In: *SIAM Journal on Discrete Mathematics* 5.1 (1992), pp. 25–53.

[Gab94] H. N. Gabow. “Efficient Splitting off Algorithms for Graphs”. In: *Proceedings of the 26th Annual ACM SIGACT Symposium on Theory of Computing (STOC ’94)*. 1994, pp. 696–705.

[GH61] R. E. Gomory and T. C. Hu. “Multi-Terminal Network Flows”. In: *Journal of the Society for Industrial and Applied Mathematics* 9.4 (1961), pp. 551–570.

[GKS95] J. W. Grossman, D. M. Kulkarni, and I. E. Schochetman. “On the minors of an incidence matrix and its Smith normal form”. In: *Linear Algebra and its Applications* 218 (1995), pp. 213–224.

[GLS84] M. Grötschel, L. Lovász, and A. Schrijver. “Corrigendum to our paper ‘The ellipsoid method and its consequences in combinatorial optimization’”. In: *Combinatorica* 4.4 (1984), pp. 291–295.

[GLS93] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization*. 2nd. Vol. 2. Algorithms and combinatorics. Springer, 1993.

[God04] L. A. Goddyn. *Some Open Problems I Like*. http://people.math.sfu.ca/~goddyn/Problems/problems.html. 2004.

[Goe06] M. X. Goemans. “Minimum bounded degree spanning trees”. In: *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS ’06)*. Berkeley, 2006, pp. 273–282.

[GR95] M. X. Goemans and V. S. Ramakrishnan. “Minimizing Submodular Functions over Families of Sets”. In: *Combinatorica* 15.4 (1995), pp. 499–513.

[GSW21] C. Glanzer, I. Stallknecht, and R. Weismantel. “On the Recognition of $\{a, b, c\}$-Modular Matrices”. In: *Proceedings of the 22nd Conference on Integer Programming and Combinatorial Optimization (IPCO ’21)*. Atlanta, 2021, pp. 238–251.

[GV16] C. Gottschalk and J. Vygen. “Better s-t-Tours by Gao Trees”. In: *Proceedings of 18th Conference on Integer Programming and Combinatorial Optimization (IPCO)*. 2016, pp. 126–137.

[Kar93] D. R. Karger. “Global Min-Cuts in $\mathcal{RNC}$, and Other Ramifications of a Simple Min-Cut Algorithm”. In: *Proceedings of the 4th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA ’93)*. Austin, 1993, pp. 21–30.

[Kho06] S. Khot. “Ruling Out PTAS for Graph Min-Bisection, Dense k-Subgraph, and Bipartite Clique”. In: *SIAM Journal on Computing* 36.4 (2006), pp. 1025–1071.

[KKG21] A. R. Karlin, N. Klein, and S. O. Gharan. “A (slightly) improved approximation algorithm for metric TSP”. In: *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing (STOC ’21)*. Rome, 2021, pp. 32–45.

[KKR12] K. Kawarabayashi, Y. Kobayashi, and B. Reed. “The disjoint paths problem in quadratic time”. In: *Journal of Combinatorial Theory, Series B* 102.2 (2012), pp. 424–435.

[KR00] J. Könemann and R. Ravi. “A matter of degree: Improved approximation algorithms for degree-bounded minimum spanning trees”. In: *Proceedings of the 32nd Annual ACM SIGACT Symposium on Theory of Computing (STOC ’00)*. 2000, pp. 537–546.

[KR03] J. Könemann and R. Ravi. “Primal-dual meets local search: Approximating MSTs with nonuniform degree bounds.” In: *Proceedings of the 35th Annual ACM SIGACT Symposium on Theory of Computing (STOC ’03)*. 2003, pp. 389–395.

[KS96] D. R. Karger and C. Stein. “A new approach to the minimum cut problem”. In: *Journal of the ACM* 43.4 (1996), pp. 601–640.

[KV18] B. Korte and J. Vygen. *Combinatorial Optimization, Theory and Algorithms*. 6th. Springer, 2018.

[Len83] H. W. Lenstra. “Integer Programming with a Fixed Number of Variables”. In: *Mathematics of Operations Research* 8.4 (1983), pp. 538–548.

[Lov76] L. Lovász. “On some connectivity properties of Eulerian graphs”. In: *Acta Mathematica Academiae Scientiarum Hungarica* 28.1 (1976), pp. 129–138.

[Lov79] L. Lovász. *Combinatorial Problems and Exercises*. Amsterdam: North-Holland, 1979.

[LPSX20] J. Lee, J. Paat, I. Stallknecht, and L. Xu. “Improving Proximity Bounds Using Sparsity”. In: *Proceedings of the 6th International Symposium on Combinatorial Optimization (ISCO ’20)*. Montreal, 2020, pp. 115–127.

[LPSX21] J. Lee, J. Paat, I. Stallknecht, and L. Xu. *Polynomial upper bounds on the number of differing columns of Δ-modular integer programs*. 2021. arXiv: 2105.08160 [math.OC].

[LS16] A. Linhares and C. Swamy. “Approximating min-cost chain-constrained spanning trees: A reduction from weighted to unweighted problems”. In: *Proceedings of the 18th Conference on Integer Programming and Combinatorial Optimization (IPCO ’16)*. Liège, 2016.

[LY13] L. Lau and C. Yung. “Efficient Edge Splitting-Off Algorithms Maintaining All-Pairs Edge-Connectivities”. In: *SIAM Journal on Computing* 42.3 (2013), pp. 1185–1200.

[Mad78] W. Mader. “A Reduction Method for Edge-Connectivity in Graphs”. In: *Annals of Discrete Mathematics* 3 (1978), pp. 145–164.

[Mar91] R. Martin. “Using separation algorithms to generate mixed integer model reformulations”. In: *Operations Research Letters* 10.3 (1991), pp. 119–128.

[Meg79] N. Megiddo. “Combinatorial Optimization with Rational Objective Functions”. In: *Mathematics of Operations Research* 4.4 (1979), pp. 414–424.

[Meg83] N. Megiddo. “Applying Parallel Computation Algorithms in the Design of Serial Algorithms”. In: *Journal of the ACM* 30.4 (1983), pp. 852–865.

[NI00] S. Nagamochi H. and Nakamura and T. Ibaraki. “A Simplified $\mathcal{O}(nm)$ Time Edge-Splitting Algorithm in Undirected Graphs”. In: *Algorithmica* 26.1 (2000), pp. 50–67.

[NI94] K. Nagamochi H. and Nishimura and T. Ibaraki. “Computing all small cuts in undirected networks”. In: *International Symposium on Algorithms and Computation (ISAAC)*. 1994, pp. 190–198.

[NI96] H. Nagamochi and T. Ibaraki. “Deterministic $\mathcal{O}(nm)$ Time Edge-splitting in Undirected Graphs”. In: *Proceedings of the 28th Annual ACM SIGACT Symposium on Theory of Computing (STOC ’96)*. Philadelphia, 1996, pp. 64–73.

[Nik15] A. Nikolov. “Randomized Rounding for the Largest Simplex Problem”. In: *Proceedings of the 47th Annual ACM SIGACT Symposium on Theory of Computing (STOC ’15)*. Portland, 2015, pp. 861–870.

[NNI97] H. Nagamochi, K. Nishimura, and T. Ibaraki. “Computing all small cuts in an undirected network”. In: *SIAM Journal in Discrete Mathematics* 10.3 (1997), pp. 469–481.

[NSZ19] M. Nägele, B. Sudakov, and R. Zenklusen. “Submodular Minimization Under Congruency Constraints”. In: *Combinatorica* 39.6 (2019), pp. 1351–1386.

[NSZ21] M. Nägele, R. Santiago, and R. Zenklusen. *Congruency-Constrained TU Problems Beyond the Bimodular Case*. 2021. arXiv: 2109.03148 [math.OC].

[NZ19a] M. Nägele and R. Zenklusen. “A new contraction technique with applications to congruency-constrained cuts”. In: *Proceedings of the 20th Conference on Integer Programming and Combinatorial Optimization (IPCO ’19)*. Ann Arbor, 2019, pp. 327–340.

[NZ19b] M. Nägele and R. Zenklusen. “A new dynamic programming approach for spanning trees with chain constraints and beyond”. In: *Proceedings of the 30th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA ’19)*. San Diego, 2019, pp. 1550–1569.

[NZ20] M. Nägele and R. Zenklusen. “A new contraction technique with applications to congruency-constrained cuts”. In: *Mathematical Programming* 183 (2020), pp. 455–481.

[OZ18] N. Olver and R. Zenklusen. “Chain-Constrained Spanning Trees”. In: *Mathematical Programming* 167.2 (2018), pp. 293–314.

[PR82] M. W. Padberg and M. R. Rao. “Odd Minimum Cut-Sets and b-Matchings”. In: *Mathematics of Operations Research* 7.1 (1982), pp. 67–80.

[PSW21] J. Paat, M. Schlöter, and R. Weismantel. “The integrality number of an integer program”. In: *Mathematical Programming* (2021). To appear.

[Sch03] A. Schrijver. *Combinatorial Optimization, Polyhedra and Efficiency*. Springer, 2003.

[Sch98] A. Schrijver. *Theory of linear and integer programming*. John Wiley & Sons, 1998.

[Seb13] A. Sebő. “Eight-Fifth Approximation for the path TSP”. In: *Proceedings of 16th Conference on Integer Programming and Combinatorial Optimization (IPCO)*. 2013, pp. 263–373.

[Sey80] P. D. Seymour. “Decomposition of regular matroids”. In: *Journal of Combinatorial Theory, Series B* 28.3 (1980), pp. 305–359.

[SL07] M. Singh and L. C. Lau. “Approximating minimum bounded degree spanning trees to within one of optimal”. In: *Proceedings of the 39th Annual ACM SIGACT Symposium on Theory of Computing (STOC ’07)*. ACM, 2007, pp. 661–670.

[STV18] O. Svensson, J. Tarnawski, and L. A. Végh. “A Constant-Factor Approximation Algorithm for the Asymmetric Traveling Salesman Problem”. In: *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC ’18)*. 2018, pp. 204–213.

[SZ16] A. Sebő and A. van Zuylen. “The Salesman’s Improved Paths: A $3/2 + 1/34$ Approximation”. In: *Proceedings of 57th Annual IEEE Symposium on Foundations of Computer Science (FOCS ’16)*. New Brunswick, 2016, pp. 118–127.

[Tar86] É. Tardos. “A Strongly Polynomial Algorithm to Solve Combinatorial Linear Programs”. In: *Operations Research* 34.2 (1986), pp. 250–256.

[TV18] V. Traub and J. Vygen. “Approaching $\frac{3}{2}$ for the s-t path TSP”. In: *Proceedings of the 29th ACM-SIAM Symposium on Discrete Algorithms (SODA)*. 2018, pp. 1854–1864.

[TV20] V. Traub and J. Vygen. “An improved approximation algorithm for ATSP”. In: *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, (STOC ’20)*. Chicago, 2020, pp. 1–13.

[TVZ20] V. Traub, J. Vygen, and R. Zenklusen. “Reducing path TSP to TSP”. In: *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, (STOC ’20)*. Chicago, 2020, pp. 14–27.

[VC09] S. I. Veselov and A. J. Chirkov. “Integer program with bimodular matrix”. In: *Discrete Optimization* 6.2 (2009), pp. 220–222.

[Vyg16] J. Vygen. “Reassembling Trees for the Traveling Salesman”. In: *SIAM Journal on Discrete Mathematics* 30.2 (2016), pp. 875–894.

[Wol80] L. A. Wolsey. “Heuristic analysis, linear programming and branch and bound”. In: *Mathematical Programming Studies* 13 (1980), pp. 121–134.

[Zen12] R. Zenklusen. “Matroidal degree-bounded minimum spanning trees”. In: *Proceedings of the 23rd Annual ACM-SIAM Symposium on Discrete Algorithms (SODA ’12)*. 2012, pp. 1512–1521.

[Zen19] R. Zenklusen. “A 1.5-Approximation for Path TSP”. In: *Proceedings of the 30th ACM-SIAM Symposium on Discrete Algorithms (SODA ’19)*. San Diego, 2019, pp. 1539–1549.