EPFL

Thèse n° 7727

New Results in Integer and Lattice Programming

Présentée le 19 juin 2020

à la Faculté des sciences de base
Chaire d'optimisation discrète
Programme doctoral en mathématiques

pour l'obtention du grade de Docteur ès Sciences

par

Christoph HUNKENSCHRÖDER

Acceptée sur proposition du jury

Prof. K. Hess Bellwald, présidente du jury
Prof. F. Eisenbrand, directeur de thèse
Prof. S. Weltge, rapporteur
Prof. R. Zenklusen, rapporteur
Prof. M. Kapralov, rapporteur

École
polytechnique
fédérale
de Lausanne

2020

Acknowledgements

Looking back on 13 years of school, 6 years of studies in Bonn, and 4 years of doing my PhD in Lausanne, too many awesome people accompanied me during these times to name them all. In particular in the last four years I met a lot of people who left dear memories and contributed in many different ways to this thesis and my work in academia.

Let me start with my advisor Friedrich Eisenbrand. Thank you for these four years, for your support, stimulating ideas and asking the right questions, trusting me and giving me the freedom to follow my interests and visit conferences and workshops all over the world. For providing the necessary financial support, I want to thank the Swiss National Science Foundation. I also want to thank Kim-Manuel Klein, Martin Koutecký, Asaf Levin, Shmuel Onn, Gina Reuland and Matthias Schymura for fruitful discussions and inspiring collaboration that led to this thesis.

It was quite an experience to finish this thesis and defend it in front of a friendly and genuinely interested jury, comprising my advisor, Kathryn Hess Bellwald, Michael Kapralov, Stefan Weltge and Rico Zenklusen. Thank you for the time you spent on reviewing this work and the encouraging feedback after my defense. I am indebted to Adrian Vetta, Martin Koutecký, Asaf Levin, Shmuel Onn and José Verschae for their hospitality and good discussions.

A big *thank you!* to the whole Disopt group and my fellow PhD students and postdocs I met here. First I'd like to thank Alfonso, Manuel, Igor, Yuri, Linda and Natalie who welcomed me into the Disopt group. Moreover, thanks to Jana, Jonas, Martina and Moritz for reviewing parts of this thesis, good (also non-mathematical) discussions, and so many other things. Special thanks go to Georg and Matthias. I learned a lot from you and value the time and effort you spent on answering my questions and sharing your experience in academia. I also do not want to miss out on the chance to thank Jocelyne for her good nature, her help in finding an apartment, organizing skiing trips, and her expertise in how things work at EPFL. A lot more people made these four years a good time with hikes, board games, and pancake parties, to name a few things.

I'd like to thank my bachelor's thesis supervisor Jens Vygen. Building up on my bachelor's thesis with him, I was able to visit Adrian Vetta, and I consider these two people to be the ones establishing my first contact with research. In my master studies, Nicolai Hähnle introduced me to lattices and the whole area I'm still working in and enjoy. He has my gratitude for this and his supervision of my master's thesis.

Acknowledgements

There are the friends I made in Bonn, sharing memories of countless rounds of Doppelkopf or Skat, sneak previews, DSA, trips to Innsbruck and many other fun activities. I had a great time in Bonn, and I'd like to thank everyone who contributed to this.
My oldest friends still provide me with a great time whenever I'm back in the German Münsterland and help me restoring my batteries when work piled up. Be it playing cards, having New Year's breakfast, baking Christmas cookies, enjoying a glass of wine on St. Lambert's church atop the roofs of Münster, or simply cooking and enjoying an evening on the couch. Thanks for pulling my mind out of math every once in a while.
When I tell strangers that I study mathematics, most people are awed, and seem to have the opinion that math is nothing for regular folks, and only the smartest people can do it. My former teacher Josef Muth has a talent in showing people that this prevailing image of mathematics is wrong, and showed a lot of students that basically everyone is capable of math (admittedly, to a certain extent). I had a lot of fun in his classes, and remember various class mates who agree. Thank you for all the encouragement with which you met so many students, and for making math approachable for everyone.
Everything is math, but math isn't everything. I want to thank Chuck Ragan, Isaac Childres and Jim Butcher, representing their respective profession, for giving me a lot to do on my days off.
Most important, I want to thank my family. I grew up in a loving environment, with parents that always had my back and believed in me. Thanks to my brothers, who don't always share my opinion, but on whom I can rely without any doubt. Also my grandparents, godparents, aunts, uncles and cousins showed me how nice it is to be part of a family in a broader sense, and my gratitude extends to them. A big chunk of who I am today is due to you.

Sassenberg, April 14, 2020 Christoph Hunkenschröder

Abstract

An integer program (IP) is a problem of the form $\min\{f(x) : Ax = b,\ l \leq x \leq u,\ x \in \mathbb{Z}^n\}$, where $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, $l, u \in \mathbb{Z}^n$, and $f : \mathbb{Z}^n \to \mathbb{Z}$ is a separable convex objective function. The problem of finding an optimal solution for an integer program is known as integer programming. Integer programming is NP-hard in general, though several algorithms exist: Lenstra provided an algorithm that is polynomial if the dimension n is fixed. For variable dimension, the best known algorithm depends linearly on n, and exponentially on the number of equalities as well as the largest absolute value of an entry in the matrix A.

The first part of this thesis considers integer programming for variable dimensions and sparse matrices. We measure the sparsity of a matrix by the tree-depth of the dual graph of A. A typical example for these integer programs are N-fold IPs, used for scheduling and social choice problems. We obtain the currently fastest fixed-parameter tractable algorithm with parameters tree-depth and the largest absolute value of the entries in A. The running time we achieve is near-linear in the dimension. With a slightly worse running time, we are able to show that N-fold integer programs of constant block size can be solved in strongly polynomial time. Assuming the exponential time hypothesis, we complement these results with a lower bound on the parameter dependency that almost matches the parameter dependency of the running time. As a consequence, we provide the currently strongest lower bound for N-fold integer programs.

Another problem closely related to integer programming is the closest vector problem. A lattice is a discrete additive subgroup of $\mathbb{R}^n$. The closest vector problem (CVP) asks for a lattice point closest to a given target vector. An important tool for solving the closest vector problem is the Voronoi cell $\mathcal{V}$ of a lattice $\Lambda \subseteq \mathbb{R}^n$, which is the set of all points for which 0 is a closest lattice point. It is a polytope whose facets are induced by a set of lattice vectors, the Voronoi relevant vectors. A generic lattice has exponentially many Voronoi relevant vectors, leading to exponential space for certain CVP algorithms.

In the second part of this thesis, we introduce the notion of a c-compact lattice basis $B \in \mathbb{R}^{n \times n}$ that facilitates to represent the Voronoi relevant vectors with coefficients bounded by c. Such a basis allows to reduce the space requirement of Micciancio's & Voulgaris' algorithm for the closest vector problem from exponential to polynomial, while the running time becomes exponential in c. We show that for every lattice an n^2-compact basis exists, but there are lattices for which we cannot choose $c \in o(n)$. If the Voronoi cell is a zonotope, we can choose $c = 1$, providing a single-exponential time and polynomial

space algorithm for CVP, assuming a 1-compact basis is known.
Deciding whether a given lattice has a certain structure that helps to solve the closest vector problem more efficiently is a reappearing and non-trivial problem. The third part of this thesis is concerned with the specific structure of having an orthonormal basis. We show that this problem belongs to NP ∩ co-NP. Moreover, it can be reduced to solving a single closest vector problem. We also show that if a separation oracle for the Voronoi cell is provided, CVP is solvable in polynomial time.

keywords: integer programming, n-fold IP, tree-depth, Graver basis, fixed-parameter tractable, exponential time hypothesis, lattice, Voronoi cell, closest vector problem, co-NP, characteristic vector

Zusammenfassung

Ein ganzzahliges Programm (IP) ist ein Problem der Form $\min\{f(x) : Ax = b,\ l \leq x \leq u,\ x \in \mathbb{Z}^n\}$, wobei $A \in \mathbb{Z}^{m \times n}$, $B \in \mathbb{Z}^m$, $l, u\ in\ Z^n$ und $f : \mathbb{Z}^n \to \mathbb{Z}$ eine separabel-konvexe Zielfunktion ist. Ganzzahlige Programmierung, also das Finden einer optimalen Lösung für ein IP, ist im Allgemeinen NP-schwer, obwohl verschiedene Algorithmen existieren: Lenstra stellte einen polynomiellen Algorithmus zur Verfügung, wenn die Dimension n festgelegt ist. Bei variabler Dimension hängt der derweil schnellste Algorithmus linear von der Dimension und exponentiell von der Anzahl der Gleichungen sowie dem größten Absolutbetrag eines Eintrags in der Matrix A ab.

Im ersten Teil dieser Arbeit betrachten wir ganzzahlige Programmierung für variable Dimension und spärliche Matrizen. Wir messen diese Spärlichkeit an der Baumtiefe des dualen Graphen von A. Ein typisches Beispiel für derartige ganzzahlige Programme sind N-fache IPs, die unter Anderem für Planungsprobleme und in der Sozialwahltheorie Anwendung finden. Wir erhalten den derzeit schnellsten parametrisierten Algorithmus, der durch die Baumtiefe und den größten absoluten Wert der Einträge in A parametrisiert ist. Die erreichte Laufzeit ist nahezu linear in der Dimension. Mit einer etwas schlechteren Laufzeit können wir zeigen, dass N-fache ganzzahlige Programme mit konstanter Blockgröße in stark polynomieller Zeit gelöst werden können. Unter der Annahme der Exponentialzeithypothese ergänzen wir diese Ergebnisse mit einer Untergrenze für die Parameterabhängigkeit, die nahezu mit der Parameterabhängigkeit unserer Laufzeit übereinstimmt. Als direkte Konsequenz liefern wir die derzeit stärkste Untergrenze für N-fache ganzzahlige Programme.

Ein weiteres Problem das eng mit der ganzzahligen Programmierung zusammenhängt, ist das Problem des nächsten Vektors. Ein Gitter ist eine diskrete additive Untergruppe von $\mathbb{R}^n$. Das Problem des nächsten Vektors (CVP) fragt nach dem Gitterpunkt, der einem bestimmten Zielvektor am nächsten liegt. Ein wichtiges Werkzeug zur Lösung des nächsten Vektorproblems ist die Voronoizelle. Die Voronoizelle $\mathcal{V}$ eines Gitters $\Lambda \subseteq \mathbb{R}^n$ ist die Menge aller Punkte, für die 0 ein am nächsten liegender Gitterpunkt ist. Es ist ein Polytop, dessen Facetten durch Gittervektoren induziert werden, welche wir fortan Voronoi-relevante Vektoren nennen. Ein generisches Gitter hat exponentiell viele Voronoi-relevante Vektoren, was zu exponentiellem Speicherbedarf für bestimmte CVP-Algorithmen führt.

Wir führen im zweiten Teil dieser Arbeit den Begriff einer c-kompakten Gitterbasis $B \in \mathbb{R}^{n \times n}$ ein, die es ermöglicht, die Voronoi-relevanten Vektoren mit durch c begrenzten

Koeffizienten darzustellen. Mithilfe einer solchen Basis kann der Speicherbedarf des Algorithmus von Micciancio & Voulgaris für das nächste Vektorproblem von exponentiell auf polynomiell reduziert werden, während die Laufzeit exponentiell in c wird. Wir zeigen, dass für jedes Gitter eine n^2-kompakte Basis existiert, aber es gibt Gitter, für die keine c-kompakte Basis mit $c \in o(n)$ existiert. Wenn die Voronoi-Zelle ein Zonotop ist, existiert eine 1-kompakte basis B und wir erhalten einen Algorithmus für CVP mit einfach-exponentieller Laufzeit und polynomiellem Speicherbedarf, vorausgesetzt wir kennen eine 1-kompakte Basis.
Die Entscheidung ob ein gegebenes Gitter eine bestimmte Struktur aufweist, die dazu beiträgt, das Problem des nächsten Vektors effizienter zu lösen, ist ein Problem das fortwährend auftritt und zumeist keine einfache Lösung kennt. Der dritte Teil dieser Arbeit beschäftigt sich mit der spezifischen Struktur einer orthonormalen Basis. Obwohl der schnellste bekannte Algorithmus für dieses Problem exponentiell ist zeigen wir, dass dieses Problem zu NP $\cap$ co-NP gehört. Darüber hinaus kann es gelöst werden, indem wir ein einziges Problem des nächsten Vektors lösen. Wir zeigen auch, dass CVP in Polynomzeit lösbar ist wenn ein Separationsorakel für die Voronoizelle bereitgestellt wird.

Schlüsselwörter: ganzzahlige Programmierung, n-faches IP, Baumtiefe, Graverbasis, parametrisierter Algorithmus, Exponentialzeithypothese, Gitter, Voronoizelle, Problem des nächsten Vektors, co-NP, charakteristischer Vektor

Contents

Introduction

Integer programming in variable dimension.

Many problems arising in discrete optimization, such as the traveling salesman problem or the constraint satisfaction problem, can naturally be formulated in the setting of integer linear programming. An area that is concerned with the following problem.

Integer Linear Programming Problem

INSTANCE:	A system of linear equations $Ax = b$, $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, bounds $l \leq x \leq u$ with $l, u \in \mathbb{Z}^n$, an objective $f(x) = c^\intercal x$ with $c \in \mathbb{Z}^n$.
TASK:	Find a solution $x^\star \in \mathbb{Z}^n$ to $Ax = b$, $l \leq x \leq u$, minimizing $f(x)$, or assert that no solution exists.

An instance of the integer linear programming problem is called an integer linear program (ILP). Integer programming is a versatile tool not only for theoretical problems, it also has numerous applications in science, engineering or decision making. As famous NP-hard problems can be modeled as an integer program, it is no surprise that it belongs to Karp's classic 21 NP-hard problems [Kar72].

Though this indicates that solving a general ILP is a hard problem, several approaches have led to a variety of algorithms. Lenstra showed that if the number n of variables is fixed, the problem can be solved in polynomial time [Len83]. Shortly after, Kannan improved on this result by achieving a running time of $2^{\mathcal{O}(n \log_2(n))} \cdot \text{poly}(\varphi)$, where φ is the size of the input [Kan87]. Since then, all improvements on the running time have been restricted to the constant in the exponent; at the time of writing, the best constant was shown by Dadush [Dad12]. The question whether there is an algorithm with single-exponential running time is a famous open problem.

Another approach was chosen by Papadimitriou [Pap81]. He assumed the entries of A to be bounded and the number m of constraints to be small, whereas n is allowed to vary. Later, the running time of his approach was improved to $n \cdot (2m \, ||A||_\infty)^{\mathcal{O}(m^2)}$ by Eisenbrand & Weismantel [EW18].

These two approaches share the same idea. They specify certain parameters k of the problem (for Kannan, this is $k = n$, for Papadimitriou, this is $k = (m, ||A||_\infty)$) which they assume to be not too large. Then, they develop an algorithm for which the running time is $f(k) \cdot \text{poly}(\varphi)$, where f is any computable function only depending on the parameter and φ is the size of the instance. If the parameter is a constant, this gives a polynomial algorithm. This idea is the cornerstone in the field of parameterized complexity, and an algorithm whose running time can be written as $f(k) \cdot \text{poly}(\varphi)$ is called *fixed-parameter tractable (FPT)*, parameterized by k.

Yet another approach for solving integer linear programs employs the structure of the matrix A. Several combinatorial problems have their own specific structure. When modeled as an ILP, the matrix A inherits this structure. The most prominent example are certain flow problems, for which A is a totally unimodular matrix and thus lends itself to polynomial methods from linear programming. Another example comes from scheduling problems, for which the ILP models are N-fold integer programs. In an N-fold ILP, the constraint matrix $A \in \mathbb{Z}^{(r+Ns)\times Nt}$ is of block-shape,

$$A = \begin{pmatrix} B & B & \dots & B \\ C & & & \\ & C & & \\ & & \ddots & \\ & & & C \end{pmatrix},$$

where $B \in \mathbb{Z}^{r\times t}$ and $C \in \mathbb{Z}^{s\times t}$ have small dimensions, and are repeated N times. Additionally, $||A||_\infty$ is bounded by some constant $\Delta \geq 2$. It was a large success in the field of parameterized complexity when Hemmecke, Onn & Romanchuk [HOR13] showed that for these structured integer programs, there is also a parameterized algorithm. They showed that an N-fold ILP can be solved in time $L(r, s, t) \cdot N^3$, where L is a function independent of both the number of variables n and the number of constraints m.

In the following years, N-fold ILPs gained new momentum. On the side of applications, they were used to obtain fixed-parameter tractable algorithms for scheduling problems [Jan+19; KK18], social choice problems [KKM17b], or other problems [KKM17a]. On the theoretical side, it was shown that we can actually allow different matrices $B_1, \dots, B_N$ and $C_1, \dots, C_N$ and the problem is still fixed-parameter tractable, even with an improved running time [EHK18]. Jansen, Lassota & Rohwedder showed further improvements, presenting an algorithm with running time $n(\log(n))^6 \cdot \varphi^2 \cdot (rs\,||A||_\infty)^{\mathcal{O}(r^2 s + s^2)}$, presenting the first algorithm with a near-linear dependency on n [JLR19]. At the same time, it was shown in [KLO18] that an N-fold ILP is the most prominent example for an integer program where the parameter *tree-depth* of a certain associated graph is small.

The first part of this thesis is concerned with these theoretical results, building an extensive theoretic framework for integer programs with small tree-depth, culminating

in the currently fastest algorithms for these problems. In particular, we show that the dependency on the dimension n can be brought down to $n(\log_2(n))^2$. This has direct consequences for N-fold ILPs, where $n = Nt$. We emphasize these results:

1. The integer linear programming problem belongs to the complexity class FPT parameterized by the tree-depth of the dual graph $G_D(A)$ and $||A||_\infty$.
2. N-fold integer linear programs can be solved in time

$$n(\log_2(n))^2 \cdot \varphi^2 \cdot (2rs\,||A||_\infty)^{\mathcal{O}(r^2s+rs^2)},$$

   where φ denotes the encoding size of $||l||_\infty, ||u||_\infty$, and $||c||_\infty$.

In addition, we will revisit several classic results in the area of integer programming, strengthening these results further. Tardos showed that the dependency on $||c||_\infty$ is unnecessary in the case of linear programming [Tar86], and can be replaced with a term depending on the variable bounds l, u and the dimension n only. We develop a similar result, and show a generalization for separable convex objective functions.

If we further want to avoid dependency on l and u, we need to restrict the space in which we search for an optimal solution. Typically, one uses *proximity results*, stating that if $x^\star$ is an optimal solution to the LP relaxation, then an optimal solution to the integer program has to be close by. We prove a proximity result that is specifically tailored to our needs, and linear in n. Combining both techniques with the algorithm of Frank & Tardos [FT87], we obtain a running time independent of l, u, b and c.

3. N-fold ILPs and their generalization tree-fold ILPs are strongly fixed-parameter tractable. In particular, N-fold ILPs with constant block-sizes and $||A||_\infty$ are solvable in strongly polynomial time.

Most of the techniques we use are already known to work for separable convex objective functions as well, immediately giving similar results for separable convex functions. However, it was usually assumed that the encoding size of the quantity $f_{\max} := \max\{|f(x) - f(y)| : x, y \text{ feasible}\}$ is part of the input. But if f is given to us as an oracle, this assumption can be misleading. We are able to avoid this dependency.

As a complement to our derived algorithms, we provide lower bounds for the integer programming problem parameterized by $||A||_\infty$ and the tree-depth d, assuming the exponential time hypothesis (ETH). This is the widely accepted assumption that the 3-satisfiability problem cannot be solved in subexponential time. Though an asymptotically equal lower bound on the running time was already shown by [KP], we provide a more structured construction, yielding also lower bounds for specific subclasses of ILPs. To this end, we introduce a new parameter *topological height* of a tree. A direct consequence

is the currently best known lower bound for N-fold ILPs, which turn out to be the ILPs with topological height 2. Setting $\Delta := \max\{||A||_\infty, 2\}$, we have.

4. Unless the exponential time hypothesis fails, there is no algorithm for the integer programming problem with running time

$$\Delta^{o\left(\left(\frac{d}{\ell}+\frac{1}{12}\right)^{\ell}\right)}$$

in the worst case, where $d = \mathrm{td}(F)$ is the tree-depth, and $\ell = \mathrm{th}(F)$ is the topological height for any dual td-decomposition F of A.

5. Specifically, assuming ETH, no algorithm solves every generalized N-fold IP in time

$$\Delta^{o\left((r+s)^2\right)}.$$

Compact representations of Voronoi cells of lattices.

If we only want to know whether a given IP is feasible, we can rephrase the question in the following way. Given a polytope P, is $P \cap \mathbb{Z}^n$ non-empty? The set of integers $\mathbb{Z}^n$ is the easiest example of a *(Euclidean) lattice* Λ, i.e. a discrete additive subgroup of $\mathbb{R}^n$. In general, a lattice Λ is defined as the integer linear span of linearly independent vectors $b_1, \ldots, b_d \in \mathbb{R}^n$,

$$\Lambda = \{\alpha_1 b_1 + \cdots + \alpha_d b_d : \alpha_1, \ldots, \alpha_d \in \mathbb{Z}\}.$$

From this point of view, integer programming is a sub-area in the *geometry of numbers*. This field of research was initiated by Minkowski and focuses on the interplay between lattices and geometric structures, asking, for example, whether a polytope contains a lattice point. Perhaps not surprisingly, the mentioned approach of Tardos for reducing the dependency on the objective c relies on Diophantine approximation, an area closely related to the geometry of numbers. The famous algorithm of Kannan [Kan87] also relies on an important result in this area, the well-known *flatness theorem*.

One of the most prominent algorithmic problems in the geometry of numbers is the closest vector problem: Given a lattice $\Lambda \subseteq \mathbb{R}^n$ and a target $t \in \mathbb{R}^n$, which point $v^\star \in \Lambda$ attains the minimum distance $\min_{v\in\Lambda} \mathrm{dist}(t, v)$? This NP-hard problem has applications for integer programming, cryptography, and other areas, and has been intriguing mathematicians and computer scientists for several decades. As shown by Kannan [Kan87], this problem can be solved in time $n^{\mathcal{O}(n)}$. It was considered a huge breakthrough when Micciancio & Voulgaris [MV13] gave a deterministic single-exponential time algorithm for the problem, which we call the MV-algorithm.

Though theoretically this algorithm has a better running time, Kannan's algorithm is still faster in practice. One of several reasons is that Kannan's algorithm only needs

polynomial space, whereas the MV-algorithm needs exponential space, and accessing space is expensive when measured in realtime.

The second part of this thesis is concerned with the question whether a single-exponential time algorithm with polynomial space exists. Before we can provide more details, we introduce the *Voronoi cell* $\mathcal{V}$ of a lattice Λ. The Voronoi cell is the set of all points $x \in \mathbb{R}^n$ for which 0 is a closest lattice point. Since all points at least as close to 0 as to another lattice vector v can be described by a linear inequality, it should be no surprise that $\mathcal{V}$ is a polytope that can be described by a finite set $\mathcal{F} \subseteq \Lambda$ of lattice vectors. Maybe more surprisingly, the set $\mathcal{F}$ contains at most $2(2^n - 1)$ vectors, and for a generic lattice, this number is tight. The MV-algorithm essentially consists of two steps. First, it computes the set $\mathcal{F}$. In the second step, it uses the Voronoi cell to iteratively move from a lattice point v to a lattice point v' closer to the target t.

We introduce the concept of a *c-compact basis* for a lattice. This is a lattice basis $\{b_1, \ldots, b_n\}$ such that

$$\mathcal{F} \subseteq \left\{ \sum_{i=1}^{n} \alpha_i b_i : \alpha_i \in \mathbb{Z}, -c \leq \alpha_i \leq c \right\}.$$

Instead of storing the set $\mathcal{F}$ explicitly, we only need to store n vectors and can iterate over a superset of $\mathcal{F}$ roughly in time $(2c + 1)^n$. If c is a constant independent of n, this yields a single-exponential time algorithm with polynomial space. We obtain the following results.

6. Given a c-compact basis, we can solve the closest vector problem in time $(2c + 1)^{\mathcal{O}(n)} \operatorname{poly}(n)$ and polynomial space.
7. Every lattice $\Lambda \subseteq \mathbb{R}^n$ possesses an n^2-compact basis.

These results imply that we can already recover Kannan's running time asymptotically in the worst case, provided we know an n^2-compact basis. We are also able to show that with the chosen approach, we cannot hope to do better in general. However, there are certain lattices for which we obtain the best possible constant $c = 1$. This is captured by the following results, where a *zonotope* is the Minkowski sum of line segments.

8. There is a class of lattices that do not have a c-compact basis for $c \in o(n)$.
9. If the Voronoi cell $\mathcal{V}$ is a zonotope, then Λ has a 1-compact basis B. Moreover, we can choose B within $\mathcal{F}$.

The question how to find a c-compact basis arises naturally. More generally, if there is a class of lattices for which we can solve CVP more efficiently than with a general

algorithm (maybe if some additional information is given), how hard is it to recognize this class?

The closest vector problem with additional information.

Though CVP is NP-hard in general, there are several lattice classes for which better algorithms are known, provided some additional information is given.

For instance, CVP becomes trivial when we are given an orthogonal basis. Another example are the c-compact bases discussed in the previous paragraph. If we know such a basis, we can reduce the space requirements from exponential to polynomial. McKilliam, Grant & Clarkson showed that CVP can be solved in polynomial time on lattices that have an *obtuse superbasis* [MGC14]. In some sense, they show that if such a basis is known, the MV-algorithm can be implemented quite efficiently. Another improvement on the MV-algorithm was given by Dadush & Bonifas [DB15]. While Micciancio & Voulgaris require an exponential number of improving steps, Dadush & Bonifas show that a polynomial number suffices.

We generalize these results in some sense, showing that we can solve CVP in polynomial time as soon as we can separate over the Voronoi cell. As a consequence, the class of lattices in which the Voronoi cell is a zonotope allows for a polynomial-time algorithm, provided the generators of the Voronoi cell are known.

10. If we have a separation oracle for the Voronoi cell, we can solve the closest vector problem in polynomial time.

11. If the Voronoi cell is a zonotope and its generators are given, we can solve the closest vector problem in polynomial time.

In all examples mentioned before, it is not the lattice per se but the additional information that grants more efficient algorithms. For this reason, recognizing these lattice classes (and finding this extra information) is an important and reappearing problem. As it turns out, already for an easily describable lattice class this problem is quite involved.

Given a lattice basis B, can we decide whether the lattice generated by B has an orthonormal basis? If a lattice Λ has an orthonormal basis, it can be seen that it has to be a rotation of the standard Euclidean lattice $\mathbb{Z}^n$. Therefore, we call this problem the *rotated standard lattice problem (RSLP)*. This problem was considered by Lenstra & Silverberg [LS17], who gave a polynomial algorithm if additional information on the symmetry group of the lattice is provided. In general, only algorithms for the more general *lattice isomorphism problem (LIP)* are known. While the algorithm of Haviv & Regev for LIP has a running time of $n^{\mathcal{O}(n)}$ [HR14], we can show that the more specific RSLP can be reduced to solving the closest vector problem once, yielding a running

time of $2^{\mathcal{O}(n)}$. We also provide reason to hope for a polynomial algorithm by showing that the problem is in NP ∩ co-NP. Moreover, we show a connection to the *unimodular decomposition problem*: Given a positive definite unimodular matrix G, decide whether there exists a unimodular matrix U such that $U^\intercal U = G$. We believe that this problem is interesting in its own right.

12. The rotated standard lattice problem is in NP ∩ co-NP.
13. RSLP is equivalent to the unimodular decomposition problem.

Summarizing overview and sources.

The thesis starts with fixing notation and giving a brief background to some essential topics in Chapter 1. This is not meant to be an introduction to the area, and a certain background on the topics is assumed.

In Chapter 2, we derive new results for integer programming parameterized by the tree-depth and the largest absolute value of the entries in the constraint matrix. Several classic techniques are adapted and tailored to our setting. The chapter is based on the results in the articles [EHK18] and [Eis+19].

In Chapter 3, we investigate the question whether the Voronoi cell can be stored in polynomial space, using the novel concept of c-compact bases. The chapter is based on the results in the article [HRS20]. Parts have been published in the Master's Thesis of Reuland [Reu18].

In Chapter 4, we focus on subclasses of lattices on which CVP is easy to solve. Specifically, we show that a separation oracle for the Voronoi cell suffices for a polynomial-time algorithm. We also show that deciding whether an orthonormal basis exists is in NP ∩ co-NP. Parts of this chapter can be found in the arXiv preprint [Hun19].

1 Basics

In this section, we aim to fix the notation used in this thesis, and recall all necessary definitions and results established in the literature. Should the reader be unfamiliar with any of the mentioned concepts, every section provides references to text-books where more details on the subjects can be found. Especially the Sections 1.6 and 1.7 provide concepts that are assumed to be known for Chapter 2.

1.1 Notation for sets, vector spaces, and functions

This Section mostly fixes standard notation from linear algebra. We refer to [Lan87] for the theory. The computational results can for instance be found in [KV18].

The symbols $\mathbb{Z}, \mathbb{Q}, \mathbb{R}$ denote the sets of integral, rational, and real numbers, respectively. Together with addition and multiplication, $\mathbb{Z}$ is a ring, whereas $\mathbb{Q}$ and $\mathbb{R}$ are fields. For $a \in \mathbb{Z}$, we will also often use the abbreviations $\mathbb{Z}_{\geq a} := \{z \in \mathbb{Z} : z \geq a\}$ and $\mathbb{Z}_{>a} := \mathbb{Z}_{\geq a+1}$. For $n \in \mathbb{Z}_{\geq 1}$ and a set X, the set of n-tuples of elements in X is denoted by X^n. If $X \in \{\mathbb{Z}, \mathbb{Q}, \mathbb{R}\}$, the tuple is always understood as a column vector, and $\mathbb{R}^n, \mathbb{Q}^n$ are considered as vector spaces over $\mathbb{R}$, $\mathbb{Q}$ respectively, whereas $\mathbb{Z}^n$ is a free module over $\mathbb{Z}$. The spaces $\mathbb{R}^n, \mathbb{Q}^n, \mathbb{Z}^n$ are equipped with a scalar product, denoted by $\langle x, y\rangle := x^\intercal y$. The *support* of a vector $x \in \mathbb{R}^n$ is defined as $\operatorname{supp}(x) := \{i \in \{1, \dots, n\} : x_i \neq 0\}$.

We write $[n] := \{1, 2, \dots, n\} \subseteq \mathbb{Z}$, and $[n : m] := \{n, n+1, \dots, m\} \subseteq \mathbb{Z}$. In contrast, $[x, y] := \{\alpha \in \mathbb{R} : x \leq \alpha \leq y\} \subseteq \mathbb{R}^n$ denotes the closed interval between $x \in \mathbb{R}$ and $y \in \mathbb{R}$. This notation generalizes for $x, y \in \mathbb{R}^n$ to a *line segment* $[x, y] := \{x + \alpha(y - x) : \alpha \in \mathbb{R}, 0 \leq \alpha \leq 1\}$ in higher dimension. A set $S \subseteq \mathbb{R}^n$ is called *convex*, if for any two points $x, y \in S$, we have $[x, y] \subseteq S$.

For two sets $U, V \subseteq \mathbb{R}^n$, we define the *Minkowski sum* $U + V := \{u + v : u \in U, v \in V\}$. If $u \in U$, we write $u + V := \{u\} + V$ for short. Similarly, we denote $U - V := \{u - v : u \in U, v \in V\}$, $U - v := U - \{v\}$. In contrast to this, the set $\{u \in U : u \notin V\}$ will be

denoted with $U \setminus V$. For a finite set S, we denote by R^S the set of all maps $\varphi : S \to R$.

We will often consider restrictions of vectors, i.e. for a vector $x \in \mathbb{R}^n$, and an index set $S \subseteq [n]$, we are interested in the $|S|$-dimensional vector obtained from x by forgetting all entries x_i with $i \notin S$. In this case, we will consider elements in $\mathbb{R}^S$ as vectors indexed by S. Formally, if $S = \{i_1, \dots, i_m\} \subseteq [n]$ with $i_j < i_{j+1}$ for $j = 1, \dots, m-1$, there is a canonic embedding $\varphi : \mathbb{R}^{|S|} \to \mathbb{R}^n$ given by

$$(\varphi(x))_\ell := \begin{cases} x_j & \ell = i_j \in S, \\ 0 & \ell \notin S. \end{cases}$$

We identify $\mathbb{R}^S$ with the subspace $\varphi(\mathbb{R}^{|S|})$. The same holds for the sets $\mathbb{Q}^S$ and $\mathbb{Z}^S$, the vector space $\mathbb{Q}^{|S|}$, and the module $\mathbb{Z}^{|S|}$.

For $\alpha \in \mathbb{R}$, we will denote $\lceil \alpha \rceil := \min\{z \in \mathbb{Z} : \ z \geq \alpha\}$, and $\lfloor \alpha \rfloor := \max\{z \in \mathbb{Z} : \ z \leq \alpha\}$. We define

$$\lfloor \alpha \rceil := \begin{cases} \lceil \alpha \rceil & \lceil \alpha \rceil - \alpha \leq \frac{1}{2}, \\ \lfloor \alpha \rfloor & \lceil \alpha \rceil - \alpha > \frac{1}{2}. \end{cases}$$

This notation extends to vectors component-wise.

For $S \subseteq \mathbb{R}^n$, we denote the linear, affine, convex, and conic hull by

$$\begin{aligned} \operatorname{lin}(S) &:= \left\{ \sum_{i=1}^m \lambda_i a_i \;\middle|\; m \in \mathbb{Z}_{\geq 1}, \forall i \in [m] : a_i \in S, \lambda_i \in \mathbb{R} \right\}, \\ \operatorname{aff}(S) &:= s_0 + \operatorname{lin}(S - s_0), s_0 \in S, \\ \operatorname{conv}(S) &:= \left\{ \sum_{i=1}^m \lambda_i a_i \;\middle|\; m \in \mathbb{Z}_{\geq 1}, \forall i \in [m] : a_i \in S, \lambda_i \in \mathbb{R}_{\geq 0}, \sum_{i=1}^m \lambda_i = 1 \right\}, \\ \operatorname{cone}(S) &:= \{\lambda a \mid \lambda \in \mathbb{R}_{\geq 0}, a \in \operatorname{conv}(S)\}, \end{aligned}$$

respectively, where the choice of $s_0 \in S$ does not matter for the definition of $\operatorname{aff}(S)$. The *(affine) dimension* of a non-empty set $S \subseteq \mathbb{R}^n$, denoted by $\dim(S)$, is the dimension of the linear subspace $\operatorname{lin}(S - s_0)$ for some $s_0 \in S$, equivalently $\dim(S) := \dim(\operatorname{aff}(S) - s_0)$ for any $s_0 \in S$. We extend this notion by $\dim(\emptyset) := -1$.

For $x \in \mathbb{R}^n$ and $p \in \mathbb{R}_{\geq 1}$, we denote the *p-norm* by $||x||_p := (\sum_{i=1}^n |x_i|^p)^{1/p}$. If $p = 2$, we also call it the *Euclidean norm*. The *infinity-norm* is given by

$$||x||_\infty := \lim_{p \to \infty} ||x||_p = \max\{|x_i| : i \in [n]\}.$$

If $K \subseteq \mathbb{R}^n$ is a convex compact, full-dimensional set with $x \in K \Leftrightarrow -x \in K$, we define

$$||x||_K := \min\{r \geq 0 : \ x \in rK\}.$$

The function $||\cdot||_K$ is also a norm. We extend the infinity-norm to matrices $A \in \mathbb{R}^{m\times n}$ as

$$||A||_\infty := \max\{|a_{i,j}| : i \in [m], j \in [n]\}.$$

We denote a ball of radius $r > 0$ around $x \in \mathbb{R}^n$ by

$$B(x,r) := \{y \in \mathbb{R}^n : ||x-y||_2 \leq r\}.$$

The *relative interior* of a set $S \subseteq \mathbb{R}^n$ is the set

$$\text{rel. int}(S) := \{x \in S : \exists \varepsilon > 0 : B(x,\varepsilon) \cap S = B(x,\varepsilon) \cap \text{aff}(S)\}.$$

For a subspace $L \subseteq \mathbb{R}^n$ and a vector $x \in \mathbb{R}^n$, the *(orthogonal) projection onto* L, denoted by $\text{proj}_L : \mathbb{R}^n \to L$, is the unique linear map $x \mapsto \arg.\min_{y\in L} \text{dist}(x,y)$, where $\text{dist}(x,y) = \sqrt{(x-y)^\intercal(x-y)}$ is the Euclidean distance. We can compute the vector $\text{proj}_L(x)$ as follows. If $b_1,\ldots,b_r$ is an orthonormal basis of L, we set $\text{proj}_L(x) = \sum_{i=1}^r (b_i b_i^\intercal)x$. If $b_1,\ldots,b_r$ is any basis of L, we can compute the *Gram-Schmidt* orthogonalization $b_1^\dagger,\ldots,b_r^\dagger,x^\dagger$ of $b_1,\ldots,b_r,x$; it holds that $\text{proj}_L(x) = x - x^\dagger$ [Lan87]. The Gram-Schmidt orthogonalization can be computed in polynomial time.

A matrix $U \in \mathbb{Z}^{n\times n}$ is *unimodular*, if $\det(U) \in \{\pm 1\}$. By Cramer's rule equivalently, a matrix $U \in \mathbb{Z}^{n\times n}$ is unimodular if and only if $U^{-1} \in \mathbb{Z}^{n\times n}$. It follows that the unimodular matrices are in one-to-one correspondence to the group automorphisms of $\mathbb{Z}^n$. A matrix of full row rank is said to be in *Hermite normal form* if it has the form $(H,0)$, where H is a non-singular, lower triangular, non-negative matrix in which each row has a unique maximum entry, located on the main diagonal of H. It can be shown that for each rational matrix B, there exists a unimodular matrix U such that BU is in Hermite normal form. Moreover, we call BU the *Hermite normal form of* B, and it is unique [Sch86, Chap. 4].

For a map $f : A \to B$ and a subset $S \subseteq A$, we denote the *restriction of* f *onto* S by $f|_S$, i.e. $f|_S : S \to B$, $x \mapsto f(x)$. A continuous function $f : \mathbb{R} \to \mathbb{R}$ is called *convex* if $f(\gamma a + (1-\gamma)b) \leq \gamma f(a) + (1-\gamma)f(b)$ for $a,b \in \mathbb{R}$, $\gamma \in [0,1]$. A function $f : \mathbb{Z} \to \mathbb{Z}$ is called convex, if its *piecewise linear extension* $\overline{f} : \mathbb{R} \to \mathbb{R}$,

$$\overline{f}(x) := \begin{cases} f(x) & \text{for } x \in \mathbb{Z} \\ (x - \lfloor x \rfloor) f(\lceil x \rceil) + (\lceil x \rceil - x) f(\lfloor x \rfloor) & \text{for } x \notin \mathbb{Z} \end{cases}$$

is convex. A function $f : \mathbb{R}^n \to \mathbb{R}$ is called *separable convex*, if $f(x) = \sum_{i=1}^n f_i(x_i)$ for convex functions $f_i : \mathbb{R} \to \mathbb{R}$.

1.2 Polyhedra and linear programming

We assume that the reader is familiar with linear programming and polyhedra, and will only recall the basics briefly. For more details, we recommend the books [KV18] and [GLS93].

A *polyhedron* $P \subseteq \mathbb{R}^n$ is the intersection of finitely many *half-spaces* $H = \{x \in \mathbb{R}^n : a^\intercal x \leq \beta\}$, where $a \in \mathbb{R}^n$ and $\beta \in \mathbb{R}$. The set

$$\mathrm{rec}(P) := \{y \in \mathbb{R}^n \mid \forall x \in P, \lambda \in \mathbb{R}_{>0} : x + \lambda y \in P\}$$

is called the *recession cone* of P. If there exists a radius $R \in \mathbb{R}$ such that P is contained in a ball with radius R, we call P a *polytope*. A polyhedron is usually described by a system of linear inequalities, $P = \{x \in \mathbb{R}^n : Ax \leq b\} \subseteq \mathbb{R}^n$, where $A \in \mathbb{R}^{m \times n}$ is called the *constraint matrix*, and $b \in \mathbb{R}^m$ is the right hand side. An inequality $a^\intercal x \leq \beta$ is called *valid* for P if it is satisfied by all $x \in P$. A *hyperplane* $H = \{x \in \mathbb{R}^n : a^\intercal x = \beta\}$ (for $a \neq 0$) is called a *supporting hyperplane* for P, if the inequality $a^\intercal x \leq \beta$ is valid for P and $P \cap H \neq \emptyset$. A set $F \subseteq P$ is called a *face of* P if there exists a valid inequality $a^\intercal x \leq \beta$ for P such that we have $F = \{x \in P : a^\intercal x = \beta\}$. If $F \notin \{P, \emptyset\}$, we call it a *proper face*. If F is a face with $\dim(F) = \dim(P) - 1$, then F is called a *facet* of P, and the inequality $a^\intercal x \leq \beta$ is called a facet-defining inequality. If $\dim(F) = 0$, i.e. $F = \{v\}$ is a single point, then v is called a vertex, and H is said to *define* the vertex v.

The importance of polyhedra is most exemplaric in *linear programming*, an area that is concerned with minimizing a linear objective $c^\intercal x$, where x ranges over a polyhedron P. The *linear programming problem* can be stated as follows.

Linear Programming Problem

INSTANCE:	A system $Ax \leq b$ with $A \in \mathbb{Q}^{m \times n}$, $b \in \mathbb{Q}^m$, describing a polyhedron $P \subseteq \mathbb{R}^n$, a linear objective $f(x) = c^\intercal x$.
TASK:	*i*) Either assert that P is empty, or *ii*) find a vector $y \in P$ minimizing $c^\intercal x$ over P, or *iii*) find a vector $z \in \mathrm{rec}(P)$ such that $c^\intercal z \geq 1$.

An instance of this problem is called a linear program (LP). If the minimum exists, we call the LP *bounded*. By changing from $c^\intercal x$ to $-c^\intercal x$, it can be seen that it does not matter whether we maximize or minimize for linear objectives.

The function $f(x) = c^\intercal x$ is called the *objective function*, or just *objective*. Considering c to live in the dual space of $\mathbb{R}^n$, this term might also refer to the vector c. The set of *feasible points*, or *solutions* to the LP is the polyhedron P. A solution $x^\star$ is called *optimal* if for all $x \in P$, we have $c^\intercal x^\star \leq c^\intercal x$.

While we assume here that $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ is given by a system of linear inequalities, i.e. $A \in \mathbb{Q}^{m \times n}$ and $b \in \mathbb{Q}^m$, we can always introduce *slack variables* s such that we obtain an equivalent system $(A, \mathbf{1})\binom{x}{s} = b$, $s \geq 0$. This form will be preferable for Chapter 2. In the last Chapter, we will find ourselves in the situation that P is given to us *implicitly*, i.e. we do not know the system $Ax \leq b$. To distinguish this setting from linear programming, we introduce the following problem, where we do not make any assumptions on the representation of P.

Strong Optimization Problem

INSTANCE: A polyhedron $P \subseteq \mathbb{R}^n$ and a vector $c \in \mathbb{Q}^n$.
TASK: *i*) Either assert that P is empty, or
ii) find a vector $y \in P$ maximizing $c^\intercal x$ over P, or
iii) find a vector $z \in \text{rec}(P)$ such that $c^\intercal z \geq 1$.

In the late 70's, Khachiyan applied the *ellipsoid method* to linear programming [Kha80]. Based upon this, it was shown that *i*) linear programming can be solved in polynomial time, and *ii*) the strong optimization problem is equivalent to the following separation problem.

Strong Separation Problem

INSTANCE: A polyhedron $P \subseteq \mathbb{R}^n$ and a vector $y \in \mathbb{Q}^n$.
TASK: Decide whether $y \in P$, and if not, find a hyperplane separating y from P; more exactly, find a vector $c \in \mathbb{R}^n$ such that $c^\intercal y > \max\{c^\intercal x : x \in P\}$.

As we need to discuss this relation in more detail in Section 4.1, we keep this statement rather informal at this point, and refer to [GLS81] or [GLS93, Thm. 6.4.9] for details.

The result for linear programming was improved by Frank & Tardos, who showed that if A is integral, we can find an optimal solution in time polynomial in n and the encoding size of A alone, see [Tar86] and [FT87].

Whether the dependency on $||A||_\infty$ in the running time can be avoided as well is an interesting open problem.

1.3 Integer programming

We saw before that linear programming is concerned with finding an optimal point inside a polyhedron P. However, if we want to find an optimal *integral* point in P, the problem becomes much more difficult. As mentioned in the introduction, it belongs to Karp's classic 21 NP-hard problems. A good reference for the material presented in this section

is the text book of Schrijver [Sch86].

The problem of our concern is as follows.

Integer Linear Programming Problem

INSTANCE:	A system of linear equations $Ax = b$ with $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, lower and upper bounds $l, u \in (\mathbb{Z} \cup \{\pm\infty\})^n$ on the variables, a linear objective $f(x) = c^\intercal x$.
TASK:	Find a solution $x^\star \in \mathbb{Z}^n$ to $Ax = b$, $l \leq x \leq u$ minimizing $f(x)$, or assert that either no solution or no finite minimum exists.

An instance of this problem is called an *integer linear program (ILP)*. If the minimum exists, we call the ILP *bounded*. We will usually assume that $l, u \in \mathbb{Z}^n$, and discuss infinite bounds in a specific subsection. Given an ILP I, the problem arising by omitting the integrality constraints $x \in \mathbb{Z}^n$ is called the *LP relaxation of* I.

Though classically, integer programming is restricted to linear objectives $f(x) = c^\intercal x$, the results in this thesis also hold for separable convex objectives (cf. Section 1.1).

To distinct the cases of having a linear objective or a more permissive separable convex objective, we introduce the following generalization of ILP.

Integer Programming Problem

INSTANCE:	A system of linear equations $Ax = b$ with $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, lower and upper bounds $l, u \in (\mathbb{Z} \cup \{\infty\})^n$ on the variables, a separable convex objective $f(x)$.
TASK:	Find a solution $x^\star \in \mathbb{Z}^n$ to $Ax = b$, $l \leq x \leq u$ minimizing $f(x)$, or assert that either no solution or no finite minimum exists.

We call an instance of this problem an *integer program (IP)*. Again, if the minimum exists, we call the IP *bounded*. We will usually assume that $l, u \in \mathbb{Z}^n$, and discuss infinite bounds in a specific subsection. Given an IP I, the problem arising by omitting the integrality constraints $x \in \mathbb{Z}^n$ is called the *continuous relaxation of* I.

A specific IP will usually be presented in the following shape.

$$\begin{aligned} \min \quad & f(x) \\ s.t. \quad & Ax = b \\ & l \leq x \leq u \\ & x \in \mathbb{Z}^n \end{aligned} \tag{1.1}$$

The first planks between integer linear programming and the geometry of numbers were

laid by Lenstra. He showed that an integer linear program with a fixed number of variables can be solved in polynomial time [Len83], which was soon after improved by Kannan [Kan87]. Subsequently, a lot of research has been done to improve on this running time, though the asymptotic running time of roughly $n^{\mathcal{O}(n)}$ times a polynomial in the input size remains the state of the art. Whether the dependency on n can be reduced to 2^n without an exponential increase in other parameters is one of the most challenging questions in the area of integer programming.

As we are interested in solving integer programs w.r.t. a specific parameter, we also want to introduce a specific class of integer programs for which the chosen parameter turns out to be small. In particular, these are N-fold integer programs and their recursive generalization tree-fold integer programs.

Definition 1.1. *An integer program is called a (generalized)* N-fold IP with parameters r and s, *if the constraint matrix has the specific shape*

$$A = \begin{pmatrix} \overline{A}_1 & \overline{A}_2 & \cdots & \overline{A}_N \\ A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_N \end{pmatrix} \in \mathbb{Z}^{(r+Ns)\times Nt},$$

where $\overline{A}_i \in \mathbb{Z}^{r\times t}$, $A_i \in \mathbb{Z}^{s\times t}$.

The notion "generalized" refers to the fact that initially, there were only two submatrices $\overline{A}_i \equiv \overline{A}$ and $A_i \equiv A$ involved, which were repeated N times. If the diagonal matrices A_i are themselves in n'-fold shape, we obtain the following recursive definition of tree-fold IPs, introduced by Chen & Marx [CM18].

Definition 1.2. *A* (generalized) tree-fold integer program of depth 2 with parameters τ_1, τ_2 *is an* N*-fold integer program with parameters* $r = \tau_1$ *and* $s = \tau_2$. *For* $d \geq 3$, *a* (generalized) tree-fold integer program of depth d with parameters $\tau_1, \ldots, \tau_d$ *is an* N*-fold integer program with parameters* τ_1 *and* $\sum_{i=2}^{d} \tau_i$, *for which the matrices* A_i *are tree-fold integer programs of depth* $d-1$ *with parameters* $\tau_2, \ldots, \tau_d$.

Especially n-fold IPs gained much attention in the last years, as they are quite well suited to model scheduling problems, and led to new tractability-results in this and other areas.

1.4 Lattices and convex bodies

In this section, we give an overview on Euclidean lattices, convex bodies, and their interplay, commonly known as the *geometry of numbers*, which was already studied

by Minkowski. We refer to [Gru07] for a more detailed introduction. Though we provide general definitions, we usually assume *full-dimensional* lattices throughout this dissertation.

Lattices

A *lattice* Λ is a discrete additive subgroup of $\mathbb{R}^n$, $n \in \mathbb{Z}_{>0}$. Equivalently, a lattice is the set

$$\Lambda = \Lambda(B) := \{Bz : z \in \mathbb{Z}^d\} \subseteq \mathbb{R}^n$$

for some matrix $B \in \mathbb{R}^{n \times d}$ of full column rank. We say that Λ is the lattice *generated by* B, and B is a *basis* for Λ.

Depending on the context, we might also understand a basis of Λ as simply the (ordered) set of vectors $\{b_1, \ldots, b_d\}$, where b_i denotes the i-th column of B. If $S \subseteq \mathbb{Q}^n$ is a finite set of possibly linearly dependent vectors, then the set $\Lambda(S) := \{\sum_{v \in S} \alpha_v v : \alpha_v \in \mathbb{Z} \, \forall v \in S\}$ is a lattice, and also called the *lattice generated by* S. In general, this is not true if $S \subseteq \mathbb{R}^n$.

A lattice vector $v \in \Lambda \setminus \{0\}$ is called *primitive* if $\alpha v \notin \Lambda$ for all $\alpha \in (0, 1)$.

We define the *rank* or *dimension* of a lattice Λ as $\dim(\Lambda) := \dim(\mathrm{lin}(\Lambda))$. If $\dim(\Lambda) = n$, we call Λ a *full-dimensional lattice*, or a *lattice of full rank*.

Two matrices B_1, B_2 generate the same lattice if and only if there exists a unimodular matrix U such that $B_1 = B_2 U$. Thus, the *determinant of* Λ,

$$\det(\Lambda) := \sqrt{\det(B^T B)}$$

is well-defined.

To keep the notation simple, we will always assume Λ to be full-dimensional from now on, i.e. $d = n$.

The following geometric interpretations will be of use in Chapter 3. The *fundamental parallelepiped of a basis* $B = \{b_1, \ldots, b_n\}$ is the parallelepiped

$$\mathcal{P}(B) := \left\{ \sum_{i=1}^{n} \alpha_i b_i : 0 \leq \alpha_i < 1 \right\}.$$

The volume of $\mathcal{P}(B)$ is $\det(\Lambda)$. It is easy to see that the lattice translates of $\mathcal{P}(B)$ *tile the space*, i.e. $\mathbb{R}^n = \cup_{v \in \Lambda}(v + \mathcal{P}(B))$.

The *dual lattice* $\Lambda^\star$ of a lattice Λ is defined as

$$\Lambda^\star := \{y \in \mathrm{lin}(\Lambda) : x^\intercal y \in \mathbb{Z}\ \forall x \in \Lambda\}.$$

This definition is independent of the basis of Λ we chose, but simple calculations reveal the relation $(\Lambda(B))^\star = \Lambda(B^{-\intercal})$. The matrix $D = B^{-\intercal}$ is called the *dual basis corresponding to* B. The relation between the primal and the dual basis also reveals a geometric trade-off between the primal and dual lattice. For every vector $v \neq 0$ in the primal lattice, the dual lattice can be covered by equidistant hyperplanes orthogonal to v, where the distance is $\frac{1}{||v||_2}$. Moreover, if v is primitive, then each of these hyperplanes contains lattice points. To phrase it differently, if Λ is *dense* in some direction, then $\Lambda^\star$ has to be *sparse* in that direction, and vice versa. Later in this section, we will quantify this property also globally.

Two of the most prominent computational problems on lattices are the *shortest vector problem (SVP)* and its inhomogeneous version, the *closest vector problem (CVP)*.

Shortest Vector Problem (SVP)

INSTANCE: A lattice basis $B \in \mathbb{Q}^{n\times n}$.
TASK: Find a vector $v \in \Lambda \setminus \{0\}$ minimizing $||v||_2$.

Closest Vector Problem (CVP)

INSTANCE: A lattice basis $B \in \mathbb{Q}^{n\times n}$, a target vector $t \in \mathbb{Q}^n$.
TASK: Find a vector $v \in \Lambda$ minimizing $||t - v||_2$.

Though the problems were also considered for other norms, we will exclusively consider the 2-norm in this dissertation. Both problems are known to be NP-hard if we consider the ∞-norm. For the 2-norm, CVP is still NP-hard while we only know randomized reductions from NP-hard problems to SVP [MG02]. The common – slightly imprecise – phrasing is that SVP is NP-hard under randomized reductions. CVP is at least as hard as SVP, meaning that there is a polynomial reduction from SVP to CVP. A reduction in the other direction is not known.

Convex bodies

A *convex body* is a compact, convex set $K \subset \mathbb{R}^n$ of full dimension, $\dim(K) = n$. Moreover, K is called *centrally symmetric* if $K = -K$. The *norm induced by a centrally symmetric convex body* K is defined as

$$||x||_K := \min\{r \geq 0 : x \in rK\}.$$

This generalizes the well-known p-norms, for instance if K is a hypercube of side length 2, then $||\cdot||_K = ||\cdot||_\infty$, and $||\cdot||_1$ is induced by the standard cross-polytope. The *polar of a convex set* K is defined as $K^\star := \{y \in \mathbb{R}^n : y^\intercal x \leq 1 \, \forall x \in K\}$. If K was a convex body with 0 in its interior, then $K^\star$ is again a convex body with 0 in its interior; if K is a polyhedral cone, then $K^\star$ is a polyhedral cone. In both cases, $(K^\star)^\star = K$.

For a full-dimensional lattice $\Lambda \subseteq \mathbb{R}^n$, a convex body $K \subseteq \mathbb{R}^n$, and $k = 1, \ldots, n$, we define the *k-th successive minimum* as

$$\lambda_k(K, \Lambda) := \min\{r > 0 : \dim(rK \cap \Lambda) \geq k\}.$$

Hence, $\lambda_1(K, \Lambda)$ is the length of a shortest non-zero vector of Λ w.r.t. $||\cdot||_K$. If K is the Euclidean ball, we might omit K as an argument and simply write $\lambda_1(\Lambda)$.

The *covering radius* of a lattice Λ for a convex body K is the smallest scaling of K so that its translates cover the whole space. This is,

$$\mu(K, \Lambda) := \inf \left\{ r > 0 : \mathbb{R}^n \subseteq \bigcup_{v \in \Lambda} (v + rK) \right\}.$$

Geometry of numbers

Minkowski's geometry of numbers was centered around the study of these parameters, and in general, the interplay of lattices and their dual with convex bodies and their polar. One of the most important results is *Minkowski's fundamental theorem*, or Minkowski's fist theorem.

Theorem 1.3 (Minkowski's first theorem, cf. [Gru07, Ch. 22])**.** *Let $\Lambda \subseteq \mathbb{R}^n$ a full-dimensional lattice, and $K \subseteq \mathbb{R}^n$ a convex body that is centrally symmetric. If* $\mathrm{vol}(K) \geq 2^n \det(\Lambda)$*, then $K \cap (\Lambda \setminus \{0\}) \neq \emptyset$.*

Another important result was shown by Hinčin [Hin48], and states that there is a trade-off between the shortest vector of a lattice Λ and the covering radius of its dual. While Hinčin was the first providing an upper bound on the product of these two quantities, the best known upper bound is due to Banaszczyk [Ban96]:

Theorem 1.4. *Let $\Lambda \subseteq \mathbb{R}^n$ be a full-dimensional lattice, and $K \subseteq \mathbb{R}^n$ a centrally symmetric convex body. Then we have*

$$\lambda_1(K, \Lambda) \cdot \mu(K^\star, \Lambda^\star) \leq \mathcal{O}(n \ln(n)).$$

A special convex body associated with a full-dimensional lattice $\Lambda \subseteq \mathbb{R}^n$ is the *Voronoi*

cell. This is the set of all points at least as close to 0 as to any other lattice point:

$$\mathcal{V} = \mathcal{V}_\Lambda := \{x \in \mathbb{R}^n : \operatorname{dist}(x, 0) = \operatorname{dist}(x, \Lambda)\},$$

where the distance is measured in the 2-norm. The Voronoi cell has a lot of nice properties. First, by reformulating $(\operatorname{dist}(x, y))^2 = ||x - y||_2^2$, we obtain the equivalent but more utilizable definition

$$\mathcal{V} = \{x \in \operatorname{lin}(\Lambda) : 2v^\intercal x \leq v^\intercal v \ \forall v \in \Lambda\}.$$

It can be shown that $\mathcal{V}$ is a polytope whose facets are induced by lattice vectors. These vectors are called the *(strictly) Voronoi relevant vectors*, or simply *facet vectors*. The set of facet vectors will be denoted by $\mathcal{F}_\Lambda$. If $2v^\intercal x \leq v^\intercal v$ defines any proper face, then $v \in \Lambda$ is called a *weakly Voronoi relevant vector*. Hence, we consider the set of Voronoi relevant vectors to be contained in the set of weakly Voronoi relevant vectors. We can characterize the (strictly/weakly) Voronoi relevant vectors in the following way.

Lemma 1.5 ([CS92, Thm. 2])**.** *Let $\Lambda \subseteq \mathbb{R}^n$ be a full-dimensional lattice. A vector $v \in \Lambda \setminus \{0\}$ is weakly Voronoi relevant if and only if it is a shortest vector in the co-set $v + 2\Lambda$. A vector $v \in \Lambda \setminus \{0\}$ is strictly Voronoi relevant if and only if v and $-v$ are the only shortest vectors in the co-set $v + 2\Lambda$.*

This lemma immediately implies that every Voronoi cell has at most $2(2^n - 1)$ facet vectors.

We can also define the Voronoi cell for any lattice point v. Due to the lattice structure the Voronoi cell around v is just a translate, $v + \mathcal{V}$. Furthermore, the Voronoi cell tiles the space, i.e. $\mathbb{R}^n \subseteq \cup_{v \in \Lambda}(v + \mathcal{V})$, where the intersection $(v + \mathcal{V}) \cap (w + \mathcal{V})$ is always a face. For two dimensions, this is depicted in Figure 1.1.

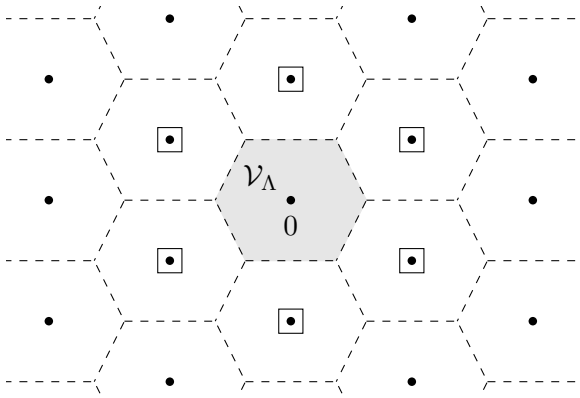

Figure 1.1 – A lattice Λ with its Voronoi cell $\mathcal{V}_\Lambda$ (shaded in gray), its Voronoi tiling, and its Voronoi relevant vectors (boxed lattice points).

In the light of the closest vector problem, the Voronoi cell turns out to be very useful. By definition, a vector $v \in \Lambda$ is a point closest to a target $t \in \mathbb{R}^n$ if and only if $t \in v + \mathcal{V}$. Hence, we only have to know in which Voronoi cell t is contained. The iterative slicer of Sommer et al. [SFS09] makes use of this fact. Later, Micciancio & Voulgaris [MV13] and also Dadush & Bonifas [DB15] refined these ideas. On a high level, we can follow the line segment $[0, t]$ and keep track of the Voronoi cell we are moving through. Whenever we leave a Voronoi cell $x + \mathcal{V}$, we enter a new Voronoi cell $y + \mathcal{V}$ due to their tiling property. The vector $v = y - x$ is Voronoi relevant, and we can update $x \leftarrow x + v$. This is discussed in more detail in Section 4.1.

1.5 Complexity

This section follows the presentation of [GLS93] and [KV18]. Assuming some background, we remain rather informal. The part on parameterized complexity is oriented on [Cyg+15]. Another thorough source is [DF13].

For $z, p \in \mathbb{Z}$, $q \in \mathbb{Z}_{>1}$ coprime to p, $r := \frac{p}{q}$, $x \in \mathbb{Q}^n$, and $A \in \mathbb{Q}^{m \times n}$, we define the *bit complexity*, or *encoding size*, as the number of bits needed to store the respective term,

$$\begin{aligned} \operatorname{size}(z) &:= \lceil \log(|z| + 1) \rceil + 1, & \operatorname{size}(r) &:= \operatorname{size}(p) + \operatorname{size}(q), \\ \operatorname{size}(x) &:= n + \sum_{i=1}^{n} \operatorname{size}(x_i), & \operatorname{size}(A) &:= nm + \sum_{i=1}^{m} \sum_{j=1}^{n} \operatorname{size}(a_{i,j}). \end{aligned}$$

For brevity, the size-function is also allowed to have several arguments, $\operatorname{size}(x_1, \ldots, x_\ell) := \sum_{i=1}^{\ell} \operatorname{size}(x_i)$, where each x_i is any of the aforementioned arguments. We give some standard estimates for calculating with encoding sizes, see [KV18, Chap. 4] for proofs. For rational numbers $r_1, \ldots, r_k$, we have $\operatorname{size}(r_1 \cdots r_k) \leq \operatorname{size}(r_1) + \cdots + \operatorname{size}(r_k)$ and $\operatorname{size}(r_1 + \ldots r_k) \leq 2(\operatorname{size}(r_1) + \cdots + \operatorname{size}(r_k))$. For rational vectors $y_1, \ldots, y_k$, we have $\operatorname{size}(y_1 + \cdots + y_k) \leq 2(\operatorname{size}(y_1) + \cdots + \operatorname{size}(y_k))$ and $\operatorname{size}(y_1^\intercal y_2) \leq 2(\operatorname{size}(y_1 + \operatorname{size}(y_2))$. For any rational square matrix A we have $\operatorname{size}(\det(A)) \leq 2 \operatorname{size}(A)$.

Theorem 1.6 ([KV18, Thm. 4.4])**.** *Suppose the rational LP* $\max\{c^\intercal x \mid Ax \leq b\}$ *has an optimum solution. Then it also has an optimum solution* $x \in \mathbb{Q}^n$ *with* $\operatorname{size}(x) \leq 4n(\operatorname{size}(A) + \operatorname{size}(b))$*, with components of size at most* $4(\operatorname{size}(A) + \operatorname{size}(b))$*.*

Throughout the whole thesis, we assume that the input for an algorithm consists of rational numbers. We say that an algorithm consisting of elementary arithmetic operations only[1] is *polynomial*, or *runs in polynomial time*, if there exist polynomials p_1, p_2 such that for each instance of size x, the number of performed operations is at most $p_1(x)$, and the and the size of all occurring numbers during the algorithm is bounded by $p_2(x)$. We say that the algorithm is *strongly polynomial* if, for an input comprising n rational numbers,

[1] This is, addition, subtraction, multiplication, division, and comparison of two numbers.

the algorithm is polynomial and the number of elementary arithmetic operations can be bounded by a polynomial in n alone. An *oracle* for a problem P takes as an input an instance of P, and outputs a correct answer. An *oracle algorithm* is an algorithm that is allowed to queue an oracle for a subproblem. An algorithm is said to run in *oracle-polynomial time*, if it is an oracle algorithm, and its number of arithmetic operations as well as the number of oracle calls is bounded by a polynomial.

The complexity class P contains all problems for which a polynomial time algorithm exists. The class NP is the class containing all decision problems which have the property that every YES-instance I has a certificate of size polynomial in size(I) that can be checked in polynomial time. The class co-NP is its complement; it contains all problems for which the NO-instances have a certificate that can be checked in polynomial time. Clearly, P $\subseteq$ NP. Whether the containment is strict is a famous open question (that is widely believed to be true).

A decision problem $P \in$ NP is NP-*complete* if for any other problem in NP there exists a Turing reduction to P. A problem P (not necessarily in NP) is called NP-hard if there is an NP-complete problem that can be Turing reduced to P. It follows that an optimization problem is NP-hard if its associated decision problem is NP-complete. To quote [GLS93] it is "quite customary convention" to also call every optimization problem NP-complete for which the associated decision problem is NP-complete.

As we generally do not hope for a polynomial algorithm if the problem at hand is NP-hard, the area of parameterized complexity is concerned with identifying how difficult NP-hard problems really are, and which parameters make them hard.

An algorithm is said to be *fixed-parameter tractable (FPT) parameterized by* k, if there is an integer α such that it runs in time $f(k) \cdot (\text{size}(x))^\alpha$, where x is the input with parameter k, and all numbers in the intermediate computations can be stored with $f'(k)(\text{size}(x))^{\alpha'}$ bits, where f' is again a computable function, and $\alpha' \in \mathbb{Z}_{\geq 1}$ is a constant. An algorithm is said to be *strongly FPT parameterized by* k if, for an input comprising n rational numbers, the algorithm is FPT and the number of elementary arithmetic operations can be bounded by a function $f(k) \cdot (n)^\alpha$, independent of the encoding size of the instance (again, f is a computable function, and $\alpha \in \mathbb{Z}_{\geq 1}$ a constant).

The complexity class FPT comprises all problems for which an FPT algorithm exists. The problems therein are also called *fixed-parameter tractable (FPT)*.

It follows that for every problem for which we have a strongly FPT algorithm, we also have a strongly polynomial algorithm whenever the parameter k is fixed. Note that the converse is not true: If the strongly polynomial algorithm has running time n^k with parameter k, we cannot separate the parameter from the size of the given instance.

In Chapter 2, we will be considering the integer programming problem parameterized by

$||A||_\infty$ and the treedepth of the dual graph of A, $\mathrm{td}_D(A)$, as will be defined in Section 1.7. This is, for an IP (1.1), we want to find an algorithm with running time

$$g(||A||_\infty, \mathrm{td}_D(A)) \cdot poly(n, \mathrm{size}(b, f_{\max}, l, u)),$$

where g is any computable function, and $f_{\max}$ is the absolute value over all values the objective f can assume.

1.6 The Graver basis

Graver bases were introduced by Jack E. Graver in 1975 [Gra75]. We follow the presentation in the book of De Loera, Hemmecke and Köppe [DHK13, Sec. 3], which also provides several historical notes and references.

Let $A \in \mathbb{Z}^{m\times n}$ be a matrix, and $\ker_\mathbb{Z}(A) := \ker(A) \cap \mathbb{Z}^n = \{x \in \mathbb{Z}^n : Ax = 0\}$. Two vectors $x, y \in \mathbb{R}^n$ are *conformal to each other* if they are in the same orthant, i.e. if $x_i y_i \geq 0$ for $i = 1, \ldots, n$. We write $x \sqsubseteq y$ if x and y are conformal and $|x_i| \leq |y_i|$ for all i. If $x, y \in \ker_\mathbb{Z}(A)$, $x \sqsubseteq y$ and $x \notin \{0, y\}$, observe that $y - x \sqsubseteq y$, and we can decompose $y = x + (y - x)$ into the sum of two conformal vectors. If such a decomposition does not exists, we call $y \in \ker_\mathbb{Z}(A)$ *indecomposable*. The set of indecomposable vectors in $\ker_\mathbb{Z}(A)$ is called the *Graver basis* of A, denoted by $\mathcal{G}(A)$. Originally, Graver defined this basis as the union of the Hilbert bases of the cones $\ker(A) \cap Q$, where Q runs through all 2^n orthants. These two definitions are equivalent, and as a Hilbert basis of a rational cone is finite, it follows that also $|\mathcal{G}(A)|$ is finite.

The Graver basis has some special properties that turn out to be very useful for our approach. In particular, when combined with separable convex functions, we obtain a certain *superadditivity*:

Lemma 1.7 (cf. [DHK13, Lem. 3.3.1])**.** *Let $f : \mathcal{D} \to \mathbb{R}$ be a separable convex function where $\mathcal{D} \subseteq \mathbb{R}^n$, and $x, y \in \mathbb{Z}^n$. The function f is superadditive, i.e. for a decomposition $y = \sum_{i=1}^m y_i$ of y into conformal vectors $y_i \sqsubseteq y$, $i = 1, \ldots, m$, we have*

$$f\left(x + \sum_{i=1}^m y_i\right) - f(x) \geq \sum_{i=1}^m \left(f(x + y_i) - f(x)\right). \tag{1.2}$$

Before we state further practical properties of the Graver basis, we give an idea of the proof. Due to separability, it suffices to show the inequality for one-dimensional convex functions. There, it follows from convexity for the decomposition into two vectors, $y = y_1 + y_2$. It extends to an arbitrary conformal decomposition of $y \in \mathbb{Z}$ by recursion.

Lemma 1.8 (cf. [DHK13, Sec. 3])**.** *Let x_0 be a feasible but not optimal solution to an IP (1.1), and $y \in \ker_\mathbb{Z}(A)$ such that $f(x_0 + y) < f(x_0)$. Then, we can write $y = \sum_{g\in S} \alpha_g g$*

as the conic combination of a set $S \subseteq \mathcal{G}(A)$ with coefficients $\alpha_g \in \mathbb{Z}_{\geq 1}$ subject to the following conditions.

i) *We have $g \sqsubseteq y$ for all $g \in S$.*

ii) *We have $|S| \leq 2n - 2$.*

iii) *For any integers $0 \leq \beta_g \leq \alpha_g$, the point $x_0 + \sum_{g \in S} \beta_g g$ is a feasible solution to the IP.*

iv) *There exists a vector $g \in S$ such that $f(x_0 + g) < f(x_0)$.*

v) *There exists a pair $(g, \alpha_g) \in S \times \mathbb{Z}$ such that*

$$f(x_0 + \alpha_g g) - f(x_0 + y) \leq \tfrac{2n-3}{2n-2}(f(x_0) - f(x_0 + y)).$$

Let x be a feasible solution to an integer program.

Points *iii)* and *iv)* imply the *test set property* of the Graver basis: If we do not find an improving direction y for a feasible solution x within the Graver basis, then x is already optimal. Some sources also say that the Graver basis is an *optimality certificate* for integer programming. This makes the Graver basis well suited for an augmenting approach.

A *feasible Graver step* for x is a vector $g \in \mathcal{G}(A)$ such that $x + g$ is again feasible for the IP. In the following, we will omit the reference "for x" for brevity. A feasible Graver step $g \in \mathcal{G}(A)$ such that $f(x + g) < f(x)$ is called a *Graver augmenting step*. A graver augmenting step g, together with an integer $\lambda \in \mathbb{Z}_{\geq 1}$ such that $x + \lambda g$ is feasible with $f(x + \lambda g) < f(x)$ is called a *Graver augmenting step pair*, where λ is called the *step length*. A Graver augmenting step pair $(g, \lambda) \in \mathcal{G}(A) \times \mathbb{Z}_{\geq 1}$ is called a *Graver-best step pair* if $f(x + \lambda g) \leq f(x + \mu h)$ for every Graver augmenting step pair $(\mu, h) \in \mathcal{G}(A) \times \mathbb{Z}_{\geq 1}$.

The *Graver augmenting procedure* for a feasible solution x of an IP (1.1) can be described in one sentence: As long as it exists, find a Graver-best step pair (g, λ), and update $x \leftarrow x + \lambda g$. By Lemma 1.8, the Graver-best step pair (g, λ) for a feasible solution x_0 satisfies

$$f(x_0 + \lambda g) - f(x^\star) \leq \tfrac{2n-3}{2n-2}\,(f(x_0) - f(x^\star)) \tag{1.3}$$

for any optimal solution $x^\star$. By iteratively finding a Graver-best step pair, one eventually reaches an optimal solution. Resolving the recursive estimate (1.3), this leads to $\mathcal{O}(n) \log_2(f(x_0) - f(x^\star))$ many iterations.

1.7 Graphs associated with constraint matrices

Before we will consider graphs that are induced by constraint matrices, we will fix some notation as used in [KV18]. A *graph* $G = (V, E)$ will be understood as a pair of sets, where V is the set of *vertices* and $E \subseteq V \times V \setminus \{(v, v) : v \in V\}$ is the set of *edges*. If the sets V and E are not mentioned explicitly, we will use $V(G)$ and $E(G)$ to refer to them. A *tree* is a connected graph without cycles, and a *rooted tree* F is a tree together with a designated element $r \in V(F)$, called the *root* of F. A *rooted forest* is a graph comprising a collection of rooted trees. Again, if the root is not explicitly mentioned, we refer to it with $r(F)$. The degree-1 vertices $v \neq r$ are called the *leaves* of F. Let (u, v) be an edge in a tree with u being closer to the root than v. We say that u is a *parent* of v, and v is a *child* of u. The *height* of a rooted tree F is the maximum number of vertices on a root-leaf path, and denoted by $\text{height}(F)$. If $u, v \in V(F)$ are two vertices in a tree, then there is a unique path between them, denoted by $P(u, v)$ and understood as a subgraph, i.e. it has an edge set and a vertex set.

A rooted tree F is called a *binary tree* if every vertex has at most two children. A binary tree is called *full* if $r(F)$ is the only vertex of even degree. A binary tree F is called *balanced* if there is an integer d such that every leaf has either depth d or $d - 1$.

Let $A \in \mathbb{Z}^{m \times n}$ be a matrix from an IP (2.1). The *primal graph* of A is the graph $G_P(A) = (V, E)$ with vertex set $V = \{1, \dots, n\}$ corresponding to the variables of the IP, and an edge for any two variables appearing together in a constraint. This is, $E = \{(i, j) : \exists\, k \in \{1, \dots, m\} :\ A_{k,i}, A_{k,j} \neq 0\}$. The *dual graph* of A is the primal graph of $A^\intercal$; i.e. the graph $G_D(A) = (V, E)$ with vertex set $V = \{1, \dots, m\}$ corresponding to the rows of A, and edge set $E = \{(i, j) : \exists\, k \in \{1, \dots, m\} :\ A_{i,k}A_{j,k} \neq 0\}$. An example for a matrix with its dual graph is given in the first two pictures of Figure 1.2.

The tree-depth of a graph

According to [Rei+14], the graph parameter tree-depth was considered under different names. As a textbook reference, we mention [NO12]. Given a graph G and a rooted forest F with $V(F) = V(G)$, consider the *closure of* F, which is defined by $V(\text{cl}(F)) := V(F)$, and

$$E(\text{cl}(F)) := \{(u, v) \in E(F) :\ \exists \text{ root-leaf path } P : u, v \in E(P)\}.$$

We say that u and v have an ancestral relationship in F if and only if $(u, v) \in E(\text{cl}(F))$.

Definition 1.9. *The* tree-depth *of a graph G, denoted by* $\text{td}(G)$*, is the minimum height of a rooted forest F such that $G \subseteq \text{cl}(F)$. A forest F for which $G \subseteq \text{cl}(F)$ is called a* tree-depth decomposition of G*, or* td-decomposition of G *for short. If additionally* $\text{height}(F) = \text{td}(G)$*, we call F an* optimal td-*decomposition.*

As in our applications G stems from a matrix A, we will always assume that G is connected. Otherwise, the matrix A decomposes into blocks where the rows (and columns) in each block have disjoint support, and the integer program decomposes into integer programs of smaller dimension. A good example for a td-decomposition is a DFS-tree (Depth-First Search) on G. This example is special in the sense that it has the additional property $E(F) \subseteq E(G)$, which is in general not required. An example of the dual graph of a matrix together with a td-decomposition is depicted in Figure 1.2.

While $G \subseteq \mathrm{cl}(F)$ is a rather global description, we can describe a td-decomposition with a more local property. For every edge $(u, v) \in E(G)$, the vertices u and v have to be in an ancestral relationship in F meaning that either v is on a path from the root to u or the other way around.

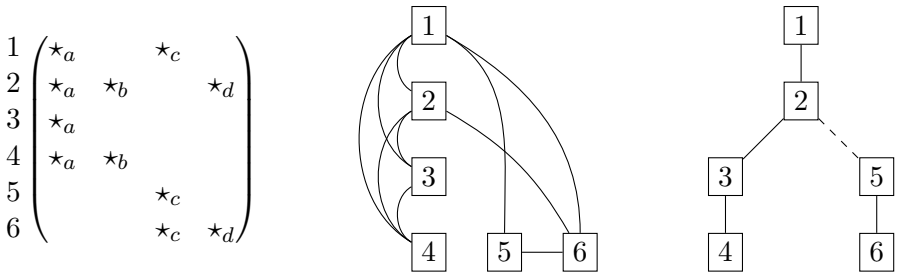

Figure 1.2 – A matrix with its dual graph $G_D(A)$ and a td-decomposition of $G_D(A)$.

The entries $\star_x$ in column $x \in \{a, b, c, d\}$ represent the non-zero elements. Column a induces a clique on vertices $\{1, 2, 3, 4\}$, column c induces a clique on $\{1, 5, 6\}$, and column d induces the edge $(2, 6)$. Column b does not contribute any edges, as $\mathrm{supp}(b) \subseteq \mathrm{supp}(a)$. To the right, a possible td-decomposition is shown. Note that the dashed edge does not exist in $G_D(A)$. It is easy to see that this small example is optimal. The td-decomposition has tree-depth 4, and the first column induces a clique of size 4, providing a lower bound on the tree-depth. For brevity, we will denote $\mathrm{td}_P(A) := \mathrm{td}(G_P(A))$ and $\mathrm{td}_D(A) := \mathrm{td}(G_D(A))$ for a given matrix A, and call these quantities the *primal tree-depth of A*, *dual tree-depth of A* respectively.

While structured IPs such as N-folds and tree-folds (Definitions 1.1 and 1.2) are essentially already encoded with a td-decomposition, we can compute a td-decomposition for a general graph in time $2^{(\mathrm{td}(G))^2} \cdot |V(G)|$, using an algorithm of Reidl et al. [Rei+14]. As this running time will be dominated by other terms, we will usually assume a td-decomposition to be given, and remember this step only when proving the main theorems.

The topological height of a tree

We continue with some notions specific to our work. As mentioned before, we always assume that the graph G is connected. In our applications, we are usually interested in how often the tree branches out on a root-leaf path, i.e. the maximal number of vertices with more than one child on any root-leaf path. To get a better grasp on this, we will introduce the notion of *topological height* next.

In the following let F be a rooted tree. A vertex $v \in V(F)$ is *degenerate* if it has exactly one child. Otherwise, v is called *non-degenerate* (including the leaves). The *topological height* of F is the height of the tree F' we obtain by contracting all degenerate vertices into their children. Alternatively, the topological height is the maximum number of non-degenerate vertices on any root-leaf path, denoted by $\mathrm{th}(F)$.

Let $P = (r = v_1, v_2, \ldots, v_N)$ be a root-leaf path in F, and $v_{j_1}, \ldots, v_{j_\ell}$ the ordered sequence of non-degenerate vertices on this path, i.e. $j_i < j_{i+1}$. Then the *first level height of P in F*, denoted by $k_1(P)$, is simply j_1, and for $2 \leq i \leq \ell$, the *i-th level height* is $k_i(P) := j_i - j_{i-1}$. To simplify notation later, we define $k_i(P) := 0$ for $i > \ell$. As $k_1(P)$ is actually independent of the root-leaf path P we choose, we also define $k_1(F) = k_1(P)$. For $i = 2, \ldots, \mathrm{th}(F)$, we define

$$k_i(F) = \max_{P\text{: root-leaf path}} k_i(P).$$

An example is given in Figure 1.3. The depicted tree F on vertices $\{1, \ldots, 7\}$ has topological height 3, the paths from the root vertex 1 to vertices 5 and 6 have level heights $2, 1, 1$, whereas $P(1, 7)$ has level heights $2, 2$.

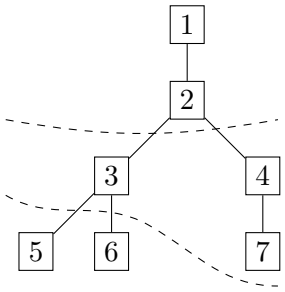

Figure 1.3 – The levels of a rooted tree F with topological height 3. We have $(k_1(F), k_2(F), k_3(F)) = (2, 2, 1)$, whereas $\mathrm{height}(F) = 4$.

Our techniques will heavily depend on the recursive structure on a td-decomposition F. If v is the unique non-degenerate vertex closest to the root, then we can decompose F into a path $P(r, v)$ together with rooted trees F_i, where the roots $r(F_i)$ are the children of v in F. If the graph stems from a matrix A, we can retract the decomposition onto A. More specifically, since the subtrees F_i are different branches, there are no edges

between them. As vertices correspond to rows, and edges indicate two rows sharing a variable, this means that the rows corresponding to $V(F_i)$ are orthogonal to the rows corresponding to $V(F_j)$ for $i \neq j$.

Lemma 1.10. *Let the dual graph of $A \in \mathbb{Z}^{m\times n}$ have a* td*-decomposition F with* $\mathrm{th}(F) \geq 2$. *Interpreting $V(F) = \{1, \dots, m\}$ as the index-set for the rows of A, there exists a partition of F into a path P and rooted trees $F_1, \dots, F_d$, together with a partition of the index set of the columns $\{1, \dots, n\} = T_1 \sqcup \dots \sqcup T_d$, such that the following conditions hold.*

i) Each F_i is a td*-decomposition of the submatrix $A_i := A_{V(F_i)\times T_i}$.*

ii) We have $\mathrm{th}(F_i) \leq \mathrm{th}(F) - 1$.

iii) Every non-zero entry of A appears either as an entry in one of the matrices A_i, or in one of the matrices $\overline{A}_i := A_{V(P)\times T_i}$.

Given A and F as an incidence matrix, these partitionings can be computed in time $\mathcal{O}(m + n)$.

By permuting rows and columns, we can use this lemma to assume that a given constraint matrix is in the following *block-shape*:

$$A = \begin{pmatrix} \overline{A}_1 & \overline{A}_2 & \dots & \overline{A}_d \\ A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_d \end{pmatrix}. \tag{1.4}$$

This shape will simplify the notation in later proofs. Note that if the matrices A_i have all the same dimension, we are left with a generalized d-fold IP.

Proof of Lemma 1.10. Let $v \in V(F)$ be the unique non-degenerate vertex closest to the root $r(F)$, and denote its children by $r_1, \dots, r_d$. For $i = 1, \dots, d$, define F_i to be the subtree of F rooted in r_i. More specifically, if we orient all edges in F away from r, then $V(F_i)$ is the set of all vertices that are reachable from r_i (including r_i itself), and F_i is the induced subgraph on these vertices. Since v is a non-degenerate vertex on every root-leaf path in F and not contained in any F_i, we have $\mathrm{th}(F_i) \leq \mathrm{th}(F) - 1$.

Moreover, there is no edge in $G_D(A)$ between $V(F_i)$ and $V(F_j)$ for $i \neq j$. Thus, for every column c there is at most one subtree F_i such that all row-indices corresponding to non-zero entries are contained in $V(P) \cup V(F_i)$. Hence, defining T_i to be the index set of all columns that have non-zero entries within the rows of $V(F_i)$ provides disjoint sets.

However, as a column might have non-zero entries only in the rows with index in $V(P)$, we define

$$T_i := \{j \in \{1,\dots,n\} : \exists \ell \in V(F_i) : A_{\ell,j} \neq 0\}, \qquad i = 2,\dots,d,$$
$$T_1 := \{j \in \{1,\dots,n\} : j \notin \cup_{i=2}^{d} T_i\}.$$

Now, Points *i)* and *ii)* follow from the definition.

The running time is trivial, as we simply have to perform a BFS (Breadth-First Search), starting in the root, and ending as soon as we find a vertex of degree at least 3. Its children are the roots of the subtrees F_i. □

The tree-width of a graph

Though we will be mainly concerned with the tree-depth of the dual graph of A, the parameter *tree-width* of a graph will also be discussed. The tree-width of a graph plays an important role in fixed-parameter tractable algorithms. Informally, it measures how close to a tree a given graph is. Formally, we first define a *tree-decomposition* of a graph G (not to be confused with a td-decomposition).

Definition 1.11. *A tree F is a tree-decomposition of a graph G, if there exists a mapping $\phi : V(F) \to 2^{V(G)}$ such that the following hold for vertices $u, v \in V(G)$ and* nodes *$s, t \in V(F)$.*

1. *$V(G) = \cup_{r \in V(F)} \phi(r)$.*
2. *If $v \in \phi(s) \cap \phi(t)$, then there exists an s-t-path in F, and $v \in \phi(w)$ for every vertex w on this path.*
3. *If $(u, v) \in E(G)$, then there exists a vertex $w \in V(F)$ such that $u, v \in \phi(w)$.*

The sets $\phi(r) \subseteq V(G)$, $w \in V(F)$ are usually called the bags of F. *The* tree-width $\mathrm{tw}(G)$ *is the minimum over all tree decompositions F of* $\max_{v \in V(F)} |\phi(v)| - 1$.

We can rephrase the conditions, saying that each edge of G has to be contained in at least one bag of F, and all bags containing a certain vertex $v \in V(G)$ have to be connected. If G arises from a constraint matrix A, we will again write $\mathrm{tw}_P(A) := \mathrm{tw}(G_P(A))$ and $\mathrm{tw}_D(A) := \mathrm{tw}(G_D(A))$ for short.

A *path-decomposition* of a graph G is a tree-decomposition in which the forest F is a path. The *path-width* of G is the minimum width over all path-decompositions of G.

2 Integer programming in variable dimension

Introduction

In this Chapter, we are concerned with solving an integer program (IP) of the form

$$\begin{aligned} \min \quad & f(x) \\ s.t. \quad & Ax = b \\ & l \leq x \leq u \\ & x \in \mathbb{Z}^n, \end{aligned} \tag{2.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a separable convex function with $f|_{\mathbb{Z}^n} : \mathbb{Z}^n \to \mathbb{Z}$ given by a comparison oracle, $A \in \mathbb{Z}^{m \times n}$, $b \in \mathbb{Z}^m$, and $l, u \in \mathbb{Z}^n$. We explicitly focus on finite bounds, though our results can be extended to possibly infinite bounds $l, u \in (\mathbb{Z} \cup \{\pm\infty\})^n$, as discussed in Section 2.5.3. The constraints $l \leq x \leq u$ are sometimes called the *box constraints*. Additionally, we will assume that A has a certain, sparse block structure. To be more precise, we will assume that the dual graph $G_D(A)$ of A (formed by taking a vertex for each row and connecting two vertices if the rows share a variable) is connected and has small tree-depth, see Section 1.7 for the precise definitions. An example for IPs with small dual tree-depth are N-fold integer programs, in which the constraint matrix has the shape

$$A = \begin{pmatrix} B & B & \dots & B \\ D & & & \\ & D & & \\ & & \ddots & \\ & & & D \end{pmatrix}.$$

The dimensions of the matrices $B \in \mathbb{Z}^{r \times t}$, $D \in \mathbb{Z}^{s \times t}$ should be imagined as small, whereas they have a large number $N \in \mathbb{Z}_{\geq 1}$ of copies. These instances occur when modeling scheduling or social choice problems, to name two examples.

Instead of solving the whole problem at once, we choose an augmenting approach, and split it into two steps.

(1) Decide whether (2.1) is feasible, and find an initial feasible solution x.

(2) Run an *augmenting procedure:*

   a) If it exists, find an augmenting vector $y \in \ker_{\mathbb{Z}}(A)$, i.e. find y such that $x + y$ is feasible, and $f(x + y) < f(x)$.

   b) As long as such a vector y exists, update $x \leftarrow x + y$ and iterate.

Let us develop some general thoughts and outline the structure of this chapter. Assuming for now that we have an initial solution x_0, we can easily see that Step (2) terminates eventually, by observing that the number of points with $l \leq x \leq u$ is finite. However, if we could choose y optimally, a single iteration would suffice. Therefore, we do not expect that finding a globally optimal y is easier than the original problem. We will perform a local search instead, i.e. we will only look for short vectors $y \in \ker_{\mathbb{Z}}(A)$. A crucial notion for this local search will be the Graver basis $\mathcal{G}(A)$, as introduced in Section 1.6. Recall that if the solution x_0 is not optimal, then an augmenting step y (i.e. $f(x_0 + y) < f(x_0)$) can be found within the Graver basis.

Thus, if we know a bound on the length of all Graver basis elements $g \in \mathcal{G}(A)$, we only need to consider vectors that are at most as long. This bound will be derived in Section 2.1, and denoted by $g_1(A)$, depending only on $||A||_\infty$ and the structure of A. This structure is more involved than simply the tree-depth of $G_D(A)$, but to give a feeling for the order of magnitude, the rather crude estimate $g_1(A) \leq (2^{\mathrm{td}(G_D(A))+1}\,||A||_\infty + 1)^{2^{\mathrm{td}(G_D(A))}}$ is always satisfied.

Having this bound at our disposal, we turn our attention to the augmenting procedure in Step (2). We will specify each iteration of the procedure and derive an auxiliary IP that will help us finding a short augmenting vector y. This IP will be called the *augmentation IP*, introduced in Section 2.2. In Subsection 2.2.1 we show how to solve this IP. In Subsection 2.2.2 we show that we can find a sufficiently good augmenting vector y by solving a small number of augmentation IPs, and analyze the convergence of the procedure. Lastly, in Subsection 2.2.3, we will introduce a data structure called *convolution tree*, allowing us to speed up the the procedure of repeatedly solving the augmentation IPs. The main theorem of this first part can be stated as follows.

Theorem 2.1. *Let an IP* (2.1)*, together with a* td*-decomposition* F *of* $G_D(A)$ *and an initial feasible solution* x_0 *be given. Define* $\zeta_{l,u} := ||u - l||_\infty$*. We can find an optimum* $x^\star$ *in time*

$$n \log_2(n) \cdot \log_2(f(x_0) - f(x^\star)) \cdot \log_2(\zeta_{l,u}) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{height}(F))}\,.$$

Moreover, the encoding size of all occurring numbers is bounded by $h(\text{size}(A), \text{height}(F)) \cdot \text{poly}(n, \text{size}(l, u))$*, where* h *is any computable function.*

In order to obtain a full algorithm for integer programming, there are several things left to discuss, split into distinct sections.

Consider first the term $f(x_0) - f(x^\star)$ in the running time. For separable convex functions, this is classically estimated by $f_{\max} := \max\{|f(x)| : x \; feasible \; for \; (2.1)\}$, a number that is assumed to be part of the input (for instance, cf. [DHK13, Thm. 3.4.1], [HOR13, Lem. 4.2]). But if f is given to us as some reasonable function evaluable on $\mathbb{Z}^n$ (say $f(x) = \sum_{i=1}^n 2^{x_i}$), or even by an oracle, this assumption might not be valid. If $f(x) = c^\intercal x$ is a linear function however, we can immediately limit $f_{\max} \leq ||c||_1 \, ||u - l||_\infty$. Moreover, by Frank & Tardos, we can replace c by an equivalent objective c', for which we can limit $||c'||_\infty$ in terms of $||l||_\infty$, $||u||_\infty$, and n alone [FT87]. In their spirit, we will derive a similar bound for separable convex objectives and improve upon their bound for linear objectives in Section 2.3. An important observation for achieving this goal is that our algorithm only needs a comparison oracle, hence we only need existence of an alternative objective. It is also shown that the bounds we get are essentially tight.

But what if the box constraints themselves are already rather large (Section 2.4)? The most prominent approach is to deploy a *proximity result*. Informally, these are results showing that to any optimal solution $x^\star \in \mathbb{R}^n$ of the LP relaxation (or continuous relaxation, if f is separable convex), there is an optimal integral solution $z^\star \in \mathbb{Z}^n$ near by, i.e. we can limit the quantity $||x^\star - z^\star||_1$. We show a novel proximity result for the case of small tree-depth, and deploy algorithms in the literature to find the fractional solutions in Subsection 2.4.1. Since it is sometimes desirable to avoid solving the continuous relaxation, we will also present another approach. Let us rephrase the problem for a moment. Given a polytope P and the lattice $\mathbb{Z}^n$, find a point in $P \cap \mathbb{Z}^n$ minimizing the objective function. For some large enough $k \in \mathbb{Z}_{\geq 0}$, it is possible to first find a solution $\hat{x}$ that is optimal within the sparser lattice $2^k\mathbb{Z}^n$, and then refine this solution, obtaining a point $x^\star$ that is optimal within the lattice $2^{k-1}\mathbb{Z}^n$. As it turns out, for the solutions $\hat{x}$ and $x^\star$ another version of our proximity result holds, allowing us to limit the search space. After running $k \in \mathcal{O}(\log_2(||u - l||_\infty))$ iterations we end up with an optimal solution in the initial lattice $\mathbb{Z}^n$, i.e. for the initial IP (Subsection 2.4.2).

In Section 2.5, we will discuss the initial assumptions. This is, in Subsection 2.5.1, we will see that finding an initial feasible solution is not really harder than optimizing. This is done via the common approach of introducing slack variables, and it is left to show that we do not change the parameters involved. We will also discuss the case of infinite bounds $l, u \in (\mathbb{Z} \cup \{\pm\infty\})^n$. In this case, we first have to know whether our IP is bounded. This problem is not decided automatically in our augmenting procedure, and the situation differs quite drastically for linear or separable convex objectives. See Subsection 2.5.2 for the discussion. If the IP at hand is bounded, we can guess the distance $||x_0 - x^\star||_\infty$ of x_0

to a closest optimal solution $x^\star$, and replace the dependency on $||u-l||_\infty$ by $||x_0-x^\star||_\infty$. For the guessing, we have to accept an additional factor of $\log_2(||x_0-x^\star||_\infty)$, giving a somewhat output-sensitive running time. For linear objectives, however, we can derive artificial bounds on an optimal solution x in terms of the entries in b and the finite entries in l, u. See Subsection 2.5.3 for details.

The developed tools are used in Section 2.6 for obtaining algorithms for the entire problem. We discuss distinct cases of given IPs, e.g. whether the objective is linear or separable convex, or whether we have finite or infinite bounds, and prove the running times for these cases. Using all of our results from before, i.e. the proximity bound of Section 2.4.1 and the reduced objective function shown in Section 2.3.1, we are even able to provide a strongly polynomial parameterized algorithm for a wide class of integer programs, namely tree-fold ILPs as introduced in Section 1.7.

To give an idea of the efficiency of the algorithms one can achieve with this toolbox, we state two of them in the following theorem.

Theorem 2.2 (cf. Theorem 2.38)**.** *Let an IP* (2.1) *be given,*

$$g_1(A) \leq (2^{\mathrm{td}(G_D(A))+1}\,||A||_\infty + 1)^{2^{\mathrm{td}(G_D(A))}}$$

an upper bound for the ℓ_1-norm of the Graver-basis elements, and define

$$\zeta_{l,u} := ||u-l||_\infty \quad \textit{and} \quad f_{\max} := \max_{x,y \textit{ feasible}} (|f(x)-f(y)|).$$

We can decide feasibility, and find an optimum solution if it exists, in time

$$\mathcal{O}(n(\log_2(n))^2) \cdot \log_2(f_{\max}) \cdot (\log_2(\zeta_{l,u}))^2 \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{td}_D(A))}.$$

In particular, integer programming with a separable convex objective is fixed-parameter tractable with respect to the parameters $\mathrm{td}_D(A)$ *and* $||A||_\infty$.

If the objective is linear, we can decide feasibility and find an optimal solution if it exists, in time

$$\mathcal{O}(n^2(\log_2(n))^3) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{td}_D(A))} + LP,$$

where LP denotes the time needed to solve the continuous relaxation. In particular, tree-fold integer linear programs can be solved in strongly fixed-parameter tractable time.

Before we close this chapter, there are two more Sections. In Section 2.7, we discuss our choice of the objective function and other parameters for integer programming. This section is rather giving an overview on the literature than presenting new results. In Section 2.8, we provide lower bounds on the running time for integer programming, assuming the exponential time hypothesis.

2.1 An upper bound for the Graver basis elements

In this section, we show that the l_1-norm of the Graver basis elements of a constraint matrix A can be limited in terms of the tree-depth of $G_D(A)$, which is a crucial ingredient to our running time analysis.

The result will heavily rely on the following Proposition. In its first, weaker variant it was shown by Steinitz [Ste16], the bounds in the Lemma below were shown by Grinberg & Sevast'yanov [GS80].

Lemma 2.3 (Steinitz' Lemma, [Ste16], [GS80])**.** *Let $||\cdot||$ be any norm on $\mathbb{R}^m$, and $x_1, \ldots, x_n \in \mathbb{Z}^m$ such that $||x_i|| \leq 1$ for $i = 1, \ldots, n$, and $\sum_{i=1}^{n} x_i = 0$.*

Then there exists a permutation $\pi \in S_n$ such that

$$\forall k \in \{1, \ldots, n\} : \left|\left|\sum_{i=1}^{k} x_{\pi(i)}\right|\right| \leq m.$$

We will first show the bound on the Graver basis elements for a generic matrix. In order to obtain a bound on the Graver basis elements for a matrix with small tree-depth, we will use recursion on $\text{th}(F)$ for a td-decomposition F of $G_D(A)$. Observe that $\text{th}(F) = 1$ implies that $\text{height}(F) = m$. Therefore, the following lemma is the induction basis.

Lemma 2.4. *Let $A \in \mathbb{Z}^{m \times n}$, $n, m \geq 1$. Then, for any $x \in \mathcal{G}(A)$, we have*

$$||x||_1 \leq (2m\,||A||_\infty + 1)^m.$$

Proof. Let $g \in \mathcal{G}(A)$, and denote the i-th column of A by A_i. We will define a sequence of vectors $x_i \in \mathbb{Z}^m$, $i = 1, \ldots, ||g||_1$ in the following manner. If $g_i \geq 0$, we add g_i copies of A_i to the sequence. If $g_i < 0$, we add $-g_i$ copies of $-A_i$ to the sequence. Hence,

$$Ag = \sum_{i=1}^{m} \frac{g_i}{|g_i|} \underbrace{(A_i + \cdots + A_i)}_{|g_i| \text{ times}} = \sum_{i=1}^{||g||_1} x_i = 0 \quad \text{with} \quad ||x_i||_\infty \leq ||A||_\infty ,$$

and by the Steinitz Lemma there is a permutation π with $\left|\left|\sum_{i=1}^{k} x_{\pi(i)}\right|\right|_\infty \leq m\,||A||_\infty$ for all $k = 1, \ldots, ||g||_1$.

For the sake of contradiction, assume that $||g||_1 > (2m\,||A||_\infty + 1)^m$. Since we have

$$|\{x \in \mathbb{Z}^n : ||x||_\infty \leq ||A||_\infty\}| \leq (2m\,||A||_\infty + 1)^m,$$

there are two indices $k_1 < k_2$ such that

$$\sum_{i=1}^{k_1} x_{\pi(i)} = \sum_{i=1}^{k_2} x_{\pi(i)} \qquad \Leftrightarrow \qquad \sum_{i=k_1+1}^{k_2} x_{\pi(i)} = 0.$$

Define a vector $y \in \mathbb{Z}^n$ such that $Ay = \sum_{i=k_1+1}^{k_2} x_{\pi(i)} = 0$ by setting

$$y_i = |\{j \in \{k_1+1, \dots, k_2\} : x_{\pi(j)} = A_i\}| - |\{j \in \{k_1+1, \dots, k_2\} : x_{\pi(j)} = -A_i\}|.$$

Observe that due to the construction of the x_i, the vector y satisfies $y \sqsubseteq g$. Hence, we can decompose $g = y + (g - y)$ into the sum of two conformal vectors, which is a contradiction to $g \in \mathcal{G}(A)$, proving the claim. □

We are now ready to prove the main result of this section.

Proposition 2.5. *Let $A \in \mathbb{Z}^{m \times n}$, $A \neq 0$, F a* td*-decomposition of its dual graph, and define*

$$K := \max_{P:\text{root-leaf path}} \prod_{i=1}^{\text{th}(F)} (k_i(P) + 1)\,, \qquad g_1(A) := (3K\,||A||_\infty)^{K-1}. \tag{2.2}$$

Then $||x||_1 \leq g_1(A)$ for all $x \in \mathcal{G}(A)$.

Remark 2.6. *Though the definition of $g_1(A)$ depends on the* td*-decomposition F we choose, we will not list F as an argument for brevity. The reason for this is that unless we explicitly assume a* td*-decomposition F to be given, the proofs work for any bound $g_1(A)$ on the Graver basis elements.*

Proof of Proposition 2.5. We will induct on the topological height of F. In the base case, $\text{th}(F) = 1$, we have $K = k_1(F) + 1 = m + 1$, and can simply use Lemma 2.4:

$$||x||_1 \leq (2m\,||A||_\infty + 1)^m \leq (3K\,||A||_\infty)^{K-1}.$$

For the induction step, recall that by Lemma 1.10, we can assume that A is in block-shape (1.4) with matrices $\overline{A}_i \in \mathbb{Z}^{k_1 \times t_i}$ and $A_i \in \mathbb{Z}^{s_i \times t_i}$ for $i = 1, \dots, d$, and a corresponding decomposition of F into a path P and td-decompositions $F_1, \dots, F_d$ corresponding to the matrices $A_1, \dots, A_d$.

Accordingly, we can decompose a vector $g \in \mathcal{G}(A)$ into bricks $g_i \in \mathbb{Z}^{t_i}$, i.e. $g^\intercal =$

$(g_1^\intercal, g_2^\intercal, \ldots, g_d^\intercal)$. With this notation, we have

$$Ag = \begin{pmatrix} \overline{A}_1 & \overline{A}_2 & \ldots & \overline{A}_d \\ A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_d \end{pmatrix} \begin{pmatrix} g_1 \\ g_2 \\ \vdots \\ g_d \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^d \overline{A}_i g_i \\ A_1 g_1 \\ A_2 g_2 \\ \vdots \\ A_d g_d \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Since $g_i \in \ker_{\mathbb{Z}}(A_i)$ for every i, we further decompose every $g_i = u_{i,1} + u_{i,2} + \cdots + u_{i,r_i}$ into a sequence of r_i conformal vectors $u_{i,j} \in \mathcal{G}(A_i)$. Consider the subsystem

$$\sum_{i=1}^{d} \overline{A}_i g_i = \sum_{i=1}^{d} \overline{A}_i \left(\sum_{j=1}^{r_i} u_{i,j} \right) = 0 \in \mathbb{Z}^{k_1}.$$

Using the induction hypothesis (observe that $\mathrm{th}(F_i) \leq \mathrm{th}(F) - 1$, we have $||u_{i,j}|| \leq L_{K'} := (3\,||A||_\infty K')^{K'-1}$ for all i, j in range. For $i = 1, \ldots, d$, and for $j = 1, \ldots, r_i$, define vectors $v_{i,j} := \overline{A}_i u_{i,j} \in \mathbb{Z}^{k_1}$, and observe that $||v_{i,j}||_\infty \leq ||A||_\infty ||u_{i,j}||_1 \leq ||A||_\infty L_{K'}$. This yields a sequence

$$\sum_{i=1}^{d} \sum_{j=1}^{r(i)} v_{i,j} = 0.$$

Arguing as in the proof of Lemma 2.4, we see that we can have at most $(2\,||A||_\infty k_1 L_{K'} + 1)^{k_1}$ vectors $v_{i,j}$, therefore at most that many vectors $u_{i,j}$. More formal, by Steinitz there is an ordering $\pi : \{1, \ldots, \sum_{i=1}^d r(i)\} \to_{bij} \{(i,j) : 1 \leq i \leq d,\, 1 \leq j \leq r(i)\}$ such that for every $1 \leq \ell \leq \sum_{i=1}^d r_i$ we have

$$\left|\left| \sum_{i=1}^{\ell} v_{\pi(i)} \right|\right|_\infty \leq k_1\,||A||_\infty L_{K'}.$$

As there are only $(2k_1\,||A||_\infty L_{K'} + 1)^{k_1}$ integer vectors in this range, having more would imply that we can decompose g into the sum of conformal elements, which is a contradiction to $g \in \mathcal{G}(A)$.

Since we have $||u_{i,j}||_1 \leq L_{K'}$, we obtain

$$||g||_1 \leq L_{K'}(2\,||A||_\infty k_1 L_{K'} + 1)^{k_1} \leq (3\,||A||_\infty K)^{K-1}.$$

This finishes the proof. □

2.2 Iterative improvement

This section is dedicated to Step (2) of the algorithm, which we briefly recall.

(2) Run an *augmenting procedure:*

a) If it exists, find an augmenting vector $y \in \ker_{\mathbb{Z}}(A)$, i.e. find y such that $x+y$ is feasible, and $f(x+y) < f(x)$.

b) As long as such a vector y exists, update $x \leftarrow x+y$ and iterate.

By Lemma 1.8, there always exists a Graver augmenting step pair (α, g) such that

$$f(x_0 + \alpha g) - f(x^\star) \leq \tfrac{2n-1}{2n}(f(x_0) - f(x^\star)).$$

If we augment with the vector $y := \alpha g$ in every iteration, this would lead to $\mathcal{O}(n \log_2(f(x_0) - f(x^\star)))$ iterations, where $x^\star$ is any optimal solution. Usually, such an augmenting step pair was found by solving the following system for each fixed $\lambda = 1, 2, 3, \ldots, ||u-l||_\infty$, where f, x, A, l and u are taken from the initially given IP:

$$\begin{aligned} \min \quad & f(x+\lambda g) \\ s.t. \quad & Ag = 0 \\ & l \leq x + \lambda g \leq u \\ & g \in \mathcal{G}(A) \end{aligned} \tag{2.3}$$

However, this is not an IP according to our definition. In particular, the set of feasible integer points might not be representable as the intersection of a polyhedron with the lattice $\mathbb{Z}^n$. Luckily, for all practical purposes, we do not need to find an optimal Graver-best step pair; in order to show convergence, any feasible step pair $(y, \lambda) \in \ker_{\mathbb{Z}}(A) \times \mathbb{Z}_{\geq 1}$ with $f(x+\lambda y) \leq f(x+\mu g)$ for all $(g, \mu) \in \mathcal{G}(A) \times \mathbb{Z}_{\geq 1}$ suffices, as the improvement is at least as good as with Graver-best step pairs.

Recall that $||g||_1 \leq g_1(A)$ for all $g \in \mathcal{G}(A)$, and consider the following *augmentation IP derived from* (2.1) *for* x *and* λ*:*

$$\begin{aligned} \min \quad & f(x+\lambda y) \\ s.t. \quad & Ay = 0 \\ & \left\lceil \frac{l-x}{\lambda} \right\rceil \leq y \leq \left\lfloor \frac{u-x}{\lambda} \right\rfloor \\ & ||y||_1 \leq g_1(A) \\ & y \in \mathbb{Z}^n \end{aligned} \tag{2.4}$$

By setting $l' = \left\lceil \frac{l-x}{\lambda} \right\rceil$, $u' = \lfloor \frac{u-x}{\lambda} \rfloor$ and $f'(y) = f(x+\lambda y)$, this is an integer program in our sense. If we are given a comparison oracle for f, a comparison oracle for $f'(y) := f(x+\lambda y)$ for some fixed x can be implemented easily and an optimal solution y provides an augmenting step pair (λ, y).

This IP will be quite important to our discussion on its own, i.e. independent of the

precise values of λ and x. Therefore, we will often omit the reference to x and λ, and call an IP of the following, general shape an *augmentation IP*:

$$\begin{aligned} \min \quad & f(y) \\ s.t. \quad & Ay = 0 \\ & l \leq y \leq u \\ & ||y||_1 \leq g_1(A) \\ & y \in \mathbb{Z}^n \end{aligned} \tag{2.5}$$

Not that we will only consider this IP on its own, i.e. the vectors l and u do not refer any more to some underlying IP.

In Subsection 2.2.1, we are concerned with solving an augmentation IP (2.4) derived for some x and λ. To be precise, we will not compute an optimal solution $y^\star$, but a vector $\hat{y}$ such that the box constraints are satisfied, we have $f(x_0 + \hat{y}) \leq f(x_0 + y^\star)$, but we might have $||\hat{y}||_1 > g_1(A)$. However, as this constraint was only introduced to reduce the search space, we can only benefit from this relaxation.

In Subsection 2.2.2, we limit the total number of augmentation IPs we have to solve. First, we limit the number of augmentation IPs we have to solve for finding one augmenting step pair. In contrast to solving one augmentation IP for every possible value of $\lambda \in \mathbb{Z}_{\geq 1}$, we will only guess $\lambda \in \{2^k : k \in \mathbb{Z}_{\geq 0}\}$. This will reduce the quality of the augmenting pair we find roughly by a factor of 2, whereas the number of augmentation IPs we have to solve decreases logarithmically. Together with limiting the convergence of our approach, we can informally conclude that solving an (2.1) can be reduced to solving

$$\mathcal{O}(n \cdot \log_2(f_{\max}) \cdot \log_2(||u - l||_\infty))$$

augmentation IPs, where $f_{\max} := \max\{|f(x) - f(y)| : x, y \text{ feasible}\}$.

Thereafter, we will see that we do not have to solve every augmentation IP from scratch, but can set up a specific data structure that allows us to retrieve an augmenting vector y easily. As $||y||_1$ is bounded by $g_1(A)$ independent of the dimension, after augmenting $x \leftarrow x + y$ only a small number of constraints $l_i \leq y_i \leq u_i$ in the augmentation IP changes for the next iteration. A *convolution tree* will allow us to retrieve an optimal vector y for given bounds l, u efficiently. After performing the augmentation $x \leftarrow x + y$ we can moreover update the convolution tree to the new bounds $l - y$, $u - y$ in time $\log(n)$. This allows us to decrease the cost for one augmentation from linear in n to $\log_2(n)$.

2.2.1 Solving the augmentation IP

This section is dedicated to solving an augmentation IP (2.5). The proof is quite similar to the dynamic programming approach of Papadimitriou [Pap81], but taking advantage of our bound $g_1(A)$ for Graver basis elements, as well as the structure of a td-decomposition F.

Proposition 2.7. *Given an augmentation IP* (2.5)*, together with a* td*-decomposition F of $G_D(A)$, we can find a vector $y^\star \in \ker_{\mathbb{Z}}(A)$ such that $l \le y^\star \le u$, and*

$$f(y^\star) \le \min\{f(y) \mid y \text{ feasible for } (2.5)\}$$

in time

$$\mathcal{O}(n)(2\,||A||_\infty\, g_1(A) + 1)^{2\,\mathrm{height}(F)}.$$

Proof. We will show a slightly stronger statement. Given two integer vectors $s, t \in \mathbb{Z}^m$, we will construct a digraph G with source s and sink t with the following two properties. First, any s-t path in G corresponds to an integer vector $y \in \mathbb{Z}^n$ such that $s + Ay = t$, and the cost of the path is $f(y)$, where f is already the objective of the augmentation IP. Second, for every vector $g \in \mathcal{G}(A)$ such that $s + Ag = t$, there exists a path corresponding to g. Thus, if $s = t = 0$, a shortest path corresponds to a vector $y \in \ker_{\mathbb{Z}}(A)$ such that $f \le f(g)$ for all $g \in \mathcal{G}(A)$. As the proof works by induction on the topological height, the generalization to arbitrary vectors s, t is necessary for the induction step. The time complexity is then the time complexity for constructing the graph and finding a shortest path.

The digraph G will be constructed recursively, starting with the case $\mathrm{th}(A) = 1$ where A is just considered as an arbitrary $m \times n$-matrix. In this case, the vertex set of G comprises $n + 1$ layers $V(G) = L_0 \cup \cdots \cup L_n$, and all edges are between two consecutive layers, i.e. $E(G) \subseteq \cup_{i=1}^{n}(L_{i-1} \times L_i)$. For every coordinate $1 \le i < n$, each L_i is a copy of the point set $\{x \in \mathbb{Z}^m : ||x||_\infty \le ||A||_\infty\, g_1(A)\}$. The outer layers L_0 and L_n will contain a single point $s \in \mathbb{Z}^m$, $t \in \mathbb{Z}^m$ respectively. For a pair $(u, v) \in L_{i-1} \times L_i$, we put an edge $e = (u, v)$ if there exists a value $y_i \in [l_i, u_i] \cap \mathbb{Z}$ such that $v = u + y_i a_i$, where a_i is the i-th column of A. For every edge e, we will store the value y_i and the cost $f_i(y_i)$ with respect to the separable convex function $f = \sum_{i=1}^{n} f_i$ we want to minimize. The graph is depicted in Figure 2.1.

An s-t-path P with (ordered) edge set $E(P) = \{e_1, \ldots, e_n\}$ in G corresponds to a vector $(y_1, \ldots, y_n)$ such that $Ay = \sum_{i=1}^{n} y_i a_i = t - s$. If we set $s = t = 0$ and observe that $||Ay||_\infty \le ||A||_\infty\, ||y||_1$, it is easy to see that for every vector $g \in \mathcal{G}(A)$, the corresponding path exists in G. Thus, a vector y corresponding to a shortest path satisfies $f(y) \le f(g)$ for all $g \in \mathcal{G}(A)$. The induction hypothesis is finished by estimating $|V(G)| \le (n - 1)(2\,||A||_\infty\, g_1(A) + 1)^m + 2$ and $|E(G)| \le (n-2)(2\,||A||_\infty\, g_1(A) + 1)^{2m} + 2(2\,||A||_\infty\, g_1(A) + 1)^m$.

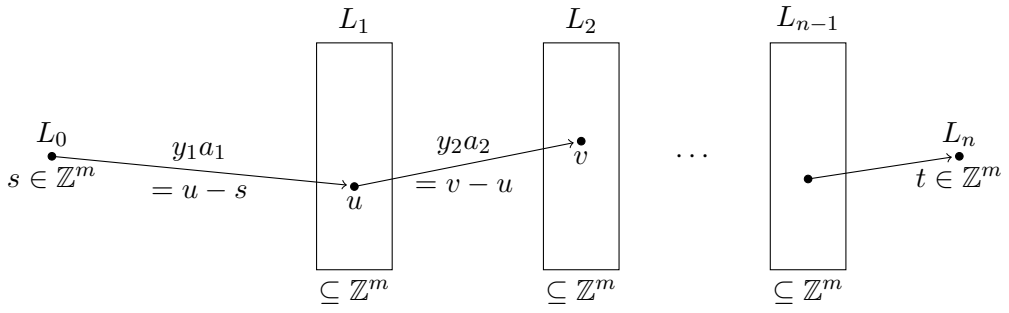

Figure 2.1 – The graph we construct to solve the augmentation IP if $\mathrm{th}(F) = 1$.

For the induction step, let F be a td-decomposition of $G_D(A)$, and v_0 the non-degenerate vertex closest to the root. We denote with d the number of children of v, the children themselves with $v_1, \ldots, v_d$, and the corresponding subtrees rooted in them with $F_1, \ldots, F_d$. Recall that k_1 denotes the number of vertices of the root-v_0 path in F. By Lemma 1.10 assume the matrix A to be in block-shape (1.4), with matrices $\overline{A}_i \in \mathbb{Z}^{k_1 \times n_i}$ in the upper block row, and matrices $A_i \in \mathbb{Z}^{|V(F_i)| \times n_i}$ on the block-diagonal below, where furthermore $\mathrm{th}(F_i) \leq \mathrm{th}(F) - 1$ and $\mathrm{height}(F_i) \leq \mathrm{height}(F) - k_1$ for all i in range.

Let $y \in \mathbb{Z}^n$ be such that $Ay = 0$ and $||y||_1 \leq g_1(A)$. We can decompose y into bricks $y_i \in \mathbb{Z}^{n_i}$ according to the matrices $\overline{A}_i$, and obtain

$$\sum_{i=1}^{d} \overline{A}_i y_i = 0 \in \mathbb{Z}^{k_1}.$$

Instead of corresponding to the columns of the whole matrix A, the layers L_i will now correspond to the submatrices $\overline{A}_i$, and instead of edges, we will recursively add subgraphs between two layers later. Again, the first layer will contain a single point $L_0 = \{s\}$, as well as the last layer $L_d = \{t\}$. The layers $L_1, \ldots, L_{d-1}$ in between will be copies of the set $\{x \in \mathbb{Z}^{k_1(F)} : ||x||_\infty \leq ||A||_\infty g_1(A)\}$, thus in particular independent of the choice of s and t.

Consider a pair of points $(u, v) \in L_{i-1} \times L_i$. The connection between them shall represent a vector $y_i \in \mathbb{Z}^{n_i}$ such that $\overline{A}_i y_i = v - u$, and $A_i y_i = 0$. If we pick a path through all layers using these connections, then we obtain a vector $y^\intercal = (y_1^\intercal, \ldots, y_d^\intercal)$ such that $Ay = t - s$. However, instead of connecting u and v by an edge, we will recursively glue a graph $G_{i-1,i}$ between the two layers L_{i-1} and L_i such that any path from u to v through $G_{i-1,i}$ is a feasible vector y_i.

Turning our attention to the matrix $\binom{\overline{A}_i}{A_i} \in \mathbb{Z}^{h_i \times n_i}$, where $h_i \leq \mathrm{height}(A)$, we see that the td-decomposition F_i extended by a path $P(r(F), r(F_i))$ on $k_1(F)$ vertices is a feasible td-decomposition $\overline{F}_i$ with root $r(\overline{F}_i) = r(F)$. Moreover, this td-decomposition satisfies $\mathrm{th}(\overline{F}_i) =\leq \mathrm{th}(F) - 1$ as we only extended F_i by a path, and we can use

induction. Construct a graph $G_{u,v}$ whose first layer is the point $\binom{u}{0} \in \mathbb{Z}^{k_1} \times \mathbb{Z}^{h_i-k_1}$ and $\binom{v}{0} \in \mathbb{Z}^k_1 \times \mathbb{Z}^{h_i-k_1}$ such that any $\binom{u}{0}$-$\binom{v}{0}$ path in $G_{u,v}$ corresponds to a vector y such that $\binom{u}{0} + \binom{\overline{A}_i}{A_i} y = \binom{v}{0}$. We glue this graph between layers L_{i-1} and L_i by identifying $u \in L_{i-1}$ with its zero-extension $\binom{u}{0}$, as well as $v \in L_i$ with $\binom{v}{0}$. As noted before, the inner layers of $G_{u,v}$ are independent of u and v, i.e. for a different pair $(u', v') \in L_{i-1} \times L_i$, the graphs $G_{u',v'}$ and $G_{u,v}$ will coincide except their start- and endpoint.

Hence we only have to insert a single graph $G_{i-1,i}$ between two consecutive layers L_{i-1} and L_i, and connect the vertices $u \in L_{i-1}$ and $v \in L_i$ appropriately to the layers of $G_{i-1,i}$.

It remains to analyze the size of the constructed graph. For $i \in \{1, \dots, d-1\}$, every layer L_i contains $(2\,||A||_\infty\, g_1(A) + 1)^{k_1}$ vertices. Between two layers L_{i-1} and L_i, we add a graph $G_{i-1,i}$ for the matrix $\binom{\overline{A}_i}{A_i} \in \mathbb{Z}^{\text{height}(F) \times n_i}$, which has

$$\begin{aligned} |V(G_{i-1,i})| &\leq (n_i - 1) \left(2 \left|\left| \begin{pmatrix} \overline{A}_i \\ A_i \end{pmatrix} \right|\right|_\infty g_1 \left(\begin{pmatrix} \overline{A}_i \\ A_i \end{pmatrix} \right) + 1 \right)^{h_i + k_1} \\ &\leq (n_i - 1)(2\,||A||_\infty\, g_1(A) + 1)^{\text{height}(F)} \end{aligned}$$

by induction. In total, we obtain at most

$$\begin{aligned} |V(G) \setminus \{s,t\}| &\leq (n-d)(2\,||A||_\infty\, g_1(A) + 1)^{\text{height}(F)} + (r-1)(2\,||A||_\infty\, g_1(A) + 1)^{k_1(F)} \\ &\leq (n-1)(2\,||A||_\infty\, g_1(A) + 1)^{\text{height}(F)} \end{aligned}$$

vertices. As every layer (either a layer L_i or a layer from the graphs $G_{i-1,i}$) has at most $(2\,||A||_\infty\, g_1(A) + 1)^{\text{height}(F)}$ vertices and there are $n-1$ inner layers, the number of edges is limited by

$$|E(G)| \leq (n-2)(2\,||A||_\infty\, g_1(A) + 1)^{2\,\text{height}(F)} + 2(2\,||A||_\infty\, g_1(A) + 1)^{\text{height}(F)}.$$

Thus, a BFS on the graph takes time at most $\mathcal{O}(n)(2\,||A||_\infty\, g_1(A) + 1)^{2\,\text{height}(F)}$, finishing the proof. □

Remark 2.8. *This proposition shows that we can find an augmenting step that is at least as good as the optimal solution of the augmentation IP. If we are really interested in finding an augmenting step y satisfying $||y||_1 \leq g_1(A)$, we could achieve this by increasing the dimension by one for keeping track of the* 1*-norm for each path in the graph.*

2.2.2 Optimizing via the augmentation IP.

Let an IP (2.1) with initial feasible solution x_0 be given. We saw in Lemma 1.8, Point v) that the Graver-best step pair always yields a sufficient improvement. However, as the Graver basis itself has a rather impractical structure (e.g., it is lacking a notion of

convexity), we will instead search for a vector y with $||y||_1 \leq g_1(A)$, together with a step length $\lambda = 2^k$, $k \in \mathbb{Z}_{\geq 1}$ such that $f(x + \lambda y)$ is as small as possible instead. Clearly, $||y||_1 \leq g_1(A)$ is satisfied by all Graver basis elements. Moreover, it is crucial to observe that we can restrict the search to a rather moderate amount of values for λ. If we restrict λ to powers of 2, the convergence only loses a factor of 2, while the number of values for λ decreases logarithmically.

We will first show that we can find one augmenting step pair of sufficient quality by solving not too many augmentation IPs. Then, we will limit the number of iterations we need. Together this yields a total number of augmentation IPs we have to solve.

Finding one augmenting step pair.

Lemma 2.9. *Let an IP* (2.1) *be given, together with a feasible solution* x_0*. Denote with* $x^\star$ *any optimal solution. There exists a pair* $(k, \overline{y}) \in \mathbb{Z}_{\geq 0} \times \mathbb{Z}^n$ *with* $||\overline{y}||_1 \leq g_1(A)$ *such that* $x_0 + 2^k\overline{y}$ *is feasible for* (2.1) *and*

$$f(x_0 + 2^k\overline{y}) - f(x^\star) \leq \tfrac{4n-5}{4n-4}(f(x_0) - f(x^\star)).$$

Moreover, we can find such a pair by either solving, or applying Proposition 2.7 to each augmentation IP (2.4) *derived for* x_0 *and* $\lambda = 2^k$*,* $k = 0, 1, \ldots, \lfloor \log_2(||u - l||_\infty) \rfloor$*.*

Proof. We first show the existence of a sufficient pair $(2^k, \overline{y})$. Consider a conformal decomposition

$$x^\star - x_0 = \sum_{g \in S} \alpha_g g \tag{2.6}$$

into at most $2n - 2$ Graver basis elements $S \subseteq \mathcal{G}(A)$. By Lemma 1.8 Point v), there exists an element $g \in S$ with

$$f(x_0 + \alpha_g g) - f(x^\star) \leq \tfrac{2n-3}{2n-2}\left(f(x_0) - f(x^\star)\right).$$

There exists a unique integer $k \in \mathbb{Z}_{\geq 0}$ such that $2^k \leq \alpha_g < 2^{k+1}$. Hence, we can write $2^k = \gamma\alpha_g$ for some rational number $\frac{1}{2} < \gamma \leq 1$. By convexity, we obtain

$$\begin{aligned}
f(x_0 + 2^k g) - f(x^\star) &\leq (1-\gamma)f(x_0) + \gamma f(x_0 + \alpha_g g) - \gamma f(x^\star) \\
&= \gamma\left(f(x_0 + \alpha_g g) - f(x_0)\right) + \left(f(x_0) - f(x^\star)\right) \\
&\leq \gamma\tfrac{1}{2n-2}\left(f(x^\star) - f(x_0)\right) + \left(f(x_0) - f(x^\star)\right) \\
&\leq \tfrac{1}{4n-4}\left(f(x^\star) - f(x_0)\right) + \left(f(x_0) - f(x^\star)\right) \\
&\leq \tfrac{4n-5}{4n-4}\left(f(x_0) - f(x^\star)\right).
\end{aligned}$$

Choosing $\overline{y} = g$ shows the first part. Moreover, observe that $\overline{y}$ is a feasible solution to

the augmentation IP (2.4) derived for x_0 and $\lambda = 2^k$. Hence, $f(x_0 + 2^k y^\star) \leq f(x_0 + 2^k \overline{y})$ for $y^\star$ being either an optimal solution to the augmentation IP or a vector obtained by Proposition 2.7. This shows that we can find a desired pair (k, y) by considering a sequence of augmentation IPs, and it remains to limit k.

But since every augmenting vector y has integral coefficients, and the bounds are finite, the step-length can be at most $||u - l||_\infty$, thus $k \leq \log_2(||u - l||_\infty)$. □

Limiting the number of improving steps.

Knowing that we have to solve $\log_2(||u - l||_\infty)$ augmentation IPs to find one improving step, we can iterate this procedure and find a sequence of solutions with improving objective value. We are now concerned with the question how fast this sequence converges.

Lemma 2.10. *Let an IP* (2.1) *be given, together with a feasible solution* x_0*. Let* $(x_i)_{i \in \mathbb{Z}_{\geq 1}}$ *be a sequence of solutions such that for each* $i \in \mathbb{Z}_{\geq 1}$*, we have*

$$f(x_{i+1}) - f(x^\star) \leq \tfrac{4n-5}{4n-4} \left(f(x_i) - f(x^\star)\right).$$

Then, for $k \geq (4n-5) \log_2 \left(f(x_0) - f(x^\star)\right)$*, we have* $f(x_k) = f(x^\star)$*.*

Using another estimate in the following proof (cf. [Eis+19, Lem. 12]), the estimate can be improved to $k \geq 3n \log_2(f(x_0) - f(x^\star))$.

Proof. Let k be maximal such that

$$1 \leq f(x_k) - f(x^\star) \leq \left(\tfrac{4n-5}{4n-4}\right)^k \left(f(x_0) - f(x^\star)\right).$$

Taking the logarithm, we obtain

$$\begin{aligned}
& \left(\tfrac{4n-5}{4n-4}\right)^k \left(f(x_0) - f(x^\star)\right) \geq 1 \\
\Leftrightarrow \quad & k \log_2 \left(\tfrac{4n-5}{4n-4}\right) \geq -\log_2 \left(f(x_0) - f(x^\star)\right) \\
\Leftrightarrow \quad & k \leq \frac{\log_2 \left(f(x_0) - f(x^\star)\right)}{\log_2 \left(1 + \frac{1}{4n-5}\right)} \\
\Rightarrow \quad & k < (4n-5) \log_2 \left(f(x_0) - f(x^\star)\right),
\end{aligned}$$

where we used strict concavity for the logarithm between 1 and 2, i.e. $\log_2(1 + \varepsilon) > \varepsilon$ for $0 < \varepsilon < 1$. Since $f|_{\mathbb{Z}^n} : \mathbb{Z}^n \to \mathbb{Z}$, the claim follows. □

Proposition 2.11. *Let an IP* (2.1) *together with a feasible solution* x_0 *be given, and define* $\zeta_{l,u} := ||u - l||_\infty$*. We can find an optimum solution* $x^\star$ *by solving at most*

$$\mathcal{O}(n) \log_2(f(x_0) - f(x^\star)) \log_2(\zeta_{l,u})$$

augmentation IPs (or applications of Proposition 2.7).

Proof. For one augmenting step, it suffices to solve $\mathcal{O}(\log_2(\zeta_{l,u}))$ many augmentation IPs, according to Lemma 2.9 (or, applying Proposition 2.7 that often). Among the solutions of these IPs, we choose the pair $(y^\star, \lambda^\star)$ minimizing $f(x+\lambda y)$, and update $x \leftarrow x + \lambda^\star y^\star$. By Lemma 2.10, $\mathcal{O}(n \log_2(f(x) - f(x^\star)))$ iterations suffice.

□

If we combine this proposition with Proposition 2.7, we obtain a running time that is quadratic in n.

Corollary 2.12. *Let an IP* (2.1) *together with a* td*-decomposition F of $G_D(A)$ and a feasible solution x_0 be given. We can find an optimum solution in time*

$$\mathcal{O}(n^2) \log_2(f(x_0) - f(x^\star)) \log_2(\zeta_{l,u}) \cdot (2\,||A||_\infty\, g_1(A) + 1)^{\text{height}(F)}.$$

As we will see in the next subsection, this can be improved to $n \log_2(n)$.

2.2.3 Convolutions and the convolution tree

Previously, we set up and solved a set of dynamic programs for each augmenting step, one for each $\lambda = 2^k$, $k = 0, 1, 2, \ldots$. Fix one value for λ. Since we are only interested in augmenting steps $x + \lambda y$ with $||y||_1 \leq g_1(A)$, the lower and upper bounds

$$\left\lceil \frac{l-x}{\lambda} \right\rceil \leq y \leq \left\lfloor \frac{u-x}{\lambda} \right\rfloor,$$

as well as the separable convex objective $f(x+\lambda y)$ change in at most $g_1(A)$ coordinates. In this subsection, we will use this fact. Instead of starting from scratch in every iteration, we will set up one data structure for each value $\lambda = 2^k$ in the beginning. This data structure will be storing the optimal solution for the augmentation IP derived for λ, and can be updated rather cheap in every iteration. More specific, only for the setup we will have to invest time linear in n once; afterwards, each updating is only logarithmic in n.

To this end, we introduce $\{\min, +\}$-*convolution* and the *convolution tree*. Convolutions in combination with integer programming were already used by Jansen & Rohwedder [JR19].

Definition 2.13. *Let $R \subseteq \mathbb{Z}^d$, and $a, b \in (\mathbb{Z} \cup \{\infty\})^R$. A map $c \in (\mathbb{Z} \cup \{\infty\})^R$ is the*

$\{\min, +\}$-convolution *of a and b, denoted by $c = \operatorname{convol}(a, b)$, if*

$$\forall\, r \in R \cap (R + R): \quad c(r) = \min_{\substack{r', r'' \in R \\ r' + r'' = r}} a(r') + b(r''),$$
$$\forall\, r \in R \setminus (R + R): \quad c(r) = \infty.$$

A map $w \in ((R \times R) \cup \{\texttt{undef}\})^R$ is called a witness *of c w.r.t. a, b if*

$$w(r) = (r', r'') \quad \Rightarrow \quad c(r) = a(r') + b(r''),$$
$$w(r) = \texttt{undef} \quad \Rightarrow \quad r \in R \setminus (R + R)$$

Computing the convolution of two functions efficiently is an interesting problem on its own. However, for our purposes it will be sufficient to use the naive approach, i.e. for every $r \in R$, we compute $c(r)$ by computing all possibilities $a(r') + b(r - r')$, $r' \in R$ and choose the best one.

Of course, only convoluting two maps is not very beneficial, as we naturally want to split our problem into $d \geq 2$ subproblems, where d is the degree of the first non-degenerate vertex in a td-decomposition of $G_D(A)$. To this end, we extend our previous definition.

Definition 2.14. *Let $A \in \mathbb{Z}^{m \times n}$, F be a* td*-decomposition of $G_D(A)$, $\rho \in \mathbb{Z}_{\geq 1}$ and $R := \{x \in \mathbb{Z}^{k_1(F)} : ||x||_\infty \leq g_1(A)\,||A||_\infty\}$. A* convolution tree *is a data structure $\mathcal{T}$ which stores two vectors $l_\mathcal{T}, u_\mathcal{T} \in (\mathbb{Z} \cup \{\pm\infty\})^n$, and a separable convex function $f_\mathcal{T} : \mathbb{R}^n \to \mathbb{R}$, and supports the following operations.*

`Init`(l, u, f) *initializes $\mathcal{T}$ to be in state $l_\mathcal{T} = l$, $u_\mathcal{T} = u$, $f_\mathcal{T} = f$.*

`Update`(i, l_i, u_i, f_i) *is defined for $i \in [n]$, $l_i, u_i \in \mathbb{Z}$ and a convex function $f_i : \mathbb{R} \to \mathbb{R}$. It updates the vectors $l_\mathcal{T}, u_\mathcal{T}$ in their i-th coordinate to the value l_i, u_i respectively, (i.e. $(l_\mathcal{T})_i \leftarrow l_i$ and $(u_\mathcal{T})_i \leftarrow u_i$), and the function $f_\mathcal{T}$ in its i-th coordinate to f_i, i.e. $(f_\mathcal{T})_i(x) \leftarrow f_i(x)$.*

$\sigma -$ `Update`(I, l_I, u_I, f_I) *is defined for $I \subseteq [n]$, and performs* `Update`(i, l_i, u_i, f_i) *for all $i \in I$ in ascending order.*

`Queue`(r, ρ) *is defined for $(r, \rho) \in R \times [0 : g_1(A)]$ and returns a vector $y \in \mathbb{Z}^n$ minimizing $f_\mathcal{T}(y)$ subject to the conditions $Ay = \binom{r}{0}$, $l_\mathcal{T} \leq y \leq u_\mathcal{T}$, $||y||_1 = \rho$.*

Lemma 2.15. *Given a matrix A together with a* td*-decomposition F, we can implement a convolution tree with the following time constraints.*

- `Init`(l, u, f) *can be done in time $2n(2\,||A||_\infty\, g_1(A))^{2\,\mathrm{height}(F)+2}$*
- `Update`(i, l_i, u_i, f_i) *can be done in time $(2\,||A||_\infty\, g_1(A))^{2\,\mathrm{height}(F)+2} \cdot \mathrm{th}(F) \log(n)$*

- $\sigma - \texttt{Update}(I, l_I, u_I, f_I)$ *can be done by calling* σ *times* `Update`*, i.e. in time* $\sigma \cdot (2\,||A||_\infty\, g_1(A))^{2\,\mathrm{height}(F)+2} \cdot \mathrm{th}(F)\log(n)$

- $\texttt{Queue}(r, \rho)$ *can be done in time* $g_1(A)$*.*

The size of the data structure is bounded by $h(g_1(A))\cdot\mathrm{poly}(n, \mathrm{size}(A))$ *for some computable function* h*.*

Proof. For the sake of presentation, we assume that f is given by an evaluation oracle. In the end, we will argue that a comparison oracle suffices.

Let us first fix some notation and outline the recursive idea. By Lemma 1.10, we can assume that A is of block-structure with blocks $\overline{A}_1, \ldots, \overline{A}_d$ in the top block row, and blocks $A_1, \ldots, A_d$ in a block-diagonal below the top row, where $\overline{A}_i \in \mathbb{Z}^{k_1 \times n_i}$ and $A_i \in \mathbb{Z}^{h_i \times n_i}$ with $h_i \leq \mathrm{height}(F) - k_1(F)$. Let $[n] = N_1 \cup \cdots \cup N_d$ be a partition of the index set of the columns according to the matrices $\overline{A}_1, \ldots, \overline{A}_d$, i.e. $n_i = |N_i|$. For a vector $y \in \mathbb{Z}^n$, let $y_i \in \mathbb{Z}^{N_i}$ denote its restriction to the coordinates in N_i. Observe that for a subset $U \subseteq [n]$, the two vector spaces $\mathbb{Z}^{|U|}$ and $\{x \in \mathbb{Z}^n : x_i = 0, i \notin N_i\} \subseteq \mathbb{Z}^n$ have a natural isomorphism. To simplify notation, we will understand $\mathbb{Z}^{N_i}$ as either one of the above vector spaces, depending on the context. This is, if we consider $\overline{A}_i y_i$, then $y_i \in \mathbb{Z}^{N_i}$ is a vector of dimension $|N_i|$, whereas we can glue two vectors $y_i \in \mathbb{Z}^{N_i}$, $y_j \in \mathbb{Z}^{N_j}$ together by simply writing $y_i + y_j$ for $i \neq j$.

Let $\mathcal{T}'$ be a full balanced binary tree for which the leaves are labeled $N_1, \ldots, N_d$, and an internal vertex is labeled $U = V \cup W$, where V and W are the labels of the children of U. Hence, the root has label $[n]$. Define $R := \{x \in \mathbb{Z}^{k_1} : ||x||_\infty \leq g_1(A)\,||A||_\infty\}$, and assume that we have two functions for each node $U = \cup_{i\in I} N_i$,

$$\phi_U : R \times [0 : g_1(A)] \to \mathbb{Z} \quad \text{and} \quad G_U : R \times [0 : g_1(A)] \to \mathbb{Z}^U,$$

with

$$\phi_U(r, \rho) = \min\{f(y) :\ y \in \mathbb{Z}^U, \sum_{i=1}^{d} \overline{A}_i y_i = r, A_i y_i = 0, l \leq y \leq u, ||y||_1 = \rho\},$$

$$G_U(r, \rho) = \arg\,.\min\{f(y) :\ y \in \mathbb{Z}^U, \sum_{i=1}^{d} \overline{A}_i y_i = r, A_i y_i = 0, l \leq y \leq u, ||y||_1 = \rho\}.$$

Then $\phi_U = \mathrm{convol}(\phi_V, \phi_W)$ and $G_U(r, \rho) = G_V(r_1, \rho_1) + G_W(r_2, \rho_2)$ where $((r_1, \rho_1), (r_2, \rho_2))$ is the pair witnessing (r, ρ). Moreover, $\min_\rho \phi_{[n]}(0, \rho)$ is the optimal value of the augmentation IP, and the optimal solution can be retrieved from $G_{[n]}$. Recursively replacing the leaves N_i with similar trees $\mathcal{T}_i'$ provides the full convolution tree. We give a detailed description.

The case $\mathrm{th}(F) = 1$. Set $d = n$ and $R := \{x \in \mathbb{Z}^m : ||x||_\infty \leq g_1(A)\,||A||_\infty\}$. To unify the notation, denote the columns of A by $\overline{A}_i$, i.e. comparing with the block shape (1.4), interpret each column of A as a block $\overline{A}_i$ of size $m \times 1$, and the matrices A_i to be empty. Consider a full balanced binary tree $\mathcal{T}$ on the columns $\overline{A}_i$ of A. Let $l_i \leq z \leq u_i$ denote the i-th coordinate of the box constraints. For every i, define $\phi_{\{i\}} \in (\mathbb{Z} \cup \{\infty\})^{R \times [0:g_1(A)]}$ by

$$\phi_{\{i\}}(r,\rho) := \begin{cases} \min\{f_i(z) : \overline{A}_i z = r,\ |z| = \rho, l_i \leq z \leq u_i\}\} & \textit{if } \min \textit{ exists},\\ \infty & \textit{otherwise}. \end{cases}$$

Accordingly, define

$$G_{\{i\}}(r,\rho) := \begin{cases} \arg\,.\min\{f_i(z) : \overline{A}_i z = r,\ |z| = \rho, l_i \leq z \leq u_i\}\} & \textit{if } \min \textit{ exists},\\ \texttt{undef} & \textit{otherwise}. \end{cases}$$

This is, the value $\phi_{\{i\}}(r,\rho)$ is only finite if $r = \pm\rho\overline{A}_i$.

For every internal node U with children V, W, we will define $\phi_U \in (\mathbb{Z} \cup \{\infty\})^{R \times [0:g_1(A)]}$ as $\phi_U := \mathrm{convol}(\phi_V, \phi_W)$, and the function G_U accordingly as follows. For all $(r,\rho) \in R \times [0 : g_1(A)]$ and $(r_1,\rho_1) \in R \times [0 : \rho]$, we queue $\phi_V(r_1,\rho_1) + \phi_W(r - r_1, \rho - \rho_1)$ if $r - r_1 \in R$, and memorize the pair (r',ρ') minimizing this sum.

Then, we set $\phi_U(r,\rho) := \phi_V(r',\rho') + \phi_W(r - r', \rho - \rho')$, and define $G_U(r,\rho) \in \mathbb{Z}^U \cup \{\texttt{undef}\}$ by

$$G_U(r,\rho) := \begin{cases} G_V(r',\rho') + G_W(r - r', \rho - \rho') & \text{if both functions are defined},\\ \texttt{undef} & \text{otherwise}. \end{cases}$$

(Here, we understand $\mathbb{Z}^V, \mathbb{Z}^W$ as subspaces of $\mathbb{Z}^n$ with disjoint support.) It is straightforward to observe that

$$AG_U(r,\rho) = \sum_{i \in U} \overline{A}_i (G_U(r,\rho))_i = r, \qquad l \leq (G_U(r,\rho)) \leq u, \qquad ||G_U(r,\rho)||_1 = \rho.$$

In the end, we obtain functions $\phi_{[n]}$ and $G_{[n]}$ as wanted.

The case $\mathrm{th}(F) \geq 2$. Now for the recursive case, consider again the first recursion level of tree $\mathcal{T}$ with leaves $N_1, \ldots, N_d$, where $[n] = \cup_i N_i$ is a decomposition of the variables according to according to the matrices $\overline{A}_i$. A td-decomposition for a matrix $\binom{\overline{A}_i}{A_i}$ can be obtained by extending the tree F_i by a path on k_1 vertices. This implies that the first level height for this tree-decomposition is $h_i := k_1 + k_1(F_i)$, and its topological height satisfies $\mathrm{th}(F_i) \leq \mathrm{th}(F) - 1$, i.e. we can use induction.

As we changed to the matrices $\binom{\overline{A}_i}{A_i}$ however, the functions ϕ_{N_i}, G_{N_i} are defined on $R_i := \{x \in \mathbb{Z}^{k_1 + k_1(F_i)} : ||x||_\infty \leq -g_1(A)\,||A||_\infty\} \times [0 : g_1(A)]$. If we wanted to convolute

different functions ϕ_{N_i}, ϕ_{N_j}, the input has different dimensions. Instead of embedding the sets R_i into $\mathbb{Z}^m$, observe that we are only interested in function values $G_{N_i}(r,\rho)$ for $r \in \mathbb{Z}^{k_1} \times \{0\}^{k_1(F_i)}$, as otherwise $\binom{\overline{A}_i}{A_i} y_i = r$ implies $Ay \neq 0$ for a global solution y. Hence, we can truncate these functions, and define

$$\tilde{\phi}_{N_i} : R \times [0 : g_1(A)], \quad \tilde{\phi}_{N_i}(r,\rho) := \phi_{N_i}\left(\begin{pmatrix} r \\ 0 \end{pmatrix}, \rho\right),$$
$$\tilde{G}_{N_i} : R \times [0 : g_1(A)], \quad \tilde{G}_{N_i}(r,\rho) := G_{N_i}\left(\begin{pmatrix} r \\ 0 \end{pmatrix}, \rho\right),$$

where $0 \in \mathbb{Z}^{k_1(F_i)}$. With these truncated functions, we can proceed as before. The construction above yields functions $\phi_{[n]}$, $G_{[n]}$ as desired.

Before we continue, we remark that though we truncate the functions, there are intermediate functions defined on vectors of dimension $(k_1 + k_1(F_i))$ (omitting ρ). If we continue with the recursion, this dimension grows further. However, in the next recursion step, we decide on one branch of F_i, and in total, the dimension on which the functions ϕ_U, G_U are defined never exceeds height(F).

Since f is given to us by a comparison oracle, observe that we can avoid calling the functions φ_U. When queuing $\varphi_U(r_1,\rho_1) + \varphi_W(r_2,\rho_2)$ for computing the function G_V with $V = U \cup W$, we can instead queue

$$\sum_{i \in U} f_i((G_U(r_1,\rho_1))_i) + \sum_{i \in W} f_i((G_W(r_2,\rho_2))_i) + \sum_{i \notin U} f_i(0).$$

The last part is since we only have a comparison oracle for f, but not for partial sums. But since f is separable convex, we can simply fix the part not concerning us. Thus, it is sufficient to implement the functions G_U.

The running time.
Every function $G_U(r,\rho)$ is defined on at most $(2g_1(A)\,||A||_\infty + 1)^{\text{height}(F)+1}$ inputs. To define the function value for each input, we have to check the values of f on all values of G_V, G_W, where $U = V \cup W$. Any value of G_U, G_V, G_W is a vector of at most $g_1(A)$ non-zero entries, stored as a list of index-value pairs. Thus, we can set up every G_U in $(g_1(A)\,||A||_\infty)^{\mathcal{O}(\text{height}(F))}$ elementary arithmetic operations. As the functions are organized in a binary tree with n leaves, we have no more than $2n$ such functions, yielding a total time of

$$2n(g_1(A)\,||A||_\infty)^{\mathcal{O}(\text{height}(F))}$$

for the initialization. Clearly, the amount of numbers stored in all G_U is bounded by the same term, and every number is an integer bounded by $\max\{g_1(A)\,||A||_\infty, \log_2(n)\}$ (the $\log_2(n)$ accounts for storing the index of the non-zero entries).

As for the update step, note that we only have to propagate the update from the affected leaf nodes $G_{\{i\}}$ to the root of $\mathcal{T}$. Since $\mathcal{T}$ is assembled from trees according to the levels of F, any leaf-root path is bounded by $\log_2(n)\,\mathrm{th}(F)$.

For the operation `Queue`, we have to queue $G_{[n]}(0,\rho)$ for each possible value of ρ, and evaluate the objective. □

We are able to give the complete statement.

Theorem 2.1. *Let an IP* (2.1)*, together with a* td*-decomposition* F *of* $G_D(A)$ *and an initial feasible solution* x_0 *be given. Define* $\zeta_{l,u} := ||u-l||_\infty$*. We can find an optimum* $x^\star$ *in time*

$$n\log_2(n)\cdot\log_2(f(x_0)-f(x^\star))\cdot\log_2(\zeta_{l,u})\cdot(g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{height}(F))}\,.$$

Moreover, the encoding size of all occurring numbers is bounded by $h(\mathrm{size}(A),\mathrm{height}(F))\cdot\mathrm{poly}(n,\mathrm{size}(l,u))$*, where* h *is any computable function.*

Proof. As observed in Lemma 2.9, we are only interested in step-lengths $\lambda = 2^k$ for $k \le \mathcal{O}(\log_2(\zeta_{l,u}))$.

For every possible value of λ, we obtain the following running time contribution. We set up a convolution-tree $\mathcal{T}^{(\lambda)}$ once, taking time

$$2n(2\,||A||_\infty\,g_1(A))^{2\,\mathrm{height}(F)+2} = n\,(||A||_\infty\,g_1(A))^{\mathcal{O}(\mathrm{height}(F))}\,.$$

Now, for each augmenting step, we have to queue each convolution tree $\mathcal{T}^{(\lambda)}$ for all possible values of ρ, and pick the value $\rho^\star$ minimizing $\phi(0,\rho)$. Then, we queue the values $G^{(\lambda)}_{[n]}(0,\rho^\star)$, and take the best value among all λ's. This can be done in time $g_1(A)+1$ per λ, plus once $g_1(A)$ for writing the output of $G^{(\lambda^\star)(0,\rho^\star)}_{[n]}$.

Finally, we have to update all trees, each taking time

$$\begin{aligned} & g_1(A)(2\,||A||_\infty\,g_1(A))^{2\,\mathrm{height}(F)+2}\,\mathrm{th}(F)\log_2(n) \\ = \quad & \log_2(n)\,(||A||_\infty\,g_1(A))^{\mathcal{O}(\mathrm{height}(F))}\,. \end{aligned}$$

By Lemma 2.10, we need at most $\mathcal{O}(n\log_2(f(x_0)-f(x^\star))$ augmenting steps to converge. Since all occurring vectors are either bounded in terms of l,u or $g_1(A)\,||A||_\infty$, the space constraints are satisfied as well. □

2.3 Reducing the objective function

Let $l, u \in (\mathbb{Z} \cup \{\pm\infty\})^n$, $B := \{x \in \mathbb{R}^n : l \leq x \leq u\}$. For a function $f : B \to \mathbb{R}$, define the value

$$f_{\max} := \max_{x,y \in P}(f(x) - f(y))$$

as the *range* of f on the domain B. For us, f will be a linear or separable convex function, and B will be domain given by the box constraints of an IP. As the upper bound on the number of iterations we need depends on the quantity $f_{\max}$, it is desirable to limit the range of f, in dependency on the domain B.

The goal of this section is to show that for every objective f (linear or separable convex), there is an objective f' such that *a*) the functions f and f' have the same comparison oracle, and *b*) the range $f'_{\max}$ is bounded in terms of n and the domain B. If both conditions are satisfied, we can estimate the running time against $f'_{\max}$ instead of $f_{\max}$, even without actually knowing f'. We say that two functions f and f' are *equivalent on* B, if for all $x, y \in B$, we have $f(x) \geq f(y) \Leftrightarrow f'(x) \geq f'(x)$. Note that every objective function $c \in \mathbb{Z}^n$ induces a partial order on the integer points $B \cap \mathbb{Z}^n$ by $x \prec_c y$ if $c^\intercal x < c^\intercal y$.

In the first subsection, we will show that for every separable convex function f, there exists an equivalent separable convex function $\hat{f}$ for which $\hat{f}_{\max}$ only depends on B and n. As the bounds we obtain are quite different, we treat the cases of a linear objective and a separable convex objective separately.

The second subsection then complements these results by constructing functions for which no equivalent objective with smaller value $\hat{f}_{\max}$ exists. Again, we will see a discrepancy between the linear and the separable convex case, showing that our results are essentially tight (up to a logarithmic factor).

2.3.1 The upper bound

In this subsection, we prove the upper bounds on linear objective functions, as well as separable convex functions. We will start with proving the linear case, allowing us to reduce the separable convex case to the linear case later. The two main propositions read as follows.

Proposition 2.16. *Let $N \in \mathbb{N}_{\geq 1}$, and $B = [a_1, b_1] \times \cdots \times [a_n, b_n] \subseteq \mathbb{R}^n$ with $a_i, b_i \in \mathbb{Z}$, $b_i - a_i \leq N$. For every vector $c \in \mathbb{Z}^n$, there exists a vector $\hat{c} \in \mathbb{Z}^n$, such that the following two conditions hold.*

i) $||\hat{c}||_1 \leq (2nN)^n$.

ii) For $y_1, y_2 \in B \cap \mathbb{Z}^n$, we have $c^\intercal y_1 < c^\intercal y_2$ if and only if $\hat{c}^\intercal y_1 < \hat{c}^\intercal y_2$.

A result similar to the following Proposition was already proven by De Loera et al. [De +10].

Proposition 2.17. *Let $N \in \mathbb{N}_{\geq 1}$, and $B = [a_1, b_1] \times \cdots \times [a_n, b_n] \subseteq \mathbb{R}^n$ with $a_i, b_i \in \mathbb{Z}$, $b_i - a_i \leq N$. For every separable convex function $f : B \to \mathbb{R}$, there exists a separable convex function $\hat{f} : B \to \mathbb{R}$ such that the following hold.*

i) For $x \in B \cap \mathbb{Z}$, we have $\hat{f}(x) \in \mathbb{Z}$.

ii) For all $x, y \in B \cap \mathbb{Z}^n$, we have $\hat{f}(x) \leq \hat{f}(y)$ if and only if $f(x) \leq f(y)$,

iii) We have $\max_{x,y \in B \cap \mathbb{Z}^n} \hat{f}(x) - \hat{f}(y) \leq n^2 N (2nN^2)^{nN-1}$.

We will give the necessary results in geometric notion. An important tool will be the following lemma of Minkowski, linking the inner and outer description of a polyhedral cone.

Lemma 2.18 ([Min68])**.** *Let $C = \{x \in \mathbb{R}^n : \ Ax \geq 0\}$ be a polyhedral cone. Let S be the set of all possible solutions to any of the systems $My = b'$, where M consists of n linearly independent rows of the matrix $\binom{A}{I}$ and $b' = \pm e_j$ for the j-th canonic unit vector e_j.*

Then there is a subset $S' \subseteq S$ such that $C = \text{cone}(S')$.

Intuitively, the lemma states that an extreme ray of a pointed cone C is defined by $n-1$ facets, and the vector generating this ray is orthogonal to all facet normals, i.e. a column of the inverse matrix, extended to full rank. However, as C is not necessarily pointed, we omit the more cumbersome general proof. One way to show the general lemma is to intersect the cone C with axis-parallel hyperplanes, decomposing it into pointed cones.

We are now able to prove the main tool.

Lemma 2.19. *Let $C = \{x : \ Ax \geq 0\} \subseteq \mathbb{R}^n$ be a polyhedral cone with constraint matrix $A \in \mathbb{Z}^{m \times n}$. Then there exists a set S' of integral vectors subject to $C = \text{cone}(S')$ and $||v||_1 \leq (2n \, ||A||_\infty)^{n-1}$ for all $v \in S'$.*

Proof. By Lemma 2.18, there is a set S such that $C = \text{cone}(S)$. For $v \in S$, let M be the submatrix of $\binom{A}{I}$ and $k \in \{1, \ldots, n\}$ an index s.t. v is the unique solution to $Mx = b'$ with $b' = \pm e_k$. Let a^T be the k-th row of M and let $M' \in \mathbb{Z}^{(n-1) \times n}$ be the matrix M without this row. As $\text{lin}(v) = \ker(M')$, we can replace v by a Graver basis element v' of M' with $(a^T v')(a^T v) > 0$ and the set $S \setminus \{v\} \cup \{v'\}$ still generates C. As Graver basis elements come in pairs $(v', -v')$, there exists such a v', and by Lemma 2.4, the length of such a v' is bounded by $(2(n-1) \, ||A||_\infty + 1)^{n-1} \leq (2n \, ||A||_\infty)^{n-1}$. Replacing every element in S yields a generating system S' as desired. □

Proof of Proposition 2.16. Let $N \in \mathbb{N}_{\geq 1}$, $c \in \mathbb{R}^n$, and B be given as in the Proposition, and consider the partial order on $B \cap \mathbb{Z}^n$ induced by c, i.e. $x_1 \succ_c x_2$ if $c^\intercal x_1 > c^\intercal x_2$. We want to define a set of all vectors y that induce the same partial order. To this end, define the set

$$T := \{(x_1, x_2) \in (B \cap \mathbb{Z}^n) \times (B \cap \mathbb{Z}^n) : c^\intercal x_1 \geq c^\intercal x_2,\ x_1 \neq x_2\},$$

and the cone

$$C := \{y \in \mathbb{R}^n : \forall\, (x_1, x_2) \in T :\ y^\intercal (x_1 - x_2) \geq 0\}.$$

We will see that all points in $\operatorname{rel.int}(C) \cap \mathbb{Z}^n$ imply the same partial order.

Since there are finitely many pairs $(x_1, x_2) \in T$, the cone C is indeed polyhedral. Moreover, $c \in \operatorname{rel.int}(C)$, as $c^\intercal(x_1 - x_2) = 0$ implies that both pairs (x_1, x_2) and (x_2, x_1) are in T. But whereas the constraints of C ensure that for any $y \in C$, the inequality $y^\intercal x_1 > y^\intercal x_2$ implies $c^\intercal x_1 > c^\intercal x_2$, the reverse is not true in general. However, if y is in the relative interior, there cannot be an inequality with $(x_1 - x_2)^\intercal y = 0$ and $(x_1 - x_2)^\intercal c > 0$.

Thus, all points in $\operatorname{rel.int}(C) \cap \mathbb{Z}^n$ induce the same partial order.

The coefficients of the constraints $(x_1 - x_2)^\intercal y \geq 0$ are bounded by N in absolute value. By Lemma 2.19, the cone C is generated by a set S of vectors v with $||v||_1 \leq (2nN)^{n-1}$. Choosing a maximum number of linearly independent generators $v_1, \ldots, v_m$, we obtain a vector $\hat{c} = \sum_{i=1}^m v_i$ in the relative interior of C with $||\hat{c}||_1 \leq m(2nN)^{n-1} \leq (2nN)^n$, finishing the proof. □

Proof of Proposition 2.17. As f is separable convex, we can write $f(x) = \sum_{i=1}^n f_i(x_i)$ for some one-dimensional convex functions f_i. Since we are interested in evaluating f on the integer points only, we assume f itself is a function only defined on the integers. We will reduce this case to the linear case by establishing a bijection between the two sets

$$\{\, f :\ B \cap \mathbb{Z}^n \to \mathbb{Z} \mid\ f \text{ is separable}\} \quad \overset{bij}{\longleftrightarrow} \quad \mathbb{Z}^{nN},$$

and then defining a cone in $\mathbb{R}^{nN}$ such that all integral points in the relative interior map to *separable* convex functions that imply the same partial ordering on the set $B \cap \mathbb{Z}^n$ as f. We will index a vector $y \in \mathbb{Z}^{nN}$ by a pair (i, k), where $i \in [n]$ and $k \in [a_i : b_i]$. A function $g = \sum_{i=1}^n g_i$ then maps to a vector y given by the relation $y_{i,k} = g_i(k)$. Note that $g(u) = \sum_i y_{i,u_i}$ for $u \in B \cap \mathbb{Z}^n$.

With this notation, we define $C \subseteq \mathbb{R}^{nN}$ with the following two types of constraints. First

take for each pair $u, v \in B \cap \mathbb{Z}^n$, $x \neq y$, the constraint

$$\sum_{i=1}^{n} y_{i,v_i} - \sum_{i=1}^{n} y_{i,u_i} \geq 0 \qquad \text{if } f(u) < f(v), \tag{2.7a}$$

$$\sum_{i=1}^{n} y_{i,u_i} - \sum_{i=1}^{n} y_{i,v_i} \geq 0 \qquad \text{if } f(u) > f(v), \tag{2.7b}$$

$$\sum_{i=1}^{n} y_{i,u_i} - \sum_{i=1}^{n} y_{i,v_i} = 0 \qquad \text{if } f(u) = f(v). \tag{2.7c}$$

These constraints will ensure that all points in the interior induce the same partial order as the vector y_f obtained from f.

But we also have to ensure that the function corresponding to a vector $y \in C$ is actually convex. To this end, we introduce for every $i = 1, \ldots, n$ the constraints

$$y_{i,k-1} + y_{i,k+1} - 2y_{i,k} \geq 0, \qquad \forall k \in [a_i + 1 : b_i - 1].$$

Translating to the function g according to y, this is $g_i(k) \leq 1/2(g_i(k-1)+g_i(k+1))$ for all i, k in range. These "local" convexity constraints imply already general convexity. To see this, let $a, b \in [a_i : b_i]$, with $a < b-1$. By repeatedly applying either $y_{i,a} \geq 2y_{i,a+1} - y_{i,a+2}$, or $y_{i,b} \geq 2y_{i,b-1} - y_{i,b-2}$, we obtain, for any k, k' in range,

$$y_{i,a} \geq (k+1)y_{i,a+k} - ky_{i,a+k+1}, \tag{2.8}$$

$$y_{i,b} \geq (k'+1)y_{i,b-k'} - k'y_{i,b-k'-1}. \tag{2.9}$$

Choosing the relation $k' = b - a - k - 1$, adding $b - a - k$ times the first inequality and k times the second inequality culminates in

$$(b - a - k)y_{i,a} + ky_{i,b} \geq (b - a)y_{i,a+k}.$$

Now that we have established all inequalities, let A be the matrix comprising all listed inequalities. We define $C := \{y \in \mathbb{R}^{nN} : Ay \geq 0\}$ as a polyhedral cone with $||A||_\infty \leq 2$. By Lemma 2.19, we have $C = \text{cone}(S)$ for some set S of integral vectors v with $||v||_1 \leq (4nN)^{nN-1}$. We pick again a vector y in the relative interior, and pick the corresponding separable convex function. The claim follows. □

Remark 2.20. *Observe that in the separable convex case, the vector y_f obtained from f is not necessarily in the interior of the cone C. For instance, if f is chosen to be just a linear function, the convexity constraints are not fulfilled with strict inequality, and we might be able to find a smaller separable convex function than provided by the theorem. If we want to avoid discarding linear functions, we can strengthen the Inequalities* (2.7) *by replacing the right-hand side by* 1*. This shifts the cone away from* 0*, but all contained points yield feasible functions.*

Moreover, observe that we did not require $f(z) \in \mathbb{Z}$ for $z \in \mathbb{Z}$. This allows us to analyze the convergence for any separable convex function in our algorithms in terms of $||u - l||_\infty$ and n.

2.3.2 The lower bound

In this subsection, we show that the bounds derived in Propositions 2.16 and 2.17 are essentially tight.

Linear objective functions.

Let us start with the linear objective function. Recall that an objective function $c^\intercal x$ induces a partial order on the set $[0, N]^n \cap \mathbb{Z}^n$ by $x \prec y$ if $c^\intercal x < c^\intercal y$. The idea is to construct a sequence of points $x_1, \ldots, x_m \in [0, N]^n \cap \mathbb{Z}^n$, equipped with a partial order $x_1 \prec \cdots \prec x_m$. We then show that every linear objective whose induced partial order agrees with the chosen sequence must have large infinity norm. Showing that an agreeing linear function exists completes the argument.

Lemma 2.21. *Let $n, N \in \mathbb{N}_{\geq 1}$. There exists a linear function $c \in \mathbb{Z}^n$ such that for every $c' \in \mathbb{Z}^n$ subject to*

$$c^\intercal x \geq c^\intercal y \quad \Leftrightarrow \quad c'^\intercal x \geq c'^\intercal y \qquad \forall x, y \in [0, N]^n \cap \mathbb{Z}^n,$$

we have $||c'||_\infty \geq N^{n-1}$.

Proof. The sequence is defined as follows. We start with $0 \in [0, N]^n \cap \mathbb{Z}^n$, and then take all the vectors $e_i, 2e_i, \ldots, Ne_i$ dimension-wise, i.e. take first $e_1, \ldots, Ne_1$, and end with the vectors $e_n, 2e_n, \ldots, Ne_n$. Formally, we have $x_0 = 0$, and $x_{(i-1)N+k} = ke_i$ for $(i, k) \in \{1, \ldots, n\} \times \{1, \ldots, N\}$.

Now let $c \in \mathbb{Z}^n$ be any linear function whose induced partial order coincides with $x_1 \prec \cdots \prec x_{nN}$. As $c^\intercal x_{j+1} \geq c^\intercal x_j + 1$, this leads to the inequalities

$$\begin{aligned} c_1 = c^\intercal x_1 &\geq 1 \\ c_2 = c^\intercal x_{N+1} &\geq c^\intercal x_N + 1 = Nc_1 + 1 \\ &\vdots \\ c_n = c^\intercal x_{(n-1)N+1} &\geq c^\intercal x_{(n-1)N} + 1 = Nc_{n-1} + 1, \end{aligned}$$

culminating to

$$c_n \geq Nc_{n-1} + 1 \geq \cdots \geq N^{n-1}c_1 + \sum_{i=0}^{n-2} N^i \geq N^{n-1}.$$

It is also easy to see that if we choose equality in every case, i.e. $c_1 = 1$ and $c_{i+1} = Nc_i + 1$ for $i = 1, \dots, n-1$, we indeed obtain a linear objective inducing the chosen partial order; the relations $ke_i \prec (k+1)e_i$ are fulfilled for any linear objective $c \in \mathbb{Z}^n_{>0}$, and the remaining $n-1$ relations $Ne_i \prec e_{i+1}$ are fulfilled by choice of the values c_i. □

Separable convex functions.

Similar to the case of a linear objective, we will define a sequence $(x_j)_j \subseteq [0, N]^n \cap \mathbb{Z}^n$, and force a partial ordering on the sequence that should be satisfied by a separable convex function.

Lemma 2.22. *Let $n, N \in \mathbb{N}_{\geq 1}$. There exists a separable convex function $f : \mathbb{Z}^n \to \mathbb{Z}$ such that for every separable function $f' : \mathbb{Z}^n \to \mathbb{Z}$ subject to*

$$f(x) \geq f(y) \quad \Leftrightarrow \quad f'(x) \geq f'(y) \qquad \forall x, y \in [0, N]^n \cap \mathbb{Z}^n,$$

we have

$$\max_{z \in [0,N]^n \cap \mathbb{Z}^n} f'(z) - \min_{z \in [0,N]^n \cap \mathbb{Z}^n} f'(z) \geq n^{N-1} 2^{n-1}.$$

Proof. We will initialize the sequence with $x_i = 0 \in \mathbb{Z}^n$ for $i \leq 0$. The negative indices will allow for a simpler recursive definition later. Then we define

$$x_{(k-1)n+i} = ke_i \qquad \forall\, (i, k) \in \{1, \dots, n\} \times \{1, \dots, N\},$$

i.e. we cycle through the canonic unit vectors $e_1, \dots, e_n$, then cycle through $2e_1, \dots, 2e_n$, and so forth. For the auxiliary sequence, we define

$$y_j = x_j + x_{j-1}, \text{ where } j = 1, \dots, nN.$$

It is crucial that for any separable convex function f, the function value on any y_j is solely determined by the function values on x_j and x_{j-1}. Moreover, we do not have $y_j \in \{x_\ell\}_\ell$ (apart from $x_1 = y_1$). Hence, we can enforce the ordering $f(x_{j+1}) = f(y_j) = f(x_j) + f(x_{j-1})$.

We now want to construct a separable convex function $f = \sum_{i=1}^n f_i$ with the relations $f(x_{j+1}) = f(y_j) = f(x_j) + f(x_{j-1})$ for $j \geq 1$. Without loss of generality, we may set $f_i(0) = 0$ for $i = 1, \dots, n$, as adding a constant to the functions f_i does not change $f_{\max}$. For $j = (k-1)n + i$, this implies $f(x_j) = f(ke_i) = f_i(k)$, i.e. when fixing a function value $f(x_j)$, we actually only fix the value for the corresponding f_i. As every $ke_i \in \{x_j\}_j$, this means that fixing $f(x_1)$ completely determines any separable function f which induces the chosen relations between the two sequences $\{x_j\}_j$ and $\{y_j\}_j$. Choosing $f(x_1) = 1$, the smallest possible value enforcing $f(x_0) < f(x_1)$, implies that $f(x_j) = f(y_{j-1}) = f(x_{j-1}) + f(x_{j-2}) = F_j$ is the j-th Fibonacci number. Hence,

$f(x_{nN}) \approx \varphi^{nN}$, where $\varphi = \frac{1+\sqrt{5}}{2}$ is the golden ratio.

It remains to show that our function is separable convex. The separability follows by construction, as we only define values on multiples of the canonic unit vectors. Convexity follows from considering f as a piecewise linear function with break points $\{1, \dots, N-1\}$ and growing slope, i.e. verifying $F_{j+n} - F_j > F_j - F_{j-n}$. □

2.4 Reducing the box constraints

We saw that the number of iterations we need depends on $f(x_0) - f(x^\star)$, where x_0 and $x^\star$ are initial and optimal feasible solutions. For separable convex functions, this was usually estimated by $f_{\max}$, where $f_{\max} := \max_{x,y \text{ feasible}}(|f(x) - f(y)|)$ was supposed to be part of the input. If $f(x) = c^\intercal x$ is a linear objective, this value can be limited by $||c||_1$ and $||x^\star - x_0||_\infty$. Also in the case of a separable convex objective, Proposition 2.17 involves the term $||u - l||_\infty$, even appearing in the exponent! It is therefore desirable to obtain a better grasp on this value, and replace the bounds l, u with artificial, smaller bounds in certain scenarios.

In this section, we will give an approach to restrict the dependency on $||u - l||_\infty$. Surprisingly, this can be done in terms of the already developed bound on the elements in $\mathcal{G}(A)$, and the number of non-integral entries of $x^\star$.

The general idea is that if we find an optimum solution $x^\star$ of the continuous relaxation, then there is an optimum solution $z^\star$ of the IP near by. Hence, we can replace the given bounds l, u that might be rather large with smaller box constraints, centered around $x^\star$. Results of this kind are usually called *proximity results*, cf. [Sch86, Sec. 17.2]. We provide a proximity result specific to our setting, obtaining a bound linear in n in Subsection 2.4.1.

When it comes to IPs without a linear objective, the optimal point $x^\star$ might be irrational. It is straightforward to see that we can also apply the proximity results to an approximation $\tilde{x}$ of the optimum $x^\star$. Here, it is important that $\tilde{x}$ is an approximation to $x^\star$ in terms of metric distance, not in terms of the objective function value. Under reasonable assumptions to the objective, Hochbaum & Shanthikumar [HS90] also show that being close in terms of metric distance implies that the objective value cannot be too far from optimal as well.

Definition 2.23 ([HS90])**.** *Let an IP* (2.1) *be given, and let* $\epsilon > 0$*. A fractional point* $\tilde{x}$ *is an* ϵ-accurate solution to the continuous relaxation of (2.1)*, if there exists an optimum solution* $x^\star$ *of the relaxation with* $||\tilde{x} - x^\star||_\infty \leq \epsilon$*.*

Using Chubanov's algorithm [Chu16, Theorem 12], we also have an algorithm that gives us an ϵ-accurate solution for non-linear objectives. The running time is weakly

polynomial, and its dependency on n is roughly $n^4 \log_2(n) + Tn \log_2(n)$, where T is the time needed to solve an auxiliary LP.

However, we can also avoid solving the continuous relaxation altogether, as will be shown in Subsection 2.4.2.

2.4.1 The proximity result

We start by showing a proximity result for an IP and its continuous relaxation. For the infinity-norm and n-folds, this result was already shown by Hemmecke, Köppe and Weismantel [HKW14]. Our result can be seen as a strengthened generalization of theirs, while the proofs follow the same essential insights.

Proposition 2.24. *Let an IP* (2.1) *be given, and let $x^\star$ be an optimum solution of the continuous relaxation, and $z^\star$ be an optimum solution to the IP.*

i) There exists an optimum solution $\tilde{z}$ to the IP such that

$$||x^\star - \tilde{z}||_1 \leq |S| g_1(A), \text{ where } S := \{i \in [n] : x_i^\star \notin \mathbb{Z}\}.$$

ii) There exists an optimum solution $\tilde{x}$ to the continuous relaxation such that

$$||\tilde{x} - z^\star||_1 \leq |S| g_1(A), \text{ where } S := \{i \in [n] : \tilde{x}_i \notin \mathbb{Z}\}.$$

Moreover, these results hold even if the box-constraints $l \leq x \leq u$ are not integral.

In order to streamline the proof of this proposition, we outsource the following claim beforehand. Intuitively, it states that if a set of fractional numbers sum up to an integer, we can round all numbers suitably so that their sum does not change. We will later consider the sum of fractional vectors, and apply the lemma component-wise.

Lemma 2.25. *Let $a_1, \ldots, a_r \in \mathbb{R}$ such that $\sum_{i=1}^r a_i \in \mathbb{Z}$. There is a map $\varphi(\cdot)$ on $\{a_1, \ldots, a_r\}$ such that, for every $k \in [r]$, we have $\varphi(a_k) \in \{\lfloor a_k \rfloor, \lceil a_k \rceil\}$, and $\sum_{i=1}^r a_i = \sum_{i=1}^r \varphi(a_i)$.*

The map φ can be found using $\mathcal{O}(r)$ arithmetic operations, plus $\mathcal{O}(r)$ rounding operations.

Proof. We will define $\varphi(\cdot)$ iteratively, starting with $\varphi(a_i) = \lceil a_i \rceil$. Now assume we assigned values to $\varphi(a_1), \ldots, \varphi(a_k)$. We define

$$\varphi(a_{k+1}) = \begin{cases} \lceil a_i \rceil & \text{if } \sum_{i=1}^k \varphi(a_i) < \sum_{i=1}^k a_i, \\ \lfloor a_i \rfloor & \text{if } \sum_{i=1}^k \varphi(a_i) \geq \sum_{i=1}^k a_i. \end{cases}$$

Since $\max\{\lceil a_i \rceil - a_i, a_i - \lfloor a_i \rfloor\} < 1$, the constraint $|\sum_{i=1}^{k} \varphi a_i - \sum_{i=1}^{k} a_i| < 1$ is maintained for every k. The claim follows from $\sum_{i=1}^{m} a_i \in \mathbb{Z}$.

Storing the partial sums $\sum_{j=1}^{i-1} a_j$ and $\sum_{j=1}^{i-1} \varphi(a_j)$, we can decide the value of each $\varphi(a_i)$ with two arithmetic operations and one comparison only. For rational values, each rounding can be implemented with $\mathcal{O}(1)$ arithmetic operations as well. □

Proof of Proposition 2.24. For the last remark of the Proposition, it suffices to observe that we will not use $l, u \in \mathbb{Z}^n$ throughout the proof.

For $x^\star$, $z^\star$ as in the Proposition, let $\tilde{z}$ be an optimum solution to the IP closest to $x^\star$, and $\tilde{x}$ be an optimum solution to the LP closest to $z^\star$. As both cases work similar, let $(x, z) \in \{(x^\star, \tilde{z}), (\tilde{x}, z^\star)\}$ be one of the pairs we want to relate.

Let $\lfloor x \rceil_z \in \mathbb{Z}^n$ denote the vector x rounded towards z, i.e. $(\lfloor x \rceil_z)_i := \lfloor x_i \rfloor$ if $z_i \leq x_i$, and $(\lfloor x \rceil_z)_i := \lceil x_i \rceil$ otherwise. With $\{x\} := x - \lfloor x \rceil_z$ we denote the fractional rest of x. Let $S := \{i \in [n] : x_i \notin \mathbb{Z}\}$, and observe that

$$A\{x\} = \sum_{i \in S} \{x\}_i A_i \in \mathbb{Z}^m,$$

where A_i denotes the i-th column of A. We can now apply Lemma 2.25 component-wise to this sequence of vectors $(\{x\})_i A_i$, and obtain vectors $\hat{A}_i \in \mathbb{Z}^m$ such that $A\{x\} = \sum_{i \in S} \hat{A}_i$. Let $\hat{A}$ denote the matrix with columns $\hat{A}_i$, $i \in S$.

We claim that $g_1(A, \hat{A}) = g_1(A)$. Indeed, the dual graphs have the same set of vertices, and the columns $\hat{A}_i$ do not introduce new edges. Moreover, as $|(\{x\})_i| < 1$, we have that $|\hat{A}_i| \leq |A_i|$, where the absolute value is taken component-wise. Now we can write

$$0 = A(x - z) = A(\lfloor x \rceil_z + \{x\} - z) = (A, \hat{A})y, \qquad y := \begin{pmatrix} \lfloor x \rceil_z - z \\ \mathbf{1} \end{pmatrix} \in \mathbb{Z}^{n+|S|}.$$

This allows us to decompose the vector $y = \sum_{i=1}^{K} g_i$ into K Graver basis elements $g_i \in \mathcal{G}(A, \hat{A})$.

If $K \leq |S|$, we have $||x - z||_1 \leq ||y||_1 \leq K g_1(A, \hat{A}) = |S| g_1(A)$.

If not, there is an index k such that $g_k \in \mathcal{G}(A) \times \{0\}^{|S|}$. Since x and z are both feasible to the continuous relaxation, so are $z + g_k$ and $x - g_k$. Setting $y_1 = g_k$ and $y_2 = x - (z + g_k)$, we have $x = z + y_1 + y_2$, and supperadditivity of separable convex functions yields

$$\begin{aligned} & f(z + y_1 + y_2) - f(z) \geq f(z + y_1) - f(z) + f(z + y_2) - f(z) \\ \Leftrightarrow \quad & f(z + y_1 + y_2) - f(z + y_2) \geq f(z + y_1) - f(z) \\ \Leftrightarrow \quad & f(x) - f(x - g_k) \geq f(z + g_k) - f(z). \end{aligned}$$

Since $f(x)$ and $f(z)$ are optimal (w.r.t. the continuous relaxation, the IP respectively), we must have

$$0 \geq f(x) - f(x - g_k) \geq f(z + g_k) - f(z) \geq 0,$$

implying $f(x - g_k) = f(x)$ and $f(z) = f(z + g_k)$. But this is a contradiction, since either x was chosen closest to z and $x - g_k$ is closer to z, or the other way around. □

2.4.2 Reducing the bounds by iterative scaling

If we want to avoid solving the continuous relaxation of an IP (2.1), but still have large box constraints $||u - l||_\infty$, it can already be helpful to reduce the dependency on l, u of the algorithm. For instance, we might expect that $f_{\max} \geq f(x_0) - f(x^\star)$ is at least linear in $||u - l||)_\infty$ for a separable convex function.

The idea is to first optimize with rather large step lengths λ only i.e. updating $x \leftarrow x + \lambda y$ with $\lambda \geq 2^k$. This practically replaces l, u by smaller bounds $\frac{l}{\lambda}, \frac{u}{\lambda}$, and we can apply the results of Section 2.3 for limiting the objective function values. Iteratively refining the solution leads to an optimal solution eventually. The motivation for these results was taken from the techniques in [HS90].

We lay out the approach more specifically. Assume for now that 0 is a feasible solution, i.e. we are concerned with the following IP.

$$\begin{aligned} \min \quad & f(x) \\ s.t. \quad & Ax = 0 \\ & l \leq x \leq u \\ & x \in \mathbb{Z}^n. \end{aligned} \tag{2.10}$$

Instead of looking for a solution in the lattice $x \in \mathbb{Z}^n$, we first look for a solution in the scaled lattice $z \in s\mathbb{Z}^n$, for some $s \in \mathbb{Z}_{\geq 1}$.

$$\begin{aligned} \min \quad & f(x) \\ s.t. \quad & Ax = 0 \\ & l \leq x \leq u \\ & x \in s\mathbb{Z}^n. \end{aligned} \tag{2.11}$$

Observe that both systems have the same continuous relaxation. Hence, an optimal solution $x^\star$ to the continuous relaxation of (2.10) is also an optimal solution to the continuous relaxation of (2.11). With this, we can relate an optimal solutions $z^\star$ of (2.10) with an optimal solution $\hat{z}$ of (2.11).

Lemma 2.26. *Let an IP* (2.10) *be given, and* $s \in \mathbb{Z}_{\geq 1}$. *For every optimal solution* $z^\star$

of (2.10)*, there exists an optimum solution* $\hat{z}$ *of the scaled IP* (2.11) *such that*

$$||z^\star - \hat{z}||_1 \leq (s+1)ng_1(A).$$

Vice versa, to every $\hat{z}$*, there exists a* $z^\star$ *within the same distance.*

Proof. Let $z^\star$ be an optimum solution of (2.10). By proximity, there exists an optimum solution $x^\star$ to the continuous relaxation with

$$||z^\star - x^\star||_1 \leq ng_1(A). \tag{2.12}$$

As the continuous relaxations coincide, $x^\star$ is also an optimum to the continuous relaxation of (2.11). Substituting $x' := \frac{1}{s}x$, we obtain an objective-value preserving bijection between the solutions of (2.11) and the solutions of the *s-scaled IP*

$$\begin{aligned} \min \quad & f(sx') \\ s.t. \quad & Ax' = 0 \\ & \tfrac{l}{s} \leq x' \leq \tfrac{u}{s} \\ & x' \in \mathbb{Z}^n. \end{aligned} \tag{2.13}$$

In particular, $x' := \frac{1}{s}x^\star$ is an optimum solution to the continuous relaxation of (2.13). We remark that the box constraints of (2.13) might not be integral. However, this is irrelevant for the proximity result, and if we want to work with the IP (2.13), we can round the new variable bounds towards zero for any practical purposes.

Again by proximity, the instance (2.13) has an optimal solution z' with

$$||x' - z'||_1 \leq ng_1(A) \quad \Leftrightarrow \quad ||sx' - sz'||_1 \leq sng_1(A) \tag{2.14}$$

Substituting back, $\hat{z} := sz'$ is an optimal solution to (2.11). Plugging this into (2.14) and combining with (2.12), we obtain

$$||z^\star - \hat{z}||_1 \leq ||z^\star - x^\star||_1 + ||x^\star - \hat{z}||_1 \leq (s+1)ng_1(A).$$

The other direction, starting with $\hat{z}$, works similar. □

As a 2^k-scaled instance is also a 2-scaled instance of a 2^{k-1}-scaled instance, this gives the following idea for an algorithm to solve a general IP (2.1).

(1) Find an initial feasible solution x_{init}, and recenter the instance to (2.10). Let $k = \log_2(\max(||u||_\infty, ||l||_\infty)) + 1$, and set $x_k := 0$ (i.e. the shift of x_{init}). As it is the only feasible solution, x_k is an optimal solution of the 2^k-scaled instance.

(2) If $k > 0$, replace the box constraints of the 2^{k-1}-scaled instance with the proximity bounds $||x_k - x||_\infty \leq 3ng_1(A)$. Solve this instance with initial solution x_k to optimality, and let x_{k-1} denote the found optimum. Update $k \leftarrow k-1$ and repeat.

(3) Output x_0.

Analyzing the running time, we obtain the following proposition.

Proposition 2.27 (cf. [Eis+19])**.** *Given an IP* (2.1) *with feasible solution* x_0 *and finite bounds* l, u*, let* $\zeta_{l,u} := ||u - l||_\infty$.

i) If $f(x) = c^\intercal x$ *is a linear objective, we can find an optimum solution in time*

$$\mathcal{O}(n^2(\log_2(n))^3) \cdot \log_2(\zeta_{l,u}) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\text{height}(F))}.$$

ii) If $f(x)$ *is a separable convex objective, we can find an optimum solution in time*

$$\mathcal{O}(n^3(\log_2(n))^3) \cdot \log_2(\zeta_{l,u}) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\text{height}(F))}.$$

Proof. W.l.o.g. assume that $l \leq x_0 = 0 \leq u$ is a feasible solution. This can be achieved with a substitution $x' = x - x_0$, leaving $\zeta_{l,u}$ unaffected. For $k = \log_2(\zeta_{l,u}) + 1$, the only feasible (and therefore optimal) solution in the 2^k-scaled instance is $x_k := 0$. Observe that this 2^k-scaled instance is also a 2-scaled instance of the 2^{k-1}-scaled instance (of the original).

Repeat the following while $k > 0$. Apply the proximity result, Proposition 2.26, to obtain new box constraints $l_{k-1} \leq x \leq u_{k-1}$ for the 2^{k-1}-scaled instance, centered around x_k. Using x_k as a feasible solution for this modified instance, find an optimal solution x_{k-1} in time

$$\mathcal{O}(n\log_2(n)) \cdot \log_2(f(x_k) - f(x_{k-1})) \cdot \log_2(||u_{k-1} - l_{k-1}||_\infty) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\text{height}(F))}$$

by Theorem 2.1, where $||u_{k-1} - l_{k-1}||_\infty \leq 6ng_1(A)$ by the proximity result. Using the objective reductions of Proposition 2.16 for linear objectives (Case *i*)), and Proposition 2.17 for separable convex objectives (Case *ii*)), we can also estimate

$$\begin{array}{rrcl} i) & \log_2(c^\intercal(x_k - x_{k-1})) & \leq & \mathcal{O}(n\log_2(ng_1(A))), \\ ii) & \log_2(f(x_k) - f(x_{k-1}) & \leq & \mathcal{O}(n^2\log_2(n)g_1(A)). \end{array}$$

We finish each iteration by updating $k \leftarrow k-1$.

With the initial choice of k, the claimed running times follows. If necessary, we resubstitute $x^\star = \hat{x} + x_0$ the found solution $\hat{x}$. □

We close this subsection with two remarks. First, applying Proposition 2.17 in order to reduce the objective in each step is optional. This is, if we already have a good grasp on $f_{\max}$, we might analyze the running time with the initial term, reducing the dependency on n.

Second, once we reached an optimal solution in the initial IP (2.1), we could continue scaling, and look for a solution in the superlattice $\frac{1}{2}\mathbb{Z}^n$. Continuing further, this can be used to find a 2^{-k}-accurate solution for the continuous relaxation of (2.1). Before we cite this result from [Eis+19], we introduce the necessary notation. For a function g, we denote

$$g_{\max}^{[pl,pu]} := \max_{pl \leq x,y \leq pu} |g(x) - g(y)|$$

as the largest discrepancy g can have in the box $pl \leq x \leq pu$. Moreover, $g_\infty(A) := \max_{g \in \mathcal{G}(A)} ||g||_\infty$.

Proposition 2.28 ([Eis+19, Cor. 65])**.** *Let an IP* (2.1) *be given, define* $p := \frac{1}{\varepsilon} n g_\infty(A)$*,* $g(z) := f(\frac{z}{p})$*, and assume that* f *satisfies* $\forall z \in \mathbb{Z}^n : g(z) \in \mathbb{Z}$*. If we can solve* (2.1) *in time* $T(A, ||u-l||_\infty, ||b||_\infty, f_{\max})$*, then an* ε*-accurate solution of the continuous relaxation of* (2.1) *can be found in time* $T(A, p\,||u-l||_\infty, p\,||b||_\infty, g_{\max}^{[pl,pu]})$*.*

Remark 2.29. *We can circumvent the integrality constraint of* f *by applying Proposition 2.17 for finding an integer-valued equivalent function for* $g(x) := f(\frac{1}{2^k}x)$*. While this is circumventing dependency on* f *at all, this approach is increasing the dependency on* n*.*

2.5 Feasibility and finiteness

So far, we always assumed that we are given an initial feasible solution to a *bounded* IP (2.1). This section is devoted to justifying these assumptions, starting with feasibility in Subsection 2.5.1. We will see that finding an initial feasible solution is basically as easy as optimizing an existing feasible solution.

If we do not have finite bounds, the first problem we have to solve is deciding boundedness. As we are working with iterative augmenting, it is crucial to check this before we start optimizing, and is not discovered during the algorithm. In the case of a linear objective, this can easily be done by solving one augmentation IP, whereas the problem is undecidable for separable convex functions in general, see Subsection 2.5.2.

These results justify the assumption that we are only given bounded instances. Subsection 2.5.3 is devoted to discussing possibly infinite bounds $l, u \in (\mathbb{Z} \cup \{\pm\infty\})^n$. We will see that we can still find an optimum solution. For linear objectives, we can show artificial bounds on an optimum solution $x^\star$ in terms of b and the finite entries of l, u. For separable convex functions, we guess a bound on the distance between our starting solution x_0, and an optimum solution $\overline{x}$ closest. This replaces every factor $||u-l||_\infty$ in the running time by $||x_0 - \overline{x}||_\infty$, plus an additional factor of $||x_0 - \overline{x}||_\infty$ for guessing.

2.5.1 Deciding feasibility and finding an initial feasible solution

So far, we were always assuming that an initial feasible solution is given to us. This might be the case if the IP at hand is modeling a problem for which an initial solution can easily be observed, e.g. if the IP is a model of a scheduling problem. Alternatively, we might have an approximation algorithm that provides us with an initial feasible solution for the modeled problem at hand. If none of these cases apply, we need a more general approach to find an initial feasible solution, which is done in this section.

Generally, there are two ways of obtaining an initial feasible solution. One way is to start with a vector x such that $l \leq x \leq u$ is satisfied, but $Ax = b$ might be violated. By introducing slack variables, we can deploy the augmenting procedure to obtain $\hat{x}$ out of x such that $A\hat{x} = b$ holds, while $l \leq \hat{x} \leq u$ remains satisfied. We choose this approach. Alternatively, we can compute the Hermite normal form of A in order to obtain a vector $\tilde{x}$ with $A\tilde{x} = b$, but possibly violating the box constraints. We then introduce a new separable convex objective $\tilde{f}(x) := \sum_i \operatorname{dist}(x_i, [l_i, u_i])$ that we want to minimize first, using $\tilde{x}$ as initial feasible solution.

The following proposition will also discuss infinite bounds already, as we will need the results for the following chapters. Jansen, Lassota and Rohwedder [JLR19] came up with an approach to limit the size of an initial feasible solution x_0 for N-fold IPs, and the technique here can be considered a generalization.

Proposition 2.30. *Let an IP* (2.1) *be given, together with a* td*-decomposition F of $G_D(A)$. Define $\zeta_{l,u} := ||u - l||_\infty$. We can either decide that the IP is infeasible, or find an initial feasible solution x_0 in time*

$$\mathcal{O}(n \log_2(n)) \cdot \log_2(n\zeta_{l,u}) \cdot \log_2(\zeta_{l,u}) \cdot (g_1(A)\, ||A||_\infty)^{\mathcal{O}(\text{height}(F))}.$$

Proof. We proceed as follows. Let $\overline{x}$ be a vector with $l \leq x \leq u$. We recenter the initial instance by substituting $x' = x - \overline{x}$, obtaining

$$\begin{aligned} \max f'(x') &:= f(\overline{x} + x'), \\ Ax' = b' &:= b - A\overline{x}, \\ x' \geq l' &:= l - \overline{x}, \\ x' \leq u' &:= u - \overline{x}, \end{aligned} \tag{2.15}$$

and assume $b' \geq 0$, which can be achieved by multiplying single rows with -1. Observe that $||b'||_1 \leq n\zeta_{l,u}\, ||A||_\infty$. Moreover, $x' = 0 \in \mathbb{Z}^n$ satisfies the new box constraints.

Therefore, the vector $\binom{0}{b'}$ is a feasible solution to the following *auxiliary IP*:

$$\begin{aligned} \min \quad & \mathbf{1}^\intercal y \\ s.t. \quad & (A, \mathbb{1}) \begin{pmatrix} z \\ y \end{pmatrix} = b' \\ & l' \leq z \leq u' \\ & 0 \leq y \leq b' \\ & y, z \in \mathbb{Z}^n \end{aligned} \tag{2.16}$$

As we only attached the identity to A, we have $G_D(A) = G_D(A, \mathbb{1})$, and all parameters remain unchanged. Clearly, the initial solution $\binom{0}{b'}$ has objective value $||b'||_1$ and the optimal value is at least 0, with equality if and only if the initial IP (2.1) has a feasible solution. By Theorem 2.1 and assuming $m \leq n$, we can find an optimum solution $\binom{z^\star}{y^\star}$ to this auxiliary instance in time

$$\mathcal{O}(n \log_2(n)) \cdot \log_2(||b'||_1) \cdot \log_2(\zeta_{l,u}) \cdot (g_1(A)\, ||A||_\infty)^{\mathcal{O}(\text{height}(F))}.$$

Plugging in the estimate $\log_2(||b'||_1) \leq \log_2(n\zeta_{l,u}) \log_2(||A||_\infty)$, and hiding the term involving A in the Laundau notation in the exponent, we obtain the claimed running time. If $y^\star \neq 0$, we declare the initial IP infeasible. Otherwise, we output $x_0 := z^\star + \overline{x}$. □

2.5.2 Deciding boundedness

In this section, we will see that for the case where $f(x)$ is a linear function, we can decided whether there exists a finite optimum fairly efficient.

However, if f is an arbitrary separable convex function, we cannot decide whether there exists a finite optimum in general.

The case of a linear objective.

We basically have two approaches at our disposal. If we want to solve the LP relaxation of the given IP anyways, the following lemma reduces the question whether the IP is bounded to deciding feasibility.

Lemma 2.31. *Let a rational IP* (2.1) *with linear objective be given, and assume its LP relaxation is feasible and unbounded. Then either the IP is feasible and unbounded, or it is infeasible.*

Proof. If the IP is infeasible, there is nothing to show, hence let x be a feasible integer point. As all input data is rational, the recession cone of the LP relaxation contains a rational vector v such that $x + \lambda v$ is feasible and $c^\intercal(x + \lambda v) < c^\intercal x$ for any $\lambda \in \mathbb{R}_{\geq 0}$. But

then there exists a positive scaling v' of v such that $x + \alpha v'$ is integral, feasible, and has a better objective value for any $\alpha \in \mathbb{N}$. Thus, the IP is unbounded as well. □

If we want to avoid solving the LP relaxation, we can first decide feasibility, and then make a single augmenting step.

Lemma 2.32. *Let an IP instance* (2.1) *be given, where $f(x) = c^\intercal x$ is a linear objective, together with an initial feasible solution. Then we can decide whether the optimum is finite by solving a single augmentation IP.*

Proof. Let x be the initial feasible solution. If the optimum is infinite, the recession cone of the LP relaxation has dimension at least 1. As all the input is rational, so is the recession cone and by scaling, there exists an integral vector v such that $x + \lambda v$ is feasible for any $\lambda \in \mathbb{R}_{\geq 0}$. This implies that whenever $v_i \neq 0$, we either have $l_i = -\infty$ if $v_i < 0$, or $u_i = +\infty$ if $v_i > 0$. Decomposing v into the sum of conformal Graver basis elements, we see that the augmentation IP specified by the following box constraints has a solution y with $c^\intercal y < 0$:

$$l'_i = \begin{cases} 0 & l_i \in \mathbb{N} \\ -\infty & l_i = -\infty \end{cases}, \qquad \text{and} \qquad u'_i = \begin{cases} 0 & u_i \in \mathbb{N} \\ \infty & u_i = \infty \end{cases}.$$

Vice versa, if the augmentation IP has a solution y with $c^\intercal y < 0$, it is easy to see that $x + \alpha y$ is feasible integral for any $\alpha \in \mathbb{N}$, therefore the initial IP is unbounded. □

The case of a separable convex objective.

For the case of a separable convex objective, it cannot be decided whether a given IP is bounded. The following lemma justifies that we assume to be given a bounded IP in the case of a separable convex function.

Lemma 2.33 ([Eis+19], cf. also [Onn10], Section 1.3.3)**.** *The problem of deciding whether a given IP* (2.1) *is bounded is undecidable, even in one dimension.*

Proof. Let $l = -\infty$, $u = \infty$, and $0 \cdot x = 0$, i.e. the set of feasible points is simply $\mathbb{Z}$. Assume there exists a finite algorithm for deciding whether a given IP is bounded. For the objective, whenever queued, assume we get $f(x) = x$. If there is a finite algorithm, it can only queue a finite number of points, hence let p be the minimum value for which f was queued. If the algorithm outputs that the IP was unbounded, we claim that the objective was actually $f(x) = |x - p| + p$. If the algorithm outputs that the IP was bounded, we claim that the objective was chosen to be $f(x) = x$. In both cases, the output is wrong. □

2.5.3 Handling infinite bounds

We first show that in the case of infinite bounds, we can still guarantee the existence of a reasonably small feasible solution, as well as a bound on an optimal solution in the case of linear objectives. Having these results, we can simply replace all infinite bounds and run the algorithm for finite bounds.

Lemma 2.34. *Let a feasible and bounded IP* (2.1) *with possibly infinite bounds* $l, u \in (\mathbb{Z} \cup \{\pm\infty\})^n$ *be given. Define* $\zeta_{l,u} := \max\{|l_i| : i \in [n], l_i \in \mathbb{Z}\} \cup \{|u_i| : i \in [n], u_i \in \mathbb{Z}\}$, *and* $\zeta_{l,u,b} := \max\{\zeta_{l,u}, ||b||_\infty\}$.

i) There exists a feasible solution x_0 *with* $||x_0||_1 \leq 3n\zeta_{l,u,b} ||A||_\infty g_1(A)$.

ii) If the objective $f(x) = c^\intercal x$ *is linear, there exists an optimal solution* $x^\star$ *with* $||x^\star||_1 \leq 4n\zeta_{l,u,b} ||A||_\infty (g_1(A))^2$.

Proof. We start with showing the existence of x_0. The proof uses the same construction as we used for showing Proposition 2.30, though we have to adapt the definition of $\zeta_{l,u}$ according to the finite entries of l, u. Let $\overline{x}$ be the vector with $l \leq x \leq u$ minimizing $||x||_1$. We recenter the initial instance by substituting $x' = x - \overline{x}$, obtain

$$\begin{aligned} \max \quad & f'(x') := f(\overline{x} + x'), \\ s.t. \quad & Ax' = b' := b - A\overline{x}, \\ & x' \geq l' := l - \overline{x}, \\ & x' \leq u' := u - \overline{x}, \end{aligned} \tag{2.15}$$

and assume $b' \geq 0$, which can be achieved by multiplying single rows with -1. Observe that $||b'||_1 \leq ||b||_1 + n\zeta_{l,u} ||A||_\infty$. Moreover, $x' = 0 \in \mathbb{Z}^n$ satisfies the new box constraints. Therefore, the vector $\binom{0}{b'}$ is a feasible solution to the following *auxiliary ILP*:

$$\begin{aligned} \min \quad & \mathbf{1}^\intercal y \\ s.t. \quad & (A, \mathbb{1}) \begin{pmatrix} z \\ y \end{pmatrix} = b' \\ & l' \leq z \leq u' \\ & 0 \leq y \leq b' \\ & y, z \in \mathbb{Z}^n \end{aligned} \tag{2.16}$$

As we only attached the identity to A, we have $G_D(A) = G_D(A, \mathbb{1})$, and all parameters remain unchanged. Clearly, the initial solution $\binom{0}{b'}$ has objective value $||b'||_1$ and the optimal value is at least 0, with equality if and only if the initial ILP (2.1) has a feasible solution.

Let $\binom{z^\star}{0}$ be an optimum solution of minimum 1-norm to (2.16), and decompose the vector $\binom{z^\star}{0} - \binom{0}{b'} = \sum_{g\in S} \alpha_g g$ into a weighted sum of at most $4n-2$ Graver basis elements $g \sqsubseteq \binom{z^\star}{-b'}$, $S \subseteq \mathcal{G}((A, \mathbb{1}))$. Moreover, we know that for each g, either $\alpha_g = 0$ or $(0, \mathbf{1}^\intercal)g < 0$. Otherwise considering $\binom{z^\star}{0} - g\ \binom{z^\star}{0}$ yields a contradiction to the minimality of $\left|\left|\binom{z^\star}{0}\right|\right|_1$. But then $\sum_{g\in S} \alpha_g \leq \mathbf{1}^\intercal b' = ||b'||_1$, and thus

$$\begin{aligned} ||z^\star||_1 &\leq \sum_{g\in S} \alpha_g \, ||g||_1 \leq g_1(A) \, ||b'||_1 \\ &\leq g_1(A)(||b||_1 + n\zeta_{l,u} \, ||A||_\infty) \\ &\leq 2n\zeta_{l,u,b} \, ||A||_\infty \, g_1(A). \end{aligned}$$

Changing from the solution $z^\star$ of (2.6) back to our original IP (2.1), we obtain a feasible solution $x_0 := \overline{x} + z^\star$ with

$$||x_0||_1 \leq n\zeta_{l,u,b} + g_1(A)2n\zeta_{l,u,b} \, ||A||_\infty \leq 3n\zeta_{l,u,b} \, ||A||_\infty \, g_1(A). \tag{2.17}$$

Showing the existence of a short optimal solution for linear objectives works similar. Let x_0 be any feasible solution, and let $x^\star$ be an optimum solution closest to x_0 in terms of 1-norm. Similar as before, we can assume $0 \leq x^\star - x_0$, and decompose $x^\star - x_0 = \sum_{g\in S} \alpha_g g$ into the weighted sum of at most $2n-2$ Graver basis elements $S \subseteq \mathcal{G}(A)$. Since $x^\star$ is closest to x_0, we can assume that either $\alpha_g = 0$ or $c^\intercal g < 0$ for all $g \in S$, by considering $x^\star - g$. Fix some g. For every coordinate i, we must either have $0 < g_i \leq u_i < \infty$, or $-\infty < l_i \leq g_i < 0$, as otherwise the IP was unbounded (we can add g to $x^\star$). Since all $g \in S$ are sign-compatible, this implies $\sum_{g\in S} \alpha_i \leq ||x_0||_1 + n\zeta_{l,u}$. We obtain

$$\begin{aligned} ||x^\star||_1 &\leq ||x_0||_1 + \sum_{g\in S} \alpha_g \, ||g||_1 \\ &\leq ||x_0||_1 + g_1(A)(||x_0|| + n\zeta_{l,u}) \\ &\leq 4n\zeta_{l,u} \, ||A||_\infty \, (g_1(A))^2, \end{aligned}$$

using the previously derived bound (2.17) on x_0. □

A direct consequence is that we can also find an initial solution to IPs with infinite bounds.

Corollary 2.35. *Let a bounded IP* (2.1) *with possibly infinte bounds* $l, u \in (\mathbb{Z} \cup \{\pm\infty\})^n$ *be given, together with a* td*-decomposition* F *of* $G_D(A)$. *Define* $\zeta_{l,u} := \max\{|l_i| : i \in [n], l_i \in \mathbb{Z}\} \cup \{|u_i| : i \in [n], u_i \in \mathbb{Z}\}$, *and* $\zeta_{l,u,b} := \max\{\zeta_{l,u}, ||b||_\infty\}$.

i) We can find an initial feasible solution in time

$$\mathcal{O}(n \log_2(n)) \cdot (\log_2(n\zeta_{l,u,b}))^2 \cdot (g_1(A) \, ||A||_\infty)^{\mathcal{O}(\text{height}(F))}.$$

ii) *We can find an initial feasible solution in time*

$$\mathcal{O}(n^2(\log_2(n))^3) \cdot (\log_2(n\zeta_{l,u,b})) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\text{height}(F))}.$$

iii) *Assume the objective $f(x) = c^\intercal x$ is linear. If we can find an optimal solution for any finite bounds l', u' in time $T(||u' - l'||_\infty)$, then we can find an optimal solution for the given instance in time*

$$T(8n\zeta_{l,u,b}\,||A||_\infty\,(g_1(A))^2).$$

Proof. The first part follows from Proposition 2.30, by replacing l, u with l', u' defined as

$$l'_i := \begin{cases} l_i & l_i \in \mathbb{Z} \\ -3n\zeta_{l,u,b}\,||A||_\infty\,g_1(A) & l_i = -\infty \end{cases},$$
$$u'_i := \begin{cases} u_i & u_i \in \mathbb{Z} \\ 3n\zeta_{l,u,b}\,||A||_\infty\,g_1(A) & u_i = \infty \end{cases}$$

respectively. By Lemma 2.34, the new instance with finite bounds has a feasible solution if and only if the original instance has a solution. We then apply Proposition 2.30 to obtain Point *i*).

For the second point, we also start with replacing the bounds, and then construct the auxiliary instance (2.16) as in the proof of Proposition 2.30. Recall that for this auxiliary instance the vector $\binom{0}{b'}$ is an initial feasible solution, and all bounds are limited by $n\zeta_{l,u,b}$. But instead of running the normal optimization procedure, we apply the scaling algorithm of Proposition 2.27, obtaining a running time of

$$\mathcal{O}(n^2(\log_2(n))^3) \cdot \log_2(n\zeta_{l,u,b}) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\text{height}(F))}.$$

For the third point, we replace the bounds l, u in the same manner, but use the finite bounds of Point *ii*) in Lemma 2.34. □

Regarding separable convex objectives, we cannot limit the norm of any optimal solution. If this was the case, we could decide unboundedness: Replace the infinite bounds, and find an optimum solution $x^\star$ for this modified IP with finite bounds. Then, change back to the infinite bounds and solve a single augmentation IP 2.4, derived for $x^\star$ and $\lambda = 1$. If the problem is bounded, $x^\star$ is optimal already and we do not find an augmenting step. If the problem is unbounded, there exists an augmenting step and we find it with the derived augmentation IP 2.4. This is a contradiction to Lemma 2.33.

But if we have a feasible solution x_0 for a bounded IP, we can guess a bound ξ on

$||x_0 - x^\star||_\infty$, where $x^\star$ is an optimal solution closest to x_0. This leads to the following algorithm for bounded IPs with possibly infinite bounds l, u, with a somewhat unavoidable output-sensitive running time.

Proposition 2.36. *Let a bounded IP* (2.1) *with possibly infinite bounds and initial feasible solution x_0 be given. If there is an algorithm finding an optimal solution with running time $T(||u' - l'||_\infty)$ for any finite bounds l', u' for which x_0 is feasible, then there is an algorithm finding an optimal solution with running time*

$$(\log_2(||x_0 - \overline{x}||_\infty) + 1) \cdot T(4\,||x_0 - \overline{x}||_\infty),$$

for any infinte bounds l, u, where $\overline{x}$ is an optimal solution closest to x_0.

Proof. The proof follows from the discussion beforehand. We guess ξ, and run the algorithm for the instance with modified bounds $l'_i := \max\{l_i, (x_0)_i - \xi\}$, $u'_i := \min\{u_i, (x_0)_i + \xi\}$ respectively. Note that if $\xi \geq ||\overline{x} - x_0||_\infty$, the new instance with finite bounds contains a solution that is optimal for the original instance.

We run the algorithm for finite bounds on this instance, and obtain an optimum $\hat{x}$ for this instance. Then, we solve one more augmentation IP derived from the original instance for $\hat{x}$ and $\lambda = 1$. If we do not find an augmenting step, we output $\hat{x}$ as optimal. If we do, we must have $\xi < ||\overline{x} - x_0||_\infty$. We update $\xi \leftarrow 2\xi$, and repeat the procedure from above for x_0 again. Clearly, $\log_2(||\overline{x} - x_0||_\infty) + 1$ iterations suffice. □

One could formulate a similar result without demanding x_0, by letting the running time depend on $||\overline{x}||_\infty$. However, if we do not know whether the given IP is feasible, we have to guess at least $\xi \geq 3n\zeta_{l,u,b} g_1(A)\,||A||_\infty$.

2.6 The overall running time

In this section, we will assemble the pieces we discussed and derive several running times in which we can solve the integer programming problem (2.1). Before we recall the different parts of an algorithm, let us briefly address some necessary preprocessing steps, whose running time will be dominated by the rest of the algorithm.

First, we used an explicit td-decomposition of $G_D(A)$ for several results. While finding a (not necessarily optimal) td-decomposition is straight-forward for N-fold and tree-fold IPs, in general we can use an algorithm of Reidl et al. [Rei+14], computing a td-decomposition F roughly in time $2^{(\mathrm{td}(G))^2} \cdot |V(G)|$. As we have $g_1(A) \geq 2^{\mathrm{height}(F)}$, this running time is dominated by the term $(g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{height}(F))}$ appearing for solving the augmentation IP.

Furthermore, we assumed $m \leq n$ at certain points. Should the system $Ax = b$ have

row-rank $r < m$, we might want to extract an irredundant subsystem $A'x = b'$ of rank r. This is a reappearing problem in integer and linear programming, and can easily be solved with Gaussian elimination. We mention [GLS93, Thm. 1.4.8] as a reference. In the case of small tree-depth however, there is even a faster algorithm, computing an irredundant subsystem in time roughly $(\text{height}(F))^2(n+m)$, see [Fom+18].

We recall the developed tools at our disposal.

The proximity result. Optionally, we can apply the proximity result of Section 2.4.1. This gives us control on $||u-l||_\infty$, but we need to solve the continuous relaxation of the problem at hand. In particular, this allows us to assume that l, u are finite. To find a fractional solution (or at least, ε-accurate solution), we can for instance use Chubanov's algorithm [Chu16].

An initial solution. Before we can optimize, we have to find an initial feasible solution, which we achieve by setting up an auxiliary IP and solving it. The running time for this is

$$\mathcal{O}(n\log_2(n)) \cdot \log_2(||b'||_1) \cdot \log_2(\zeta_{l,u}) \cdot (g_1(A)\,||A||_\infty)^{\text{td}(F)},$$

where the second factor stems from the objective of the auxiliary IP (2.16). In the case of finite bounds, we are able to estimate $||b'||_1 \leq n\zeta_{l,u}$. In the case of infinite bounds the estimate has to depend on the right-hand side b as well.

Reducing the objective If the box constraints are finite, we can optionally apply the results of Section 2.3 for estimating the dependency on the objective. However, for the separable convex case, the linear factor $||u-l||_\infty$ would enter the running time, thus its benefit without a grasp on l, u (e.g. via the proximity results) is questionable.

The scaling approach If we deal with rather large bounds, we can start by finding an optimal solution in the sparser sublattice $2^k\mathbb{Z}^n$, refine it to a solution in the lattice $2^{k-1}\mathbb{Z}^n$, and iterate. This is very beneficial in combination with the reduced objective results for separable convex functions, as we can deploy the proximity result in each step and only have to reduce the objective on a small area.

Optimization via augmentation IPs As the core of the developed theory, we showed that optimization can be reduced to solving a family of augmentation IPs. The number of IPs we have to solve in total is $\mathcal{O}(n\log_2(n))\log_2(f_{\max})\log_2(\zeta_{l,u})$ (cf. Proposition 2.11).

Algorithm for augmentation IPs Finally we add in the running time for solving these augmentation IPs. Either, by solving every augmentation IP individually (Subection 2.2.1), or by using the strengthening via convolution in Subection 2.2.3.

In the following, we will discuss several scenarios. The most interesting ones were already stated in Theorem 2.2.

Several general running times for finite bounds.

In this subsection, we combine different results to obtain several running times. We will phrase the running times in two propositions, the first for linear objectives, the second for separable convex objectives. As the design of the algorithms will be the same for the case of linear and separable convex objectives, the proofs will only differ in some estimates. Therefore, we will give a combined proof for both propositions.

The intent of the following propositions is to highlight the dependency on n, therefore dependencies on l, u, c are expressed in the infinity-norm. As a consequence, rather crude estimates are made, such as $\log_2(||c||_1) \leq \log_2(n) \log_2(||c||_\infty)$. We also remarks that the assumption $f_{\max} \geq ||u-l||_\infty$ or even $f_{\max} \geq ||u-l||_1$ is arguable, and would reduce the $\log_2$-factors.

Theorem 2.37. *Let an ILP* (2.1) *with finite bounds and linear objective be given, define* $\zeta_{l,u} := ||u-l||_\infty$ *and let LP denote the time required to solve the LP relaxation of* (2.1)*. We can decide feasibility, and find an optimum solution if it exists, in any of the following running times.*

i) If we directly compute a feasible solution and optimize, we obtain

$$\mathcal{O}(n \log_2(n)^2) \cdot (\log_2(||c||_\infty \zeta_{l,u}))^2 \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{td}_D(A))}$$

ii) If we first solve the continuous relaxation and apply the proximity result, we obtain

$$\mathcal{O}(n(\log_2(n))^3) \cdot \log_2(||c||_\infty) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{td}_D(A))} + LP.$$

iii) If we solve the continuous relaxation, apply the proximity result and analyse with a reduced objective, we obtain

$$\mathcal{O}(n^2(\log_2(n))^3) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{td}_D(A))} + LP.$$

iv) If we use the scaling approach, together with the proximity result and the reduced objective, we obtain

$$\mathcal{O}(n^2(\log_2(n))^4) \cdot \log_2(\zeta_{l,u}) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{td}_D(A))}.$$

Theorem 2.38. *Let an IP* (2.1) *be given, define*

$$\zeta_{l,u} := ||u-l||_\infty \quad \textit{and} \quad f_{\max} := \max_{x,y \textit{ feasible}} (|f(x) - f(y)|),$$

and let CP denote the time to find a 1*-accurate solution for the continuous relaxation of* (2.1)*. We can decide feasibility and boundedness, and find an optimum solution if it exists, in any of the following running times.*

i) If we directly compute a feasible solution and optimize, we obtain

$$\mathcal{O}(n(\log_2(n))^2) \cdot \log_2(f_{\max}) \cdot (\log_2(\zeta_{l,u}))^2 \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{td}_D(A))}.$$

ii) If we first solve the continuous relaxation and apply the proximity result, we obtain

$$\mathcal{O}(n(\log_2(n))^3) \cdot \log_2(f_{\max}) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{td}_D(A))} + CP.$$

iii) If we solve the continuous relaxation, apply the proximity result and analyse with a reduced objective, we obtain

$$\mathcal{O}(n^3(\log_2(n))^3) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{td}_D(A))} + CP.$$

iv) If we use the scaling approach, together with the proximity result and the reduced objective, we obtain

$$\mathcal{O}(n^3(\log_2(n))^4) \cdot \log_2(\zeta_{l,u}) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{td}_D(A))}.$$

Proof of Theorems 2.37 and 2.38. In all cases, we can first find a td-decomposition F of the graph $G_D(A)$ in time $2^{(\mathrm{td}_D(A))^2}|V(G_D(A))|$, see [Rei+14]. If necessary, we then output a non-redundant subsystem $A'x = b'$ of full rank in time $(\mathrm{height}(F))^2(n+m)$, see [Fom+18]. Moreover, infeasibility can be detected while computing an initial feasible solution (or already while solving the continuous relaxation). The time needed for these steps is dominated by other terms (unless $m \gg n$), henceforth we will omit these steps.

i) In this case, we solve the IP in the most forward way. This is, we compute an initial feasible solution and optimize then with the augmenting procedure.

Finding an initial feasible solution can be done via Proposition 2.30 in time

$$\mathcal{O}(n\log_2(n)) \cdot \log_2(n\zeta_{l,u}) \cdot \log_2(\zeta_{l,u}) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{height}(F))}$$

where $\zeta_{l,u} := ||u-l||_\infty$.

Then, we run the augmenting procedure of Theorem 2.1, and obtain a running time of

$$\mathcal{O}(n\log_2(n)) \cdot \log_2(f(x_0) - f(x^\star)) \cdot \log_2(\zeta_{l,u}) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{height}(F))}$$

for this step. For a linear objective, we can estimate

$$\log_2(c^\intercal(x_0 - x^\star)) \leq \log_2(n)\log_2(||c||_\infty \zeta_{l,u}),$$

and the second running time dominates. For separable convex objectives, aggregating the running times yields the claimed result.

ii) In this case, we start by solving the continuous relaxation first, obtaining a fractional optimum $x^\star$. As there are several possibilities to do this, we let the reader choose their favorite algorithm, and abbreviate the running time by LP in the case of a linear objective, and by CP in the case of a separable convex objective.

We can deploy the proximity result (Proposition 2.24) and replace the initial bounds by $l' := \lceil x^\star - ng_1(A)\rceil$ and $u' := \lfloor x^\star + ng_1(A)\rfloor$. Finding a feasible initial vector x_0 in this new instance can be done in time

$$\begin{aligned} & \mathcal{O}(n\log_2(n)) \cdot \log_2(n\zeta_{l',u'}) \cdot \log_2(\zeta_{l',u'}) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\text{height}(F))} \\ \leq\ & \mathcal{O}(n\log_2(n)^3) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\text{height}(F))}, \end{aligned}$$

using Proposition 2.30 and plugging in the right values.

Finally, finding an optimal solution $x^\star$ can again be done with Theorem 2.1, in time

$$\mathcal{O}(n\log_2(n)^2) \cdot \log_2(f(x_0) - f(x^\star)) \cdot (g_1(A))^{\mathcal{O}(\text{td}_D(A))}. \tag{2.18}$$

In the linear case, we can estimate $\log_2(f(x_0) - f(x^\star)) \leq \log_2(||c||_\infty) \cdot \log_2(n^2 g_1(A))$ using the proximity bounds. This gives us the desired running times.

iii) The third running time follows from the previous by applying the objective reduction of Proposition 2.16 to c, or of Proposition 2.17 for f.

We continue the computation of the previous point at the Term (2.18), and estimate

$$\log_2(c^\intercal(x_0 - x^\star)) \leq \log_2((5n^2 g_1(A))^n) \in \mathcal{O}(n\log_2(n))g_1(A)$$

for a linear objective, whereas for a separable convex function we get

$$\log_2(f(x_0) - f(x^\star)) \leq \log_2((5n^2 g_1(A))^{n^2 g_1(A)}) \in \mathcal{O}(n^2\log_2(n))g_1(A).$$

This gives total running times of

$$\mathcal{O}(n^2(\log_2(n))^3)(g_1(A))^{\mathcal{O}(\text{td}_D(A))} \quad \text{and} \quad \mathcal{O}(n^3(\log_2(n))^3)(g_1(A))^{\mathcal{O}(\text{td}_D(A))}$$

as desired.

iv) In this approach, we want to use the proximity result without solving the LP relaxation first. This is done by deploying the scaling algorithm of Proposition 2.27. For finding an initial feasible solution, we set up the auxiliary IP as usual, having

$\binom{0}{b'}$ as a feasible solution. Instead of solving it, we recenter this instance again, so that $\binom{0}{0}$ becomes a feasible solution, and the quantity $||u' - l'||_\infty$ remains the same. We can now switch to a sparser lattice such that 0 is the only solution, and apply the scaling approach, taking time

$$\log_2(n\zeta_{l,u}) \cdot \mathcal{O}(n^2(\log_2(n))^3) \cdot (g_1(A))^{\mathcal{O}(\text{height}(F))}.$$

After substituting everything back, we have feasible solution x_0 for the original instance. We then apply the scaling algorithm to optimize x_0, taking time

$$\log_2(\zeta_{l,u}) \cdot \mathcal{O}(n^2(\log_2(n))^3) \cdot (g_1(A))^{\mathcal{O}(\text{height}(F))}.$$

Adding both running times yields the stated running time. If we compute x_0 without the scaling approach, we trade a $\log_2(n)$-factor against an additional $\log_2(\zeta_{l,u})$-factor.

□

The case of infinite bounds.

For possibly infinite bounds, recall that for linear objectives, we can check whether the IP is bounded in negligible running time, whereas the problem is undecidable for general separable convex functions. Henceforth we limit the discussion to bounded IPs.

Since the running times *ii*) and *iii*) of Theorems 2.37 and 2.38 are using a continuous solver in combination with the proximity result, they apply to infinite bounds as well (possibly with variations in the continuous solver).

Theorem 2.39. *Let an IP* (2.1) *with possibly infinite bounds* $l, u \in (\mathbb{Z} \cup \{\pm\infty\})^n$ *be given.*

(*a*) *Let the objective be linear.*

i) *If we use no particular techniques, we obtain a running time of*

$$\mathcal{O}(n \log_2(n)^4) \cdot (\log_2(||c||_\infty \zeta_{l,u,b}))^2 \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\text{td}_D(A))}.$$

ii) *If we use the scaling approach for finding an initial feasible solution and for optimization, we obtain*

$$\mathcal{O}(n^2(\log_2(n))^6) \cdot \log_2(\zeta_{l,u,b}) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\text{td}_D(A))}.$$

(*b*) *Let the objective be separable convex.*

i) If we use no particular techniques, we obtain a running time of

$$n(\log_2(n))^3 \left(\log_2(f_{\max}) \cdot (\log_2(\xi)^2) + (\log_2(\zeta_{l,u,b}))^2\right) (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{td}_D(A))},$$

where $\xi := ||x_0 - \overline{x}||_\infty + 1$ denotes the distance from the feasible solution x_0 we find to a closest optimal solution $\overline{x}$.

ii) If we use the scaling approach for finding an initial feasible solution and for optimization, we obtain

$$n^3(\log_2(n))^4 \left(\log_2(\xi)^2 + \log_2(\zeta_{l,u,b})\right) (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{td}_D(A))},$$

where $\xi := ||x_0 - \overline{x}||_\infty + 1$ denotes the distance from the feasible solution x_0 we find to a closest optimal solution $\overline{x}$.

Proof. Again, the time needed for preprocessing is negligible (as long as $m \in \mathcal{O}(n)$).

(a) For a linear objective, we immediately apply the second point of Corollary 2.35, and obtain the following running times.

i) By Theorem 2.37 Point *i)*, we have

$$\mathcal{O}(n \log_2(n)^4) \cdot (\log_2(||c||_\infty \zeta_{l,u,b}))^2 \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{td}_D(A))}.$$

ii) By Theorem 2.37 Point *iv)*, we have

$$\mathcal{O}(n^2(\log_2(n))^6) \cdot \log_2(\zeta_{l,u,b}) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{td}_D(A))}.$$

(b) For separable convex objectives, we apply either the first or the second point of Corollary 2.35 to find an initial solution x_0 (and decide feasibility) in one of the times

$$\begin{array}{ll} i) & \mathcal{O}(n \log_2(n)) \cdot (\log_2(n\zeta_{l,u,b}))^2 \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{height}(F))}, \\ ii) & \mathcal{O}(n^2(\log_2(n))^3) \cdot (\log_2(n\zeta_{l,u,b})) \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{height}(F))}. \end{array}$$

Then, we apply Proposition 2.36, letting us reduce this case to finite bounds in time $(\log_2(||x_0 - \overline{x}||_\infty) + 1) \cdot T(4\,||x_0 - \overline{x}||_\infty)$, where T is any running time in dependency on finite bounds.

i) By Theorem 2.38 Point *i)*, we have

$$\mathcal{O}(n(\log_2(n))^2) \cdot \log_2(f_{\max}) \cdot (\log_2(||x_0 - \overline{x}||_\infty) + 1))^3 \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{height}(F))}.$$

ii) By Theorem 2.38 Point *iv)*, we have

$$\mathcal{O}(n^3(\log_2(n))^4) \cdot \log_2(||x_0 - \overline{x}||_\infty + 1)^2 \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\mathrm{height}(F))}.$$

Combining the running times listed in Points *i*) gives the first claim, combining the running times in Points *ii*) the second. □

A strongly polynomial time algorithm for linear objectives.

As an immediate consequence of Theorem 2.37 Point *iii*), we obtain a strongly polynomial algorithm for integer linear programming, provided we have access to a td-decomposition of the constraint matrix.

Two important classes of IPs for which we have easy access are N-fold IPs and their generalization tree-fold IPs (Definitions 1.1 and 1.2).

Corollary 2.40. *A generalized n-fold ILP with parameters r and s is solvable in strongly FPT time*

$$n^2(\log_2(n))^3 \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(r+s)} + LP.$$

A generalized tree-fold ILP with parameters $\tau_1, \ldots, \tau_d$ is solvable in strongly FPT time

$$n^2(\log_2(n))^3 \cdot (g_1(A)\,||A||_\infty)^{\mathcal{O}(\sum_i \tau_i)} + LP.$$

Here, LP denotes the time complexity of any algorithm for the LP relaxation that runs in time polynomial in n and size(A) *only, and in space polynomial in the input size.*

Proof. As the parameters r, s or $\tau_1, \ldots, \tau_d$ are given to us, we can easily derive a td-decomposition F of the constraint matrix.

We first apply the algorithm of Frank & Tardos [FT87] (or any other algorithm for the LP relaxation meeting the requirements) in order to obtain an optimum solution of the LP relaxation.

Afterwards, we apply Theorem 2.37, Point *iii*), and obtain the claimed running time.

As we are only using Theorem 2.1, all occurring numbers are bounded as well, and the claim follows. □

Remark 2.41. *Even if only stated for finite bounds, this result also holds for infinite bounds. The augmenting steps are still bounded in terms of the parameters, and by Lemma 2.34, we know that the bit complexity of an optimal solution (and thus all points we have to consider) is bounded as well.*

The reason why we restrict ourselves to ILPs is the fact that we do not know how to find an ε-accurate solution for the continuous relaxation of an IP in strongly FPT time.

If the algorithm of [Rei+14] runs in strongly polynomial time, the same result holds for any ILP with bounded tree-depth.

2.7 Other parameters for integer programming

This section is concerned with justifying our choice of parameters. We first remark that we cannot parameterize by the (primal or dual) tree-depth alone (i.e. not considering $||A||_\infty$ as a parameter). It is shown in [GO18, Thm. 12] that even for a linear objective, the problem is NP-hard even for fixed primal tree-depth. For the dual tree-depth, we refer to [KKM17b, Thm. 5]. The authors show that the problem parameterized by the dual tree-depth alone is W[1]-hard, which is strong evidence that the problem is not in FPT.

Moreover, we can neither extend our results to non-separable convex functions ([Lee+12, Prop. 1], [Eis+19, Prop. 101]), nor to separable concave functions ([Eis+19, Prop. 101]).

The primal tree-depth as a parameter

In [Eis+19], also the primal graph $G_P(A)$ and its tree-depth is considered as a parameter. On a high level, this case is rather similar to the dual graph. It is also possible to show a bound on the Graver basis elements in terms of $\mathrm{td}(G_P(A))$. However, instead of a bound for $g_1(A)$, we obtain a bound $g_\infty(A)$ on the infinity-norm for all $g \in \mathcal{G}(A)$. What is more important, instead of being doubly-exponential in the tree-depth, its bound is a tower whose height corresponds to the topological height of the chosen td-decomposition F of $G_P(A)$. As we can also solve the augmentation IP in this case, we still obtain an FPT algorithm, admittedly with a much worse running time.

We cite the main results for the primal graph.

Lemma 2.42 ([Eis+19, Lemma 22])**.** *The augmentation IP* (2.5) *can be solved in time* $\mathrm{td}_P(A)^2(2g_\infty(A)+1)^{\mathrm{td}_P(A)}n$.

The constant $g_\infty(A)$ actually depends on the chosen td-decomposition. Its magnitude becomes clearer in the following lemma.

Lemma 2.43 ([Eis+19, Lemma 26])**.** *Let* $A \in \mathbb{Z}^{m\times n}$*,* F *be a* td*-decomposition of* $G_P(A)$*. Then there exists a constant* $\alpha \in \mathbb{N}$ *such that*

$$g_\infty(A) \leq \tau\left((2\,||A||_\infty)^{2^{\mathrm{th}(F)}\cdot\alpha\cdot\mathrm{td}_P(A)^2}\right),$$

where $\tau(x)$ *is a tower of height* $\mathrm{th}(F)-1$*, i.e. for* $\varphi(x)=2^x$*, we have* $\tau(x)=\varphi^{\mathrm{th}(F)-1}(x)$.

The function τ will be used in the following proposition again, where we give a selection of running times. For the full table, see [Eis+19, Corollary 88].

Lemma 2.44 (cf. [Eis+19, Corollary 88])**.** *Let an IP* (2.1) *be given, let* F *be an optimal*

td-*decomposition of* $G_P(A)$*, and define* $\zeta_{u-l} := ||u-l||_\infty$*. Moreover, define*

$$g := \tau\left(\mathcal{O}\left((2\,||A||_\infty)^{2^{\mathrm{th}(F)}\cdot\alpha\cdot \mathrm{td}_P(A)^2}\right)\right).$$

If $f(x) = c^\intercal x$ *and* LP *denotes the time needed to solve the linear relaxation of* (2.1)*, we can find an optimal solution of* (2.1) *in any of the following running times.*

$$i) \qquad n^2 \log_2(n) \cdot g \cdot (\log_2(\zeta_{l,u,c}))^2$$

$$ii) \qquad n^3 (\log_2(n))^2 \cdot g \cdot \log_2(\zeta_{l,u})$$

$$iii) \qquad n^3 (\log_2(n))^2 \cdot g + LP$$

If $f(x)$ *is a separable convex objective and* CP *denotes the time needed to find an* ε*-accurate solution of the continuous relaxation of* (2.1)*, we can find an optimal solution of* (2.1) *in any of the following running times.*

$$i) \qquad n^2 \cdot g \cdot \log_2(\zeta_{l,u}) \cdot \log_2(f_{\max})$$

$$ii) \qquad n^4 (\log_2(n))^2 \cdot g \cdot \log_2(\zeta_{l,u})$$

$$iii) \qquad n^4 (\log_2(n))^2 \cdot g + CP$$

As a direct consequence, also the tree-depth of the primal graph (together with $||A||_\infty$) is a valid parameter for integer programming.

Corollary 2.45. *Integer programming is in FPT parameterized by* $||A||_\infty$ *and* $\mathrm{td}(G_P(A))$*.*

The tree-width as a parameter

This section is concerned with the tree-width of a graph, $\mathrm{tw}(G)$, as defined in Section 1.7. The tree-width is a weaker parameter than the tree-depth, since we always have $\mathrm{tw}(G) + 1 \leq \mathrm{td}(G)$. To see this informally, let F be a td-decomposition of G, drawn down in a reasonable way (i.e. all leaves in the bottom row and non-crossing edges). We numerate the leaves $v_1, \ldots, v_m$, and for for every root-leaf path $P(r, v_i)$, we pack one bag B_i. Then, we take the path on the ordered bags, $B_1, \ldots, B_m$.

In the next Section, we will see that integer programming remains NP-hard even if $||A||_\infty = 2$ and $\mathrm{tw}_D(A) = 2$ (Corollary 2.47) or $\mathrm{tw}_P(A) = 2$ (Corollary 2.49).

In the context of the Graver augmenting technique, the main difference is that the

tree-width is not sufficient to limit the norm of Graver basis elements $g \in \mathcal{G}(A)$, i.e. $g_1(A)$ (and $g_\infty(A)$ for the primal tree-depth) any more.

However, if we add another parameter, there are still parameterized algorithms for integer programming for the primal tree-width, as well as for the dual tree-width.

The primal tree-width and the domain

It follows from Freuder's algorithm [Fre90] and was reproven by Jansen & Kratsch [JK15] that integer programming is solvable in time

$$n \cdot ||u - l||_\infty^{\mathcal{O}(\mathrm{tw}_P(A))}.$$

If we know in addition a bound $\tilde{g}_\infty(A)$ on the infinity-norm of the graver basis elements, the box constraints in the augmentation IP (2.5) can be replaced by $\min\{\tilde{g}_\infty(A), u_i\}$, $\max\{\tilde{g}_\infty(A), l_i\}$ respectively. Using one of the aforementioned algorithms to solve the augmentation IP, we obtain an algorithm parameterized by $g_\infty(A)$, $\mathrm{tw}_P(A)$ and $||A||_\infty$. See [Eis+19, Lem. 83] for details.

The dual tree-width and the domain

Similarly to the primal tree-depth, it was shown that $\mathrm{tw}_D(A)$ together with another parameter suffices to obtain an FPT algorithm for integer programming. Ganian, Ordyniak & Ramanujan [GOR17] showed that integer programming can be solved in time $n \cdot \Gamma^{\mathcal{O}(\mathrm{tw}_I(A))}$, where

$$\Gamma := \max\left\{ \left|\left| \sum_{i=1}^{k} A_i x_i \right|\right|_\infty : i \in [n],\ x \text{ a feasible solution} \right\}.$$

Here, A_i denotes the i-th column of A and $\mathrm{tw}_I(A)$ denotes the tree-width of the *incidence graph*, where we have a vertex i for each row, a vertex j for each column, and an edge (i, j) if $a_{i,j} \neq 0$. We always have $\mathrm{tw}_I(A) - 1 \leq \mathrm{tw}_P(A), \mathrm{tw}_D(A)$ [KV00]. With a little bit more work the same argumentation as in the primal case applies: We can limit Γ in terms of $g_1(A)$ and $||A||_\infty$, provided we know a bound $g_1(A)$ on the 1-norm of the Graver basis elements of A. This yields an FPT algorithm for integer programming parameterized by $||A||_\infty, g_1(A)$ and $\mathrm{tw}_D(A)$, the details can be found in [Eis+19].

The incidence graph

Yet another graph associated with a constraint matrix is the *incidence graph* $G_I(A)$. As mentioned in the previous paragraph $G_I(A)$ is formed by taking a vertex i for each row, a vertex j for each column, and an edge (i, j) if $a_{i,j} \neq 0$.

Ganian, Ordyniak & Ramanujan [GOR17, Thm. 12] showed that integer programming is NP-hard, already when $\mathrm{tw}_I(A) = 3$ and $||A||_\infty = 2$. In the previous paragraph we saw already that if we additionally take the parameter Γ related to the prefix-sum of Ax for any feasible solution x, we obtain a fixed-parameter tractable algorithm.

The hardness result of Ganian et al. was extended by Eiben et al. [Eib+19], who showed that integer programmings is NP-hard already for $\mathrm{td}_I(A) = 5$ and $||A||_\infty = 2$. This shows that we cannot replace the dual tree-depth by this more permissive parameter.

The underlying matroid and its branch-depth

The parameter tree-depth is in some sense quite unsatisfying. If we apply unimodular row operations on the matrix A, the whole geometric structure of the problem does not change, only its description. However, as long as we do not have any zero-columns (i.e. trivial variables) we can perform row operations on A and obtain a matrix A' without any zero entries. The dual tree-depth of A' is m, though we might have started with $\mathrm{td}_D(A)$ being constant. And while both matrices describe the very same problem, our algorithms are quite inefficient for A'.

The *branch-depth of* A is a parameter coming from to the vector matroid described by the columns of A. As row operations do not change the vector matroid, the matrices A and A' still have the same branch depth. Chan et al. [Cha+19] showed that the branch depth of A equals $\min_{A'}(\mathrm{td}_D(A'))$, where the minimum is taken over all matrices A' that are row-equivalent to A. Since they provide an algorithm approximating the best row-equivalent matrix (admittedly only to an exponential factor), integer programming is also fixed-parameter tractable with respect to the parameters $||A||_\infty$ and the branch-depth. However, when computing the row-equivalent matrix, the parameter $||A||_\infty$ might grow significantly. More precisely, they denote with $ec(A)$ the *entry complexity* of A, i.e. the maximum bit complexity over all entries of A. The upper bound on the entry complexity of the row-equivalent matrix A' they obtain is bounded by $\mathrm{ec}(A') \in \mathcal{O}(d4^d \,\mathrm{ec}(A))$.

2.8 An ETH-based lower bound

In this section, we will derive lower bounds on the running time for the problem at hand, assuming the *exponential time hypothesis (ETH)*, stating that the 3-*satisfiability problem (3-Sat)* cannot be solved in subexponential time in the worst case. To the best of our knowledge, we are the first ones providing a lower bound for the primal tree-depth as a parameter. For the dual tree-depth, we obtain a lower bound that asymptotically matches the one of Knop, Pilipczuk & Wrochna [KP]. But while they are exclusively interested in the tree-depth, we will also take the topological height into consideration. Hence, we also obtain lower bounds for every fixed topological height. Applied to the

case when the topological height is 2, this yields the currently best lower bound for N-fold IPs.

Prerequisites and NP-hardness for tree-width

Crucial to our reductions will be the famous NP-hard subset sum problem:

Subset Sum

INSTANCE: Positive integers $a_1, \ldots, a_n$, and a target $b \in \mathbb{Z}_{\geq 1}$.

TASK: Decide whether there is a subset $I \subseteq [n]$ such that $\sum_{i \in I} a_i = b$.

Our goal is to encode an instance of the subset sum problem as an IP with suitable (primal or dual) tree-depth, and then take advantage of existing hardness results for the subset sum problem. On the way, we will also encounter the hardness results for the parameter tree-width, mentioned in Section 2.7.

The encoding of subset sum works in several steps. We start by taking a subset sum instance with large items a_i, and encode it in the natural way as an IP. Of course, this gives us large coefficients in the matrix A, which is undesirable for our parameter $||A||_\infty$. The next step is thus to encode the items into variables with large values, introducing new variables. Then, we show that the dual graph of the obtained constraint matrix has a tree-decomposition of width 2. Moreover, the tree we get will actually be a path P, i.e. we have a *path-decomposition* of width 2. In order to obtain this decomposition for the primal graph as well, we have to add another step in the encoding. At this point, we can conclude NP-hardness for the parameters $\mathrm{tw}_P(A)$ and $\mathrm{tw}_D(A)$.

For the primal case, we construct a td-decomposition for the primal graph, dual graph respectively. On a high level, we embed the path P from the path-decomposition into a special tree, and then show that we can "unfold" every bag and delete duplicate vertices in a way that does not increase the height of the tree too much, and keeps the structure of a td-decomposition.

The dual case is more tedious, and will construct a td-decomposition from scratch.

We begin with the natural encoding of subset sum with n boolean variables $x_1, \ldots, x_n$:

$$\begin{aligned} \sum_{i=1}^{n} a_i x_i &= b \\ 0 \leq x_i &\leq 1 \end{aligned} \tag{2.19}$$

Let $\Delta \in \mathbb{N}_{\geq 2}$, and assuming $a_i \leq b$ for all $i \in [n]$, let $L_\Delta := \lceil \log_\Delta(b+1) \rceil$. Denote by $[a_i]_\Delta = (\alpha_i^{(0)}, \ldots, \alpha_i^{(L_\Delta - 1)})$ the base-Δ encoding of a_i, i.e., $a_i = \sum_{j=0}^{L_\Delta - 1} \alpha_i^{(j)} \Delta^j$. For

each i, we introduce new variables $y_i^{(j)}$, playing the role of the powers Δ^j. Thus, $a_i x_i = \sum_{j=0}^{L_\Delta - 1} \alpha_i^{(j)} y_i^{(j)}$. Now, let

$$y_i^{(0)} = x_i \qquad \forall i \in [n] \qquad (X_i)$$

$$y_i^{(j)} = \Delta \cdot y_i^{(j-1)} \qquad \forall i \in [n], \forall j \in [L_\Delta - 1] \qquad (Y_i^{(j)})$$

$$\sum_{i=1}^{n} \sum_{j=0}^{L_\Delta - 1} \alpha_i^{(j)} y_i^{(j)} = b. \qquad (S)$$

Due to constraint (X_i), we will disregard the variables x_i from now on and work with the variables $y_i^{(0)}$.

Lemma 2.46. *Let A be the matrix of constraints $(Y_i^{(j)})$–(S). $G_D(A)$ has a path decomposition of width 2 and length $n(L_\Delta - 1) - 1$.*

Proof. Let A be the matrix of constraints $(Y_i^{(j)})$ and the constraint (S). The graph $G_D(A)$ contains the following edges:

- Between S and each $Y_i^{(j)}$.
- Between $Y_i^{(j-1)}$ and $Y_i^{(j)}$ for each $i \in [n]$ and $j \in [L_\Delta - 1]$.

We construct a tree decomposition (in fact, a path decomposition), by consecutively taking the following segment of bags for each $i \in [n]$:

$$\{S, Y_i^{(0)}, Y_i^{(1)}\}, \{S, Y_i^{(1)}, Y_i^{(2)}\}, \ldots, \{S, Y_i^{(L_\Delta - 2)}, Y_i^{(L_\Delta - 1)}\} \ .$$

Since each bag is of size 3, the treewidth is 2. Moreover, since each segment comprises $L_\Delta - 1$ bags, and there are n segments, the length of the path decomposition is $n(L_\Delta - 1) - 1$. Note that there are nL_Δ variables and $n(L_\Delta - 1) + 1$ constraints. □

This already provides us with the first hardness result.

Corollary 2.47 (tw_D hardness)**.** *Integer programming is NP-hard already when $\mathrm{tw}_D(A) = 2$ and $\|A\|_\infty = 2$.*

Proof. Let $\Delta = 2$ and apply Lemma 2.46. □

When it comes to the primal tree-width, the variables are the vertices of $G_P(A)$, and

constraint (S) corresponds to a large clique. Therefore, we split up (S) into partial sums:

$$z_i^{(j)} = \begin{cases} y_1^{(0)} & \text{if } i = 1,\ j = 0 \\ z_{i-1}^{(L_\Delta - 1)} + \alpha_i^{(0)} y_i^{(0)} & \text{if } i > 1, j = 0 \\ z_i^{(j-1)} + \alpha_i^{(j)} y_i^{(j)} & \text{if } j > 0 \end{cases} \qquad (Z_i^{(j)})$$

$$z_n^{(L_\Delta - 1)} = b \qquad (S')$$

The intuitive meaning of $z_i^{(j)}$ is that it is a prefix sum of the constraint (S), i.e.,

$$z_i^{(j)} = \left(\sum_{k=1}^{i-1} \sum_{\ell=0}^{L_\Delta - 1} \alpha_k^{(\ell)} y_k^{(\ell)} \right) + \left(\sum_{\ell=0}^{j} \alpha_i^{(\ell)} y_i^{(\ell)} \right).$$

Lemma 2.48. *Let A be the matrix of constraints $(Y_i^{(j)})$, $(Z_i^{(j)})$, and (S'). $G_P(A)$ has a path decomposition of width 2 and length at most $2nL_\Delta$.*

Proof. Let us analyze the primal treewidth of constraints $(Y_i^{(j)})$, $(Z_i^{(j)})$, and (S'). We shall again disregard the variables x_i and simply identify them with $y_i^{(0)}$. Denoting the constraint matrix as A, the graph $G_P(A)$ has the following edges:

i) $\{y_i^{(j-1)}, y_i^{(j)}\}$, $\{z_i^{(j)}, z_i^{(j-1)}\}$, and $\{y_i^{(j)}, z_i^{(j-1)}\}$ for each $i \in [n]$ and $j \in [L_\Delta - 1]$,

ii) $\{z_i^{(j)}, y_i^{(j)}\}$ for each $i \in [n]$ and $j \in [0 : L_\Delta - 1]$,

iii) $\{z_{i-1}^{(L_\Delta - 1)}, y_i^{(0)}\}$ and $\{z_i^{(0)}, z_{i-1}^{(L_\Delta - 1)}\}$ for each $i \in [n]$.

We construct a path decomposition of $G_P(A)$ of width 2 and length $2nL_\Delta - n - 1$ as follows.

- For each i, form a path segment by alternatingly taking the bags $\{y_i^{(j-1)}, z_i^{(j-1)}, y_i^{(j)}\}$ and $\{z_i^{(j-1)}, y_i^{(j)}, z_i^{(j)}\}$ for $j = 1, \ldots, L_\Delta - 1$. These bags contain the edges of the first two types.
- For connecting these segments, take a bag $\{z_{i-1}^{(L_\Delta - 1)}, y_i^{(0)}, z_i^{(0)}\}$. These bags contain the edges of the third type.

Moreover, $\|A\|_\infty = \Delta$, the number of variables is $2nL_\Delta$, and the number of constraints is $2nL_\Delta - n + 1$. □

Corollary 2.49 (tw_P hardness)**.** *Integer programming is NP-hard already when $\mathrm{tw}_P(A) = 2$ and $\|A\|_\infty = 2$.*

Proof. Again, let $\Delta = 2$ and apply Lemma 2.48. □

Let us turn our attention to tree-depth. Say that an instance $(a_1, \ldots, a_n, b)$ of subset sum is *balanced* if the encoding length of b is roughly n, i.e., if $n \in \Theta(\log_2(b))$. We will use the following ETH-based lower bound for subset sum:

Proposition 2.50 ([KP])**.** *Unless the ETH fails, there is no algorithm for the subset sum problem which solves every balanced instance in time* $2^{o(n+\log_2(b))}$.

We remark that this proposition is obtained via the standard NP-hardness reduction from 3-Sat to subset sum, which starts from a 3-Sat formula with n variables and m clauses and produces a subset sum instance with $\tilde{n} = 2(n+m)$ and $3(n+m) \leq \log_2(b) \leq 4(n+m)$ [Cor+09, Thm. 34.15], hence $\frac{3}{2}\tilde{n} \leq \log_2(b) \leq 2\tilde{n}$. This is the reason why the lower bound holds for balanced instances.

In the next definition, we want to define a tree which is in some sense maximal among all trees with the same level heights and an additional constraint on the degrees of non-degenerate vertices.

Definition 2.51. *Let* $\kappa = (k_1, \ldots, k_\ell) \in \mathbb{N}^\ell$ *and denote by* F_κ *the maximal (w.r.t. the number of vertices) rooted tree such that each root-leaf path* P *of* F *satisfies the following:*

i) it contains ℓ *non-degenerate vertices, i.e.,* $\mathrm{th}(F_\kappa) = \ell$,

ii) $k_i(P) = k_i$ *for each* $i \in [\ell]$*, thus* P *has length* $\|\kappa\|_1$,

iii) the i*-th non-degenerate vertex on* P *has (in* F*) out-degree* $k_i + 1$*, for each* $i \in [\ell - 1]$.

It is easy to verify that the conditions *i)* to *iii)* already define F_κ uniquely. Conditions *i)* and *ii)* define the depths of all non-degenerate vertices (in particular the leaves), and Condition *iii)* defines the number of children at each non-degenerate vertex. Hence, we can easily determine the number of vertices in F_κ.

Lemma 2.52. *Let* $\kappa \in \mathbb{N}^\ell$*. Then* F_κ *has* $K_\kappa := \left(\prod_{i=1}^{\ell}(k_i + 1)\right) - 1$ *vertices.*

Proof. The proof goes by induction on ℓ. In the base case when $\ell = 1$, F_{k_1} is a path on k_1 vertices and clearly $(k_1 + 1) - 1 = k_1$. In the induction step, let $\kappa' := (k_2, \ldots, k_\ell)$, so by the induction hypothesis $K_{\kappa'} = \left(\prod_{i=2}^{\ell}(k_i + 1)\right) - 1$. Observe the structure of F_κ: the segment between its root and its first non-degenerate vertex v is a path on k_1 vertices, and the subtree of each child of v is isomorphic to $F_{\kappa'}$ and hence has $K_{\kappa'}$ vertices. Thus,

the number of vertices of F_κ is

$$\begin{aligned} K_\kappa &= \underbrace{k_1}_{\text{path}} + \underbrace{(k_1+1)K_{\kappa'}}_{\text{subtrees}} \\ &= k_1 + (k_1+1)\left(\prod_{i=2}^{\ell}(k_i+1) - 1\right) \\ &= \left(\prod_{i=1}^{\ell}(k_i+1)\right) - 1 \end{aligned}$$

□

Lemma 2.53. *Let $k \in \mathbb{N}^\ell$, K_κ as in Lemma 2.52, P_{K_κ} be a path on K_κ vertices. Then F_κ is isomorphic to a* td*-decomposition of P_{K_κ}.*

For the proof, we recall that the closure of a tree is obtained by adding all edges between any two vertices in an ancestral relationship. Hence, a graph F is a td-decomposition of a graph G if and only if $|V(F)| = |V(G)|$ and there is a monomorphism sending G to $\mathrm{cl}(F)$.

Proof. We induct on ℓ. In the base case $\ell = 1$, $F_{(k_1)}$ is a path on k_1 vertices, so clearly F_{k_1} is a td-decomposition of P_{k_1}. Assume that the claim holds for all $\ell' < \ell$ and let $\kappa' := (k_2, \ldots, k_\ell)$. Note that in P_{K_κ} there exist k_1 vertices whose deletion partitions P_{K_κ} into $k_1 + 1$ paths on $K_{\kappa'}$ vertices. Denote these vertices by $v_1, \ldots, v_{k_1}$ and let P' be the path $(v_1, v_2, \ldots, v_{k_1})$. By the inductive hypothesis $F_{\kappa'}$ is a td-decomposition of $P_{K_{\kappa'}}$. Take $k_1 + 1$ copies of $F_{\kappa'}$ and connect each of its roots to v_{k_1} by an edge. Then P_{K_κ} is contained in the closure of this tree, and it is easy to see that this tree is isomorphic to F_κ. □

Lower bound for primal tree-depth

The lower bound for the primal tree-depth can be derived fairly simple. We pick a balanced subset sum instance, encode it as discussed, and obtain a path-decomposition. With Lemma 2.53, we obtain a td-decomposition $\hat{F}$ for this path. Then, we unfold every bag in a way that keeps the td structure, does not increase the topological height of the tree, and increases the height by at most a factor of 3. To finish the proof, we only have to compare the parameters of the constructed tree F with the parameters of the initial subset sum instance.

The unfolding is done with the following lemma.

Lemma 2.54. *Let a graph G be given, together with a path-decomposition P of width β. Let F be a* td*-decomposition for P. Then we can obtain a* td*-decomposition of G s.t.* $\mathrm{th}(F) \leq \mathrm{th}(F')$*, and* $\mathrm{height}(F') \leq (\beta+1) \cdot \mathrm{height}(F)$.

Proof. We obtain F' as follows. We will replace every bag $B = \{w_1, \dots, w_\beta\}$ by a path $P_B = (w_1, \dots, w_\beta)$, connect one end to the parent of B and all children of B to the other end of P_B. Having possibly several copies of each vertex $v \in V(G)$, we only keep the copy closest to the root, and contract all others into their respective parent. It remains to show that this is a td-decomposition.

Let v be a vertex of G, and $\tau(v)$ that bag in P that is closest (in F) to the root. Since $\tau(v)$ has to be connected to all other bags containing v, we know that the subtree F_v rooted at $\tau(v)$ contains all bags containing v. Now let u be a vertex connected to v in G. If the bag $\tau(u)$ is either contained in the subtree F_v or in the path from $\tau(v)$ to the root, we are done. If the bag $\tau(u)$ is in a different branch than F_v, observe that there has to be a bag B in F_v containing both u and v. But since B and $\tau(u)$ have to be connected in P, and all edges of P have to be in the closure of F, there has to be a bag containing u and being closer to the root than $\tau(u)$. This is a contradiction. □

We are now able to show the lower bound.

Theorem 2.55. *Unless the ETH fails, there is no algorithm that solves every IP with* $2 \leq ||A||_\infty \leq \Delta$ *and for which a primal* td*-decomposition* F *with* $\mathrm{th}(F) \leq \ell$ *exists in time*

$$2^{o\left(\sqrt{\log_2(\Delta)}\left(\frac{\mathrm{td}_P(A)}{3\ell}\right)^{\ell/2}\right)}.$$

Proof. Let ℓ and Δ be given, and pick a balanced subset sum instance with n large enough. In particular, we will assume $\Delta \ll b$. We have

$$\frac{3/2n}{\log_2(\Delta)} \leq \log_\Delta(b) \leq \frac{2n}{\log_2(\Delta)}.$$

In accordance with Lemma 2.48, we encode this instance into an IP with constraint matrix A, and obtain a path decomposition P of $G_P(A)$. More precisely, P has width 2 and length at most $2nL_\Delta$, where

$$\frac{3n}{2\log_2(\Delta)} \leq L_\Delta = \lceil \log_\Delta(b+1) \rceil \leq \tfrac{2n}{\log_2(\Delta)} + 2 \leq \frac{3n}{\log_2(\Delta)}.$$

We choose $k \in \mathbb{Z}_{\geq 1}$ subject to

$$k^\ell - 1 < 2nL_\Delta \leq (k+1)^\ell - 1, \tag{2.20}$$

and by Lemma 2.53, F_κ with $\kappa = \{k\}^\ell$ is a td-decomposition for P, possibly after adding dummy nodes in P. By Lemma 2.54, this gives us a td-decomposition F for $G_P(A)$ with topological height at most ℓ, level heights at most $3k$, and total height at most $3k\ell$. Now

we estimate

$$\begin{aligned} n + \log_2(b) \geq \frac{5n}{2} &\geq \sqrt{\frac{25\log_2(\Delta)}{24}} \left(\frac{6n^2}{\log_2(\Delta)}\right)^{1/2} \\ &\geq \sqrt{\frac{25\log_2(\Delta)}{24}} \left(2nL_\Delta\right)^{1/2} \\ &\geq \sqrt{\frac{25\log_2(\Delta)}{24}} \left(k^\ell - 1\right)^{1/2} \\ &\geq o\left(\sqrt{\log_2(\Delta)} \left(\frac{\operatorname{height}(F)}{3\operatorname{th}(F)}\right)^{\ell/2}\right). \end{aligned}$$

□

When $d > 3\ell$, this is a double-exponential lower bound in terms of the topological height. Vice versa, we can derive a double-exponential lower bound in terms of the primal tree-depth.

Corollary 2.56. *Unless the ETH fails, there is no algorithm that solves every IP with* $2 \leq ||A||_\infty \leq \Delta$ *and for which a primal* td*-decomposition* F *with* $\operatorname{th}(F) \leq \ell$ *exists in time*

$$2^{\sqrt{\log_2(\Delta)}2^{o(\operatorname{td}_P(A))}}.$$

Proof. This actually follows from the previous proof. Instead of choosing k according to ℓ in Equation (2.20), we fix $k = 2$, and choose ℓ such that $2^\ell - 1 \leq 2nL_\Delta \leq 3^\ell - 1$. Since $2^{\ell+1} \leq 3^\ell$, this is always possible for large enough n. The tree F we obtain this way satisfies $\operatorname{height}(F) \leq 6\ell$, and we obtain the claimed bound. □

Lower bound for dual tree-depth

For the dual case, the construction becomes more involved. An important point is the fact that the constraint (S) is the only constraint using all variables, and the other constraints naturally decompose into blocks. This means, if we constructed a td-decomposition right away, the first level height would be 1, and the resulting lower bound would essentially depend on $\operatorname{th}(F) - 1$. Since $\operatorname{th}(F) = 2$ for N-fold IPs, this bound would not be strong enough for our purposes.

In order to prevent this from happening, we blow up the constraint S into k constraints, by considering a multidimensional version of the subset sum problem.

Multidimensional Subset Sum

INSTANCE: Integral vectors $a_1, \ldots, a_n, b \in \mathbb{Z}^k$, multiplicities $u \in (\mathbb{Z}_{>0} \cup \{\infty\})^n$.
TASK: Decide whether there exists $x \in \mathbb{Z}$, $0 \leq x \leq u$ such that $\sum_{i=1}^n a_i x_i = b$.

For short, we say that an instance of the multidimensional subset sum problem of dimension k is a *k-dimensional subset sum instance*. This problem generalizes the subset sum problem in two ways. First, the vectors a_i may also contain non-positive entries. Moreover, every item a_i has a multiplicity u_i, hence we can choose it up to u_i times. The following lemma easily follows by choosing $k = 1$.

Lemma 2.57. *The multidimensional subset sum problem is NP-hard even in fixed dimension.*

Lemma 2.58. *Let $a_1, \ldots, a_n, b \in \mathbb{Z}_{\geq 1}$ be an instance of the subset sum problem, $M := \max\{a_i : i = 1, \ldots, n\} \cup \{b\}$, and $\Delta \in \mathbb{Z}_{\geq 2}$. There is an equivalent instance[1] of the multidimensional subset sum problem with dimension $\lceil \log_\Delta(M+1) \rceil$, $n + \lceil \log_\Delta(M+1) \rceil - 1$ items, and all numbers bounded by Δ in absolute value.*

Proof. Let $k := \lceil \log_\Delta(M+1) \rceil$. Recall that $[a_i]_\Delta = (\alpha_i^{(0)}, \ldots, \alpha_i^{(k-1)})$ is the base-Δ encoding of a_i, and similarly for $[b]_\Delta = (\beta^{(0)}, \ldots, \beta^{(r-1)})$. Consider the following system of equations.

$$\begin{pmatrix} \alpha_1^{(0)} & \alpha_2^{(0)} & \ldots & \alpha_n^{(0)} & -\Delta & & \\ \alpha_1^{(1)} & \alpha_2^{(1)} & \ldots & \alpha_n^{(1)} & 1 & \ddots & \\ \vdots & \vdots & & \vdots & & \ddots & -\Delta \\ \alpha_1^{(k-1)} & \alpha_2^{(k-1)} & \ldots & a_n^{(k-1)} & & & 1 \end{pmatrix} \begin{pmatrix} x \\ s \end{pmatrix} = \begin{pmatrix} \beta^{(0)} \\ \beta^{(1)} \\ \vdots \\ \beta^{(k-1)} \end{pmatrix},$$

with constraints $0 \leq x \leq 1$. If we multiply the vector $(1, \Delta, \Delta^2, \ldots, \Delta^{k-1})$ from the left on both sides, we retrieve the original subset sum instance where only the x variables occur. Hence, the projection $\pi : (x, s) \mapsto x$ maps solutions to solutions. To see that every solution x has a preimage, consider the following reformulation of the initial instance, where we set $s_0 := s_k := 0$.

$$\begin{aligned} \sum_{i=1}^n a_i x_i &= \sum_{i=1}^n \sum_{j=0}^{k-1} \alpha_i^{(j)} \Delta^j x_i = \sum_{j=0}^{k-1} \left(\sum_{i=1}^n \alpha_i^{(j)} x_i \right) \Delta^j \\ &= \sum_{j=0}^{k-1} \left(\underbrace{s_j + \sum_{i=1}^n \alpha_i^{(j)} x_i - \Delta s_{j+1}}_{=:\sigma^{(j)}} \right) \Delta^j \qquad (2.21) \\ &= \beta^{(0)} + \beta^{(1)} \Delta + \cdots + \beta^{(k-1)} \Delta^{k-1}. \qquad (2.22) \end{aligned}$$

[1] i.e., there is a bijection between the sets of solutions

We show that we can iteratively choose the $s_i \in \mathbb{Z}_{\geq 0}$ such that for each summand $\sigma^{(j)} = \beta^{(j)}$ holds, which are precisely the equalities of the k-dimensional subset sum instance.

First, observe that $\sigma^{(1)} \equiv \beta^{(1)} \mod \Delta$. Since the $\alpha_i^{(0)}$ are nonnegative, $\beta^{(0)} < \Delta$, and $s_0 = 0$, there exists a nonnegative integer s_1 such that $\sigma^{(0)} = \beta^{(0)}$. Since s_1 is fixed, we can apply the same arguments for s_2, continuing up to s_{k-1}, and obtain that $\sigma^{(j)} = \beta^{(j)}$ for $j \leq k-2$. To see that this implies already $\sigma^{(k-1)} = \beta^{(k-1)}$, observe that both Sums (2.21) and (2.22) evaluate to b, as we assumed x feasible. As all other summands coincide, the last two summands have to coincide as well.

As the s_i are uniquely determined, we have a bijection between solutions. □

Theorem 2.59. *Assuming the ETH, there is no algorithm solving every k-dimensional subset sum instance with absolute values of the entries bounded by Δ in time $2^{o(n-k)}\Delta^{o(k)}$.*

Proof. Choose a balanced subset sum instance with n' items, and for $\Delta \in \mathbb{Z}_{\geq 2}$, let $k \in \mathbb{Z}$ be the unique integer such that $\Delta^{k-1} \leq b+1 < \Delta^k$. By Lemma 2.58, there exists an equivalent k-dimensional subset sum instance with $n = n' + k - 1$ items and entries bounded by Δ. As we cannot solve the initial instance faster than $2^{o(n'+\log_2(b))}$, where $3/2n \leq \log_2(b) \leq 2n$, we cannot solve the equivalent instance faster than

$$\begin{aligned} 2^{o(n'+\log_2 b)} &\geq 2^{o(n-k)}2^{o((k-1)\log_2(\Delta))} \\ &\geq 2^{o(n-k)}\Delta^{o((k-1)}. \end{aligned}$$

□

Before we proceed, let us recall the encoding of the subset sum instance we chose. We had Constraints $(Y_i^{(j)})$ to encode the large item sizes into variables, and one Constraint (S) summing over all items, corresponding to the equation

$$a_1x_1 + \cdots + a_nx_n = b.$$

As the Constraints $(Y_i^{(j)})$ correspond to the variables, they will not change. However, for a k-dimensional subset sum instance we obtain k summing constraints. Let $Ax = b$ be a k-dimensional subset sum instance, with entries $a_{i,j}$ and base-Δ encodings $[a_{i,j}]_\Delta = (\alpha_{i,j}^{(0)}, \ldots, \alpha_{i,j}^{(L_\Delta - 1)})$, where L_Δ is the maximal Δ-encoding length of all appearing numbers. We obtain the same encoding for each row $j \in [k]$, yielding constraints

$$\sum_{i=1}^{n}\sum_{\ell=0}^{L_\Delta-1} \alpha_{j,i}^{(\ell)} y_j^{(\ell)} = b_j \qquad \forall j \in [k]. \qquad (S_j)$$

We can now turn to the lower bound. It might be tempting to start with a k-dimensional subset sum instance and reduce it to integer programming. However, we need the dimension k to fulfill a certain relation with the topological height ℓ and the largest entry Δ. For this reason, we think it is cleaner to start with a subset sum instance, allowing us to choose the k appropriately.

Theorem 2.60. *Unless the ETH fails, there is no algorithm for IPs with* $2 \leq ||A||_\infty \leq \Delta$ *and a* td*-decomposition* F *s.t.* $\mathrm{th}(F) = \ell$ *and* $\mathrm{td}(F) = d$ *in time*

$$\Delta^{o\left(\left(\frac{d}{\ell}+\frac{1}{12}\right)^{\ell}\right)}$$

in the worst case. Specifically, assuming ETH, no algorithm solves every generalized N*-fold IP in time*

$$\Delta^{o\left((r+s)^2\right)}.$$

As the computations in the proof are rather tedious, we outline the strategy first, omitting constants.

Starting with a balanced subset sum instance with large enough n and given ℓ and Δ, we choose k such that $b \approx \Delta^{k^\ell}$. We encode it into a k-dimensional subset sum instance, where all entries are bounded by $\Delta^{k^{\ell-1}}$. Then, we encode this instance further into Constraints (S_j) and $(Y_i^{(j)})$, so that all entries are bounded by Δ. For each variable, the Constraints $(Y_i^{(j)})$ form a path on $k^{\ell-1}$ vertices in $G_D(A)$. Hence, for $\kappa \approx \{k\}^{\ell-1}$, the tree F_κ constitutes a td-decomposition of level heights k and topological height $\ell - 1$. We take a path on the k Constraints (S_j), and attach one copy of F_κ for each variable at its end. This gives us a tree F of topological height ℓ, and height $k\ell$. Since the initial instance is not solvable in time $2^{o(n+\log_2(b))}$ with $n \approx \log_2(b)$, we estimate

$$\log_2(b) \approx \log_2\left(\Delta^{k^\ell}\right) \approx k^\ell \log_2(\Delta) \approx \log_2(\Delta) \cdot \left(\tfrac{\mathrm{height}(F)}{\ell}\right)^\ell.$$

Let us perform the exact computations.

Proof of Theorem 2.60. Fix $\Delta \in \mathbb{Z}_{\geq 2}$ and $\ell \geq 2$. Let n_0 be large enough (implying that the choice of k will be large enough), and take a balanced subset sum instance with $n \geq n_0$ items. Define

$$k := \left\lfloor \sqrt[\ell]{log_\Delta(b)} \right\rfloor,$$

and let $1 \leq r \leq \ell$ be the unique integer such that

$$(k+1)^{r-1}k^{\ell-r+1} \leq \log_\Delta(b) < (k+1)^r k^{\ell-r}.$$

We will assume $r \leq \ell - 1$, and discuss the degenerate case in the end. Setting

$$\widetilde{\Delta} := \Delta^{(k+1)^{r-1}k^{\ell-r}},$$

this implies that $b < \widetilde{\Delta}^{k+1}$, and due to integrality, all occurring numbers are at most $\widetilde{\Delta}^{k+1} - 1$. By Lemma 2.58, we can encode this instance as a $(k+1)$-dimensional subset sum instance with $n' = n + k$ items and with the largest absolute value of any number bounded by $\widetilde{\Delta}$.

We further encode this $(k+1)$-dimensional subset sum instance as an IP (2.1) with constraints $(Y_i^{(j)})$ and (S_j). Let A denote the constraint matrix we obtain this way, and

$$L_\Delta := \left\lceil \log_\Delta \left(\widetilde{\Delta} + 1 \right) \right\rceil \leq \Delta^{(k+1)^{r-1}k^{\ell-r}} + 1.$$

For each i, the constraints $(Y_i^{(j)})$ induce a path Γ_i on L_Δ vertices as a subgraph in $G_D(A)$. We obtain n' paths in total, and there are no edges between distinct paths in $G_D(A)$. It remains to construct a td-decomposition containing these paths, and the Constraints (S_j).

Set $\kappa := \{k\}^r \times \{k-1\}^{\ell-r-1}$, and note that F_κ has $(k+1)^r k^{\ell-r-1} - 1 \geq L_\Delta$ vertices, for k large enough. By Lemma 2.53 F_κ is a td-decomposition for each subgraph Γ_i (possibly after adding dummy constraints). Take one copy F_i of F_κ for each i.

For the subgraph Σ induced by the Constraints (S_j), take a path on $k+1$ vertices to be a td-decomposition. Declare one of its endpoints to be the root and append the copies F_i to the other end. As all edges of $G_D(A)$ are either within the subgraph Σ, a subgraph Γ_i, or between Σ and one of the graphs G_i, the tree F we obtain this way is a td-decomposition for $G_D(A)$. We have $\mathrm{th}(F) = \ell$, its first level height is $k+1$, and the other level heights either k or $k-1$. In particular, its height is

$$d := (k+1) + rk + (\ell - r - 1)(k-1) = \ell k - (\ell - r - 2) \leq \ell k + 1, \tag{2.23}$$

where we used the fact that $r \leq \ell - 1$.

By Lemma 2.50, we cannot solve this instance faster than $2^{o(n+\log_2(b))}$, and we can estimate the exponent to

$$\begin{aligned} n + \log_2(b) \geq \frac{3}{2}\log_2(b) &\geq \frac{3}{2}\log_2(\Delta)\log_\Delta(b) \\ &\geq \frac{3}{2}\log_2(\Delta)(k+1)^{r-1}k^{\ell-r+1}. \end{aligned} \tag{2.24}$$

We now distinguish two cases. First, assume $r \geq \frac{2}{3}\ell - 2$, and let $\frac{1}{4} > \varepsilon > 0$. Choosing n_0

large enough so that k is large enough, we obtain the two estimates

$$(\tfrac{k}{k+1})^3 \geq \tfrac{4}{5}, \tag{2.25}$$

$$\begin{aligned}(k+1)^2k &= k^3 + 2k^2 + k \geq k^3 + (2-\varepsilon)k^2 + \tfrac{4}{3}k + \tfrac{8}{27} \\ &\geq (k + \tfrac{2-\varepsilon}{3})^3,\end{aligned} \tag{2.26}$$

and can compute

$$\begin{aligned}(k+1)^{r-1}k^{\ell-r+1} &= \left(\frac{k}{k+1}\right)^3 (k+1)^{r+2}k^{\ell-r-2} \\ &\geq \frac{4}{5}\left((k+1)^2k\right)^{\ell/3} && (\textit{with } (2.25)), \\ &\geq \frac{4}{5}\left(k + \tfrac{2-\varepsilon}{3}\right)^{\ell} && (\textit{with } (2.26)), \\ &\geq \frac{4}{5}\left(\tfrac{d}{\ell} - \tfrac{1}{\ell} + \tfrac{2-\varepsilon}{3}\right)^{\ell} && (\textit{with } (2.23)), \\ &> \frac{4}{5}\left(\frac{d}{\ell} + \tfrac{1}{12}\right)^{\ell}. && (\textit{with } \ell \geq 2, \varepsilon < 1/4).\end{aligned}$$

The second case, $r < \ell/3 - 2$, can only occur if $\ell \geq 9$, as $r \geq 1$ by definition. Then

$$\begin{aligned}(k+1)^{r-1}k^{\ell-r+1} &\geq \left(\frac{\ell k}{\ell}\right)^{\ell} \\ &= \left(\frac{\ell k - (\ell - r - 2)}{\ell} + \frac{(\ell - r - 2)}{\ell}\right)^{\ell} \\ &> \left(\frac{d}{\ell} + \frac{1}{3}\right)^{\ell},\end{aligned}$$

where we used the definition of d for the first term, and the estimate on r and ℓ for the second. In the end, we obtain that there is no algorithm solving every IP instance (2.1) that has a td_D-decomposition with height d and topological height at most ℓ in time

$$\begin{aligned}2^{o(n+\log_2 b)} &= 2^{o\left(\log_2(\Delta)(k+1)^{r-1}k^{\ell-r+1}\right)} \\ &= \Delta^{o\left(\left(\frac{d}{\ell}+\frac{1}{12}\right)^{\ell}\right)}.\end{aligned}$$

A generalized N-fold IP with coefficients r, s has a td_D-decomposition with topological height 2 and level heights r, s. Thus, the class of generalized n-fold IPs cannot be solved faster than $\|A\|_\infty^{o((r+s)^2)}$.

It is left to discuss the case

$$(k+1)^{\ell-1}k \leq \log_\Delta(b) < (k+1)^{\ell}.$$

The idea is to multiply the initial subset sum instance by some integer, changing our initial choice from k to $k+1$. We have $\Delta^{(k+1)^{\ell-1}k} \leq b$, and since

$$(k+1)^{\ell-1}k + k^{\ell-1} = (k+1)^{\ell},$$
$$(k+1)^{\ell} + k^{\ell-1} \leq (k+1)((k+1)+1)^{\ell-1} = (k+1)(k+2)^{\ell-1},$$

we can scale $b' := b \cdot \Delta^{k^{\ell-1}}$, and the new instance implies $1 \leq r \leq \ell - 1$. We multiplied each number in the instance by at most b, which is assumed to be the largest integer. The rest of the proof holds for the modified instance, up to the Estimate (2.24), where we use that the initial instance was balanced. However, this changes to

$$n + \log_2(b) \geq \tfrac{3}{2}\log_2(b) \geq \tfrac{3}{4}\log_2(b') \geq \tfrac{3}{4}\log_2(\Delta)(k+1)^{r-1}k^{\ell-r+1},$$

and as the constant $3/4$ vanishes in the Landau notation, the claim still holds. □

Let us compare the lower bound with the running time we obtain. To this end, consider an IP instance with td-decomposition F s.t. all level heights of F are k for some integer k, as was the case in the proof for the lower bound. Define $\ell := \mathrm{th}(F)$ and $d := \mathrm{height}(F)$. We obtain the estimate

$$\Delta^{\left(\frac{d}{\ell}\right)^{\ell}} \quad \leq \quad \text{true complexity} \quad \leq \quad \left(\Delta\left(\tfrac{d}{l}\right)^{\ell}\right)^{d\left(\frac{d}{l}\right)^{\ell}}.$$

If we plug in $d = m$, i.e. we assume $\ell = 1$, this recovers an interesting open problem for general integer linear programs: Given an ILP $Ax = b$, $l \leq x \leq u$, is the an algorithm with running time $(\|A\|_\infty m)^{\mathcal{O}(m)}n$? The best lower bound is $(\|A\|_\infty m)^{o(m)}n$ [KP], whereas the best known running time is $(\|A\|_\infty m)^{\mathcal{O}(m^2)}n$ [EW18]. However, if upper bounds are not present, the authors achieved a linear exponent, i.e. $(\|A\|_\infty m)^{o(m)}n$. This is remarkable, as we handle infinite bounds by introducing artificial bounds.

For n-fold ILPs, the discrepancy looks as follows. There is an algorithm with parameter dependency $(\Delta rs)^{r^2s+s^2}$ [JLR19], and we show that under the ETH, the dependency cannot be $\Delta^{o((r+s)^2)}$.

3 Compact representations of Voronoi cells of lattices

Introduction

Two widely investigated and important problems in the algorithmic geometry of numbers, cryptography, and integer programming are the shortest vector problem and the closest vector problem. Given a lattice Λ, the shortest vector problem (SVP) asks for a shortest non-zero vector in Λ. For a target vector $t \in \mathbb{R}^n$, the closest vector problem (CVP) asks for a lattice vector $z^\star$ minimizing the Euclidean distance $\|t - z\|$ from t to a lattice point $z \in \Lambda$.

Let us recall the milestones of the algorithmic development regarding both SVP and CVP. For a more detailed overview we refer to Hanrot, Pujol & Stehlé [HPS11], as well as to the more recent Gaussian Sampling approaches, for example, the one by Aggarwal & Stephens-Davidowitz [AS18].

In the 1980's, Kannan presented algorithms solving SVP and CVP in running time $n^{\mathcal{O}(n)}$ and polynomial space [Kan87]. Although the constants involved in the running time had been improved, it took roughly fifteen years until a significantly better algorithm was discovered. In 2001, Ajtai, Kumar & Sivakumar [AKS01] gave a randomized algorithm for the shortest vector problem, only taking $2^{\mathcal{O}(n)}$ time. However, in addition to the randomness, they also had to accept exponential space dependency for their improved running time. Though their algorithm is not applicable to the closest vector problem in its full generality, they show in a follow-up work that for any fixed ε, it can be used to approximate CVP up to a factor of $(1+\varepsilon)$ with running time depending on $1/\varepsilon$ [AKS02]. These authors posed the question whether randomness or exponential space is necessary for a running time better than $n^{\mathcal{O}(n)}$.

It took again around a decade until this question was partially answered by Micciancio & Voulgaris [MV13], who obtained a deterministic $2^{\mathcal{O}(n)}$ algorithm for both problems. Their algorithm is based on computing the Voronoi cell $\mathcal{V}_\Lambda$ of the lattice, the region of

all points at least as close to the origin as to any other lattice point. But as the Voronoi cell is a polytope with up to $2(2^n - 1)$ facets, the Micciancio-Voulgaris algorithm needs exponential space for storing the Voronoi cell in the worst (and generic) case. Since storing the Voronoi cell in a different, "more compact," way than by facet-description would lead to a decreased space requirement, they raise the question whether such a representation exists in general.

The main objective of this chapter is to propose such a compact representation of the Voronoi cell and to investigate its merits towards a single-exponential time and polynomial space algorithm for the CVP. As being closer to the origin than to a certain lattice vector v expresses in the inequality $2\,x^\intercal v \leq \|v\|^2$, the facets of $\mathcal{V}_\Lambda$ can be stored as a set $\mathcal{F}_\Lambda \subseteq \Lambda$ of lattice vectors, which are called the *Voronoi relevant vectors*. We say that a basis B of a lattice Λ is *c-compact*, if each Voronoi relevant vector of Λ can be represented in B with coefficients bounded by c in absolute value. Hence, by iterating over $(2c+1)^n$ vectors, we include the set $\mathcal{F}_\Lambda$. With $c(\Lambda)$, we denote the smallest c such that there exists a c-compact basis of Λ. As a consequence of the ideas in [MV13] and our notion of compactness we obtain (cf. Corollary 3.21):

(*i*) Given a c-compact basis of a lattice $\Lambda \subseteq \mathbb{R}^n$, we can solve the closest vector problem in $(2c+1)^{\mathcal{O}(n)}\,\mathrm{poly}(n)$ time and polynomial space.

Thus, the crucial question is: How small can we expect $c(\Lambda)$ to be for an arbitrary lattice? If $c(\Lambda)$ is constant, then (i) yields asymptotically the same running time as the initial Micciancio-Voulgaris algorithm, but uses only polynomial space. Of course, this only holds under the assumption that we know a c-compact basis of Λ. This observation has consequences for the variant of CVP with preprocessing, which we discuss in Section 3.7.

As an example of a large family of lattices, we prove in Section 3.4, that lattices whose Voronoi cell is a zonotope are as compact as possible:

(*ii*) If the Voronoi cell of Λ is a zonotope, then $c(\Lambda) = 1$. Moreover, a 1-compact basis can be found among the Voronoi relevant vectors.

Furthermore, every lattice of rank at most four has a 1-compact basis (cf. Corollary 3.16). However, starting with dimension five there are examples of lattices with $c(\Lambda) > 1$, and thus we want to understand how large this compactness constant can be in the worst case. Motivated by applications in crystallography, the desire for good upper bounds on $c(\Lambda)$ was already implicitly formulated in [Eng88; EMS01], and results of Seysen [Sey99] imply that $c(\Lambda) \in n^{\mathcal{O}(\log n)}$. We improve this to a polynomial bound and, on the negative side, we show that $c(\Lambda)$ may grow linearly with the dimension (Sections 3.2 & 3.3):

(*iii*) Every lattice possesses a basis that is n^2-compact.

(*iv*) There exists a family of lattices $(\Lambda_n)_{n\geq 5}$ without an $o(n)$-compact basis.

In Section 3.6, we relax the notion of a c-compact basis as follows. Denote by $\overline{c}(\Lambda)$ the smallest constant $\overline{c}$ such that there is *any* square matrix W with

$$\mathcal{F}_\Lambda \subseteq \{Wz : z \in \mathbb{Z}^n,\, \|z\|_\infty \leq \overline{c}\}.$$

Hence, in general, the matrix W generates a superlattice of Λ. This relaxation is motivated by the fact that, given a basis, membership to a lattice can be checked in polynomial time. Thus if $\overline{c}(\Lambda)$ is much smaller than $c(\Lambda)$, this additional check is faster than iterating over a larger set. Our results regarding the relaxed compactness constant include the following:

(*v*) For every lattice Λ, we have $\overline{c}(\Lambda) \in \mathcal{O}(n \log n)$.

(*vi*) There are lattices $\Lambda \subseteq \mathbb{R}^n$ with $c(\Lambda) \,/\, \overline{c}(\Lambda) \in \Omega(n)$.

In summary, the contributions of this chapter can be described as follows: If we are given a $c(\Lambda)$-compact basis of a lattice, then we can modify the algorithm of Micciancio & Voulgaris to obtain a polynomial space algorithm for CVP. In whole generality, the time complexity of this algorithm cannot be better than $n^{\mathcal{O}(n)}$, as in Kannan's work. However, we provide evidence that there are large and interesting classes of lattices, for which this improves to single-exponential time.

3.1 The notion of a c-compact basis

Given a lattice $\Lambda \subseteq \mathbb{R}^n$, recall its *Voronoi cell*

$$\mathcal{V}_\Lambda = \{x \in \mathbb{R}^n : \|x\| \leq \|x - z\| \text{ for all } z \in \Lambda\},$$

where $\|\cdot\|$ denotes the Euclidean norm. It consists of all points that are at least as close to the origin as to any other lattice point of Λ. The Voronoi cell turns out to be a centrally symmetric polytope having outer description $\mathcal{V}_\Lambda = \{x \in \mathbb{R}^n : 2\,x^\intercal z \leq \|z\|^2 \text{ for all } z \in \Lambda\}$. A vector $v \in \Lambda$ is called *weakly Voronoi relevant* if the corresponding inequality $2\,x^\intercal v \leq \|v\|^2$ defines a supporting hyperplane of $\mathcal{V}_\Lambda$, and it is called *strictly Voronoi relevant*, or simply *Voronoi relevant*, if it is moreover facet-defining. Let $\mathcal{F}_\Lambda$ and $\mathcal{C}_\Lambda$ be the set of strictly and weakly Voronoi relevant vectors of Λ, respectively. The central definition of this chapter is the following.

Definition 3.1. *Let $\Lambda \subseteq \mathbb{R}^n$ be a lattice and let $c \in \mathbb{N}$. A basis B of Λ is called* c-compact*, if*

$$\mathcal{F}_\Lambda \subseteq \{Bz : z \in \mathbb{Z}^n,\ \|z\|_\infty \leq c\}.$$

That is, each Voronoi relevant vector is a linear combination of the basis vectors with coefficients bounded by c in absolute value. Moreover, we define

$$c(\Lambda) = \min\{c \geq 0 : \Lambda \textit{ possesses a } c\textit{-compact basis}\}$$

as the compactness constant *of* Λ.

As discussed in the introduction, the notion of a c-compact basis provides a compact representation of the Voronoi cell $\mathcal{V}_\Lambda$, the complexity of which depends on the value of the constant c. Before we set out to study the compactness constant in detail, we offer various equivalent definitions that serve as auxiliary tools and that also provide a better understanding of the underlying concept.

To this end, recall the dual lattice $\Lambda^\star = \{y \in \mathbb{R}^n : y^\intercal z \in \mathbb{Z} \text{ for all } z \in \Lambda\}$ of a lattice Λ, and the polar $K^\star = \{x \in \mathbb{R}^n : x^\intercal y \leq 1 \text{ for all } y \in K\}$ of a convex body $K \subseteq \mathbb{R}^n$ containing the origin in its interior (cf. Section 1.4). The basic properties we need are the following: If B is a basis of Λ, then $B^{-\intercal} := (B^{-1})^\intercal$ is a basis of $\Lambda^\star$, usually called the *dual basis* of B. For a matrix $A \in \mathrm{GL}_n(\mathbb{R})$ and a compact convex set K as above, we have $(AK)^\star = A^{-\intercal}K^\star$. We refer to Gruber's textbook [Gru07] for details and more information on these concepts.

Lemma 3.2. *Let* $B = \{b_1, \ldots, b_n\}$ *be a basis of a lattice* $\Lambda \subseteq \mathbb{R}^n$. *The following are equivalent:*

i) B *is* c*-compact,*

ii) $c \cdot \mathrm{conv}(\mathcal{F}_\Lambda)^\star$ *contains the dual basis* $B^{-\intercal}$ *of* $\Lambda^\star$,

iii) writing $B^{-\intercal} = \{b_1^\star, \ldots, b_n^\star\}$, *we have*

$$\mathcal{F}_\Lambda \subseteq \{x \in \Lambda : |x^\intercal b_i^\star| \leq c \textit{ for all } 1 \leq i \leq n\},$$

iv) $\mathcal{F}_\Lambda \subseteq c\,P_B$, *where* $P_B = \sum_{i=1}^n [-b_i, b_i]$.

Proof. *i)* $\iff$ *ii)*: By definition, B is c-compact if and only if $\mathcal{F}_\Lambda \subseteq \{Bz : z \in \mathbb{Z}^n, \|z\|_\infty \leq c\}$. This means that $Q = \mathrm{conv}(\mathcal{F}_\Lambda) \subseteq B[-c, c]^n$. Taking polars, we see that this is equivalent to $B^{-\intercal}\frac{1}{c}C_n^\star \subseteq Q^\star$, where $C_n^\star = \mathrm{conv}\{\pm e_1, \ldots, \pm e_n\}$ is the standard crosspolytope. Since the columns of $B^{-\intercal}$ constitute a basis of the dual lattice $\Lambda^\star$, the proof is finished.

i) $\iff$ *iii)*: $B = \{b_1, \ldots, b_n\}$ is c-compact if and only if the representation $v = \sum_{i=1}^n \alpha_i b_i$ of any Voronoi relevant vector $v \in \mathcal{F}_\Lambda$ satisfies $|\alpha_i| \leq c$, for all $1 \leq i \leq n$. By the definition of the dual basis, we have $\alpha_i = v^\intercal b_i^\star$, which proves the claim.

i) $\iff$ *iv)*: By definition, $\mathcal{F}_\Lambda \subseteq c\,P_B$ if and only if for every $v \in \mathcal{F}_\Lambda$, there are coefficients $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ such that $v = \sum_{i=1}^n \alpha_i b_i$ and $|\alpha_i| \leq c$. These coefficients are unique, and

since B is a basis of Λ, they are integral, that is $\alpha_i \in \mathbb{Z}$. Thus, the inclusion we started with is equivalent to saying that B is c-compact. □

Part iv) of the above lemma shows that the compactness constant $c(\Lambda)$ is the minimum c such that $\mathcal{F}_\Lambda \subseteq c\, P_B$, for some basis B of Λ. In this definition, the concept has been introduced already by Engel, Michel & Senechal [EMS01] together with the variant $\chi(\Lambda)$, where one replaces $\mathcal{F}_\Lambda$ by the larger set $\mathcal{C}_\Lambda$ of weakly Voronoi relevant vectors. Motivated by applications in crystallography, a reoccurring question posed in [Eng88; EMS01] is to give good upper bounds on these lattice invariants $c(\Lambda)$ and $\chi(\Lambda)$. Results of Seysen [Sey99] on simultaneous lattice reduction of the primal and dual lattice imply that

$$c(\Lambda) \leq \chi(\Lambda) \in n^{\mathcal{O}(\log n)}. \tag{3.1}$$

This is however the only bound that we are aware of.

3.2 A polynomial upper bound

In the sequel, we occasionally need Minkowski's *successive minima* of a convex body K and a lattice Λ in $\mathbb{R}^n$. For $1 \leq i \leq n$, the ith successive minimum is defined as

$$\lambda_i(K, \Lambda) = \min\{\lambda \geq 0 : \lambda K \text{ contains } i \text{ linearly independent points of } \Lambda\}.$$

Minkowski's development of his geometry of numbers was centered around the study of these lattice parameters (we refer to Gruber's handbook [Gru07] for background). With this notion, Lemma 3.2 ii) provides a lower bound on the compactness constant of a given lattice. Indeed, we have

$$c(\Lambda) \geq \lambda_n(Q^\star, \Lambda^\star),$$

where $Q = \operatorname{conv}(\mathcal{F}_\Lambda)$.

Our first result aims for an explicit upper bound on $c(\Lambda)$ only depending on the dimension of the lattice. To this end, we first need an auxiliary result.

Lemma 3.3. *For a lattice $\Lambda \subseteq \mathbb{R}^n$ with Voronoi cell $\mathcal{V}_\Lambda$ holds $\lambda_1(\mathcal{V}_\Lambda^\star, \Lambda^\star) \leq \frac{2}{\pi} n$. Hence, there exists a dual lattice vector $y^\star \in \Lambda^\star$ such that*

$$\mathcal{V}_\Lambda \subseteq \left\{x \in \mathbb{R}^n : |x^\intercal y^\star| \leq \tfrac{2}{\pi} n\right\}.$$

Proof. By Minkowski's fundamental theorem (cf. Theorem 1.3), we have

$$\lambda_1(\mathcal{V}_\Lambda, \Lambda) \leq 2\left(\frac{\det(\Lambda)}{\mathrm{vol}(\mathcal{V}_\Lambda)}\right)^{\frac{1}{n}} \quad \text{and} \quad \lambda_1(\mathcal{V}_\Lambda^\star, \Lambda^\star) \leq 2\left(\frac{\det(\Lambda^\star)}{\mathrm{vol}(\mathcal{V}_\Lambda^\star)}\right)^{\frac{1}{n}}.$$

Moreover, by a result of Kuperberg [Kup08, Cor. 1.6], $\mathrm{vol}(K)\,\mathrm{vol}(K^\star) \geq \pi^n/n!$, for every centrally symmetric convex body $K \subseteq \mathbb{R}^n$. Therefore,

$$\lambda_1(\mathcal{V}_\Lambda, \Lambda)\lambda_1(\mathcal{V}_\Lambda^\star, \Lambda^\star) \leq 4\left(\frac{\det(\Lambda)\det(\Lambda^\star)}{\mathrm{vol}(\mathcal{V}_\Lambda)\,\mathrm{vol}(\mathcal{V}_\Lambda^\star)}\right)^{\frac{1}{n}} \leq 4\left(\frac{n!}{\pi^n}\right)^{\frac{1}{n}} \leq \frac{4}{\pi}n,$$

since $\det(\Lambda)\det(\Lambda^\star) = 1$ (cf. [Mar03, Ch. 1]). The claimed bound now follows as $\lambda_i(\mathcal{V}_\Lambda, \Lambda) = 2$, for all $1 \leq i \leq n$. □

Theorem 3.4. *For every lattice $\Lambda \subseteq \mathbb{R}^n$, there exists an n^2-compact basis.*

Proof. We prove by induction on the dimension that there is a basis $D = \{y_1, \ldots, y_n\}$ of $\Lambda^\star$ such that

$$\mathcal{V}_\Lambda \subseteq \left\{x \in \mathbb{R}^n : |x^\intercal y_i| \leq \tfrac{1}{2}n^2,\ 1 \leq i \leq n\right\}. \tag{3.2}$$

Since every Voronoi relevant vector lies in the boundary of $2\mathcal{V}_\Lambda$, its inner product with each y_i is then bounded by n^2. Hence, the basis of Λ that is dual to D is an n^2-compact basis by Lemma 3.2 iii).

If $n = 1$, the containment (3.2) is trivially true, hence let $n \geq 2$. Let y_1 be a shortest vector of $\Lambda^\star$ with respect to the norm $\|\cdot\|_{\mathcal{V}_\Lambda^\star}$. By Lemma 3.3, we have $\mathcal{V}_\Lambda \subseteq \left\{x \in \mathbb{R}^n : |x^\intercal y_1| \leq \frac{2n}{\pi}\right\}$. Let $\Lambda' = \Lambda \cap \{x \in \mathbb{R}^n : x^\intercal y_1 = 0\}$, and observe that the orthogonal projection $\pi : \mathbb{R}^n \to \{x \in \mathbb{R}^n : x^\intercal y_1 = 0\}$ fulfills $\pi(\Lambda^\star) = (\Lambda')^\star$, where we dualize with respect to the linear span of Λ' (cf. [Mar03, Ch. 1]). By induction hypothesis, there is a basis $D' = \{y_2', \ldots, y_n'\}$ of $(\Lambda')^\star$, such that

$$\mathcal{V}_{\Lambda'} \subseteq \left\{x \in \mathbb{R}^n : x^\intercal y_1 = 0 \text{ and } |x^\intercal y_i'| \leq \tfrac{1}{2}(n-1)^2,\ 2 \leq i \leq n\right\}.$$

As $\Lambda' \subseteq \Lambda$, we have $\mathcal{V}_\Lambda \subseteq \mathcal{V}_{\Lambda'} + \mathrm{lin}\{y_1\}$. Moreover, as $(\Lambda')^\star$ is the projection of $\Lambda^\star$ along y_1, there exist $\alpha_i \in [-1/2, 1/2)$ such that $y_i = y_i' + \alpha_i y_1 \in \Lambda^\star$ for $2 \leq i \leq n$, and $D = \{y_1, \ldots, y_n\}$ is a basis of $\Lambda^\star$. Hence,

$$\begin{aligned}
\mathcal{V}_\Lambda &\subseteq \left\{x \in \mathbb{R}^n : |x^\intercal y_1| \leq \tfrac{2n}{\pi},\ |x^\intercal y_i'| \leq \tfrac{1}{2}(n-1)^2,\ 2 \leq i \leq n\right\} \\
&\subseteq \left\{x \in \mathbb{R}^n : |x^\intercal y_1| \leq \tfrac{2n}{\pi},\ |x^\intercal y_i| \leq \tfrac{1}{2}(n-1)^2 + \tfrac{n}{\pi},\ 2 \leq i \leq n\right\}. \\
&\subseteq \left\{x \in \mathbb{R}^n : |x^\intercal y_i| \leq \tfrac{1}{2}n^2,\ 1 \leq i \leq n\right\},
\end{aligned}$$

finishing the proof. □

Remark 3.5. *Since also the weakly Voronoi relevant vectors $\mathcal{C}_\Lambda$ lie in the boundary of $2\mathcal{V}_\Lambda$, the basis from the previous proof also shows $\chi(\Lambda) \leq n^2$, for every lattice $\Lambda \subseteq \mathbb{R}^n$ (compare with Inequality* (3.1)*).*

Let us look at the constant $c(\Lambda)$ from a different angle. A basis of a lattice is particularly nice if each Voronoi relevant vector is a $\{-1, 0, 1\}$-combination of the basis vectors. As not every lattice possesses such a basis (see Proposition 3.10 below), we relaxed the condition on the coefficients and introduced the lattice parameter $c(\Lambda)$, defined for all lattices. Another way to relax the setting above is not to insist on a basis of Λ, but rather to look for a generating set S such that each Voronoi relevant vector can be written as a $\{-1, 0, 1\}$-combination of the vectors in S. In this setting, we are interested in finding a small set S. Such an S of order $n \log n$ can be retrieved from an n^2-compact basis.

Corollary 3.6. *For every lattice $\Lambda \subseteq \mathbb{R}^n$ there exists a subset $S \subseteq \Lambda$ of cardinality $\mathcal{O}(n \log n)$ such that*

$$\mathcal{F}_\Lambda \subseteq \left\{ \sum_{s \in S} \sigma_s \, s : \sigma_s \in \{-1, 0, 1\}, \text{ for } s \in S \right\}.$$

Proof. By Theorem 3.4, there exists a c-compact basis B of Λ with $c \leq n^2$. Let $M := \lfloor \log_2 c \rfloor$. Each $0 \leq \alpha \leq c$ can be written as $\alpha = \sum_{j=0}^{M} 2^j \sigma_j$, for some unique $\sigma_j \in \{0, 1\}$. For each vector $b_i \in B$ and $0 \leq j \leq M$, we define the vector

$$s_{i,j} := 2^j b_i.$$

This gives $\mathcal{O}(n \log_2(n^2)) = \mathcal{O}(n \log n)$ vectors in total, and clearly every vector $v = \sum_{i=1}^{n} \alpha_i b_i$ with $|\alpha_i| \leq c$ can be written as a linear combination of the $s_{i,j}$ using only coefficients in $\{-1, 0, 1\}$. □

Remark 3.7. *With a different method Daniel Dadush (personal communication) proves that the subset S can be chosen to consist of* Voronoi relevant vectors *itself.*

3.3 Lattices without sublinearly-compact bases

In this part, we identify an explicit family of lattices whose compactness constant grows at least linearly with the dimension. This requires some technical work; the pure existence of such a family also follows from Proposition 3.19 iii) below. However, based on the understanding of the lattice discussed in this section, we are able to discriminate between the compactness constant and a relaxed variant, which will be introduced in the next section.

For any $a \in \mathbb{N}$ and $n \in \mathbb{N}$, we define the lattice

$$\Lambda_n(a) = \{z \in \mathbb{Z}^n : \ z_1 \equiv \cdots \equiv z_n \mod a\}, \tag{3.3}$$

whose dual lattice is given by

$$\Lambda_n(a)^\star = \left\{z \in \tfrac{1}{a}\mathbb{Z}^n : \mathbf{1}^\intercal z \in \mathbb{Z}\right\}, \tag{3.4}$$

where $\mathbf{1} = (1, \ldots, 1)^\intercal$ denotes the all-one vector. The special structure of these lattices allows us to write down the Voronoi relevant vectors explicitly.

Lemma 3.8. *Let $n \in \mathbb{N}_{\geq 4}$, $a = \lceil n/2 \rceil$, and write $\Lambda_n = \Lambda_n(a)$. Then, a vector $v \in \Lambda_n$ is strictly Voronoi relevant if and only if either $v = \pm\mathbf{1}$, or there exists an index set $\emptyset \neq S \subsetneq \{1, \ldots, n\}$ such that*

$$v_i = \begin{cases} a - \ell & i \in S \\ -\ell & i \notin S \end{cases}, \qquad \text{and} \quad \ell \in \left\{ \left\lfloor \frac{a|S|}{n} \right\rfloor, \left\lceil \frac{a|S|}{n} \right\rceil \right\}. \tag{3.5}$$

Proof. Let us first discuss the vectors $\pm\mathbf{1}$. They have squared norm n, and if there is a shorter vector v, it must contain zero coordinates. But due to the definition of Λ_n, all its coordinates are then multiples of a, so it has squared norm at least $a^2 \geq n^2/4 \geq n$ for $n \geq 4$. Hence, $\pm\mathbf{1}$ are shortest vectors of the lattice and therefore always strictly Voronoi relevant. As we are only interested in the strictly Voronoi relevant vectors in this proof, we will omit the word "strictly" henceforth.

Voronoi characterized a Voronoi relevant vector v in a lattice Λ by the property that $\pm v$ are the only shortest vectors in the co-set $v + 2\Lambda$ (cf. [CS99, p. 477]). We use this crucially to show that Voronoi relevant vectors different from $\pm\mathbf{1}$ are characterized by (3.5).

v Voronoi relevant $\Rightarrow$ v of Shape (3.5): Let $v \neq \pm\mathbf{1}$ be Voronoi relevant. We have $v \in [-a, a]^n$, as $2a\, e_i \in 2\Lambda_n$ otherwise implies that v is not a shortest vector in its co-set $v + 2\Lambda_n$. Let us first assume that there is an index i such that $v_i \in \{0, \pm a\}$. By definition of Λ_n, we have $v_i \equiv v_j \mod a$, for all j, hence $v \in \{0, \pm a\}^n$. If v has at least two non-zero coordinates, let $\widetilde{v}$ arise from v by changing the sign of exactly one of them. Observe that $\widetilde{v}$ is linearly independent from v, has the same length, and is contained in $v + 2\Lambda_n$. This contradicts the assumption that v was Voronoi relevant. If v has only one non-zero entry, say $v_j \neq 0$, then it is of Shape (3.5). Indeed, we can either take $S = \{j\}$ and $\ell = 0$, or $S = \{1, \ldots, n\} \setminus \{j\}$ and $\ell = \lceil a(n-1)/n \rceil = a$.

This leaves us with the case $v \in [-(a-1), a-1]^n$. Again, by the definition of Λ_n, there is an integer $1 \leq r \leq a - 1$ such that $v \in \{a - r, -r\}^n$. Let k be the number of entries of v that are equal to $a - r$. Note that $1 \leq k \leq n - 1$ as otherwise $v = \pm\mathbf{1}$. For the norm of v, we obtain

$$||v||^2 = nr^2 - 2akr + ka^2.$$

Seen as a rational quadratic function in r, it is minimized for $r' = ak/n$. As increasing or decreasing r by 2 corresponds to adding or subtracting $2 \cdot \mathbf{1} \in 2\Lambda_n$ to v, we must have $r \in [r' - 1, r' + 1]$. If r' is not integral, this corresponds to $r \in \{\lceil ak/n \rceil, \lfloor ak/n \rfloor\}$. If r' is integral, observe that $r = r' \pm 1$ corresponds to two linearly independent vectors in the same co-set and of the same length, hence again $r = r' \in \{\lceil ak/n \rceil, \lfloor ak/n \rfloor\}$, so that v is indeed of Shape (3.5).

v of Shape (3.5) $\Rightarrow$ v Voronoi relevant: For the other direction, let v be a vector of Shape (3.5) with index set S and parameter ℓ. Let $u \in v + 2\Lambda_n$ be a shortest vector within the co-set $v + 2\Lambda_n$. We claim that $u = \pm v$, which will prove that v is Voronoi relevant. To this end, recall from above that necessarily $u \in [-a, a]^n$. Moreover, as $u - v \in 2\Lambda_n$, we have $v_i - v_j \equiv u_i - u_j \mod 2a$. Therefore, if there are indices $i \neq j$ such that $v_i = v_j$, then we have $u_i \equiv u_j \mod 2a$. Unless we are in the extreme case $u \in \{0, \pm a\}^n$ (see Case (a)), this even implies $u_i = u_j$ (see Case (b)).

Case (a): We make a second case distinction depending on the number of non-zero entries of u. This number is always either equal to $|S|$ or $n - |S|$.

Note that the case of u having exactly 1 non-zero entry (i.e. $|S| \in \{1, n-1\}$) will be covered by Case (b) below.

If u has at least 3 non-zero entries ($|S| \in \{3, 4, \ldots, n-3\}$), observe that the vector $u' = (u'_1, \ldots, u'_n)^\intercal$ defined by $u'_i = |u_i| - 2$ is in the same co-set, but also shorter than u, a contradiction.

For the last case, u having two non-zero entries, the vector u' as above is only strictly shorter if n is odd. If n is even however, u and u' will have the same norm. In this particular case, observe that $a|S|/n \in \{1, a-1\}$, hence $\ell = a|S|/n$, as we do not round. But this is a contradiction, as u and v differ by $\mathbf{1} \notin 2\Lambda_n$, that is, they are not in the same co-set.

Case (b): Henceforth, whenever $v_i = v_j$, we have $u_i = u_j$. By possibly switching to $u' = -u$, we can assume that for some $0 \leq r \leq a$, $u_i = a - r$ for $i \in S$ and $u_j = -r$ for $j \notin S$. This is, $u_i = a - r$ whenever $v_i = a - \ell$ and $u_j = -r$ whenever $v_j = -\ell$. For the norm of u, we obtain

$$\|u\|^2 = |S|(a-r)^2 + (n - |S|)r^2 = nr^2 - 2ar|S| + |S|a^2.$$

Seen as a rational quadratic function in r, this term is uniquely minimized for $\hat{r} = a|S|/n$. Observe that there may be two choices for ℓ, $\lfloor \hat{r} \rfloor, \lceil \hat{r} \rceil$. It is clear that r also has to be one of these values, as otherwise u is not a shortest vector in its co-set. But observe that the two choices lead to two vectors whose difference is $\mathbf{1} \notin 2\Lambda_n$. As u and v have to be in the same co-set, we have $u = \pm v$, since we may have switched to $-u$ in the beginning. $\square$

Theorem 3.9. *Let $n \in \mathbb{N}_{\geq 4}$, $a = \lceil n/2 \rceil$. Then, the lattice $\Lambda_n = \Lambda_n(a)$ has compactness constant $c(\Lambda_n) \geq \lceil \frac{n}{4} \rceil$.*

Proof. For brevity, we write $c = c(\Lambda_n)$, $Q = \mathrm{conv}(\mathcal{F}_{\Lambda_n})$. As $\mathbf{1} \in \Lambda_n$, there exists a $w \in \Lambda_n^\star$ with $\mathbf{1}^\intercal w = 1$, for instance, take $w = e_1$. This implies that each basis of $\Lambda_n^\star$ contains a vector y such that $\mathbf{1}^\intercal y$ is an odd integer. In particular, using the characterization of Lemma 3.2, we know that $cQ^\star$ has to contain such a y. As $Q^\star$ is centrally symmetric, assume $\mathbf{1}^\intercal y \geq 1$. Further, since $\Lambda_n^\star$ is invariant under permutation of the coordinates, assume the entries of y are ordered non-increasingly,

$$y_1 \geq y_2 \geq \cdots \geq y_n. \tag{3.6}$$

Let us outline our arguments first: We split $\mathbf{1}^\intercal y$ into two parts, by setting $A := \sum_{i=1}^{k} y_i$, and $B := \sum_{i>k}^{n} y_i$, where $k = \lceil n/2 \rceil$. We show that $A \geq B + 1$, and construct a Voronoi relevant vector $v \in \Lambda_n$ whose first k entries are roughly $n/4$, and its last $n-k$ entries are roughly $-n/4$ by using Lemma 3.8 and choosing $S = \{1, \ldots, k\}$, $\ell = \lfloor ak/n \rfloor = \lfloor a^2/n \rfloor$. We then obtain $v^\intercal y \approx \frac{n}{4}A - \frac{n}{4}B \geq n/4$ by carefully distinguishing the four cases $n \bmod 4$.

For showing $A \geq B + 1$, consider y_k. As $y \in \Lambda_n^\star$, there is an integer z such that we can write $y_k = \frac{z}{a}$. We can assume $z > 0$, since otherwise $B \leq 0$ and we are done since $A + B = \mathbf{1}^\intercal y \geq 1$. Note that we have $A \geq k y_k = z$ and $B \leq (n-k)\frac{z}{a} \leq z$ by (3.6). Let $\alpha, \gamma \geq 0$ such that $A = z + \alpha$ and $B = z - \gamma$. As $A + B = 2z + \alpha - \gamma$ has to be an odd integer, we have $|\alpha - \gamma| \geq 1$, implying $\alpha \geq 1$ or $\gamma \geq 1$. Therefore, in fact we have $A \geq \max\{B + 1, 1\}$.

Using this inequality and carefully evaluating $v^\intercal y = (a - \ell)A - \ell B$ for the four cases $n \bmod 4$, the claim follows.

Recall that we construct the Voronoi relevant vector v by choosing $k = a = \lceil n/2 \rceil$, $S = \{1, \ldots, k\}$, $\ell = \lfloor ak/n \rfloor = \lfloor a^2/n \rfloor$, and applying Lemma 3.8.

We obtain $v^\intercal y = (a - \ell)A - \ell B$, and are ready to distinguish the four cases $n \bmod 4$.

1. $n = 4m$. Hence, we have $a = k = 2m$, and $\ell = m$. Thus,

$$v^\intercal y = (a - \ell)A - \ell B = m(A - B) \geq m = n/4.$$

2. $n = 4m + 1$. Hence, we have $a = k = 2m + 1$, and $\ell = m$. Thus,

$$v^\intercal y = (a - \ell)A - \ell B = m(A - B) + A \geq m + 1 \geq n/4.$$

3. $n = 4m + 2$. Hence, we have $a = k = 2m + 1$, and $\ell = m$. Thus,

$$v^\intercal y = (a - \ell)A - \ell B = m(A - B) + A \geq m + 1 \geq n/4.$$

4. $n = 4m + 3$. Hence, we have $a = k = 2m + 2$, and $\ell = m + 1$. Thus,

$$v^\intercal y = (a - \ell)A - \ell B = (m + 1)(A - B) \geq m + 1 \geq n/4.$$

As the constant c is integral, the claim follows. □

3.4 Compact bases and zonotopal lattices

For the sake of brevity, we call a 1-compact basis of a lattice just a *compact basis*. A class of lattices that allow for a compact representation of their Voronoi cells are the lattices of *Voronoi's first kind*. They correspond to those lattices Λ that constitute the first reduction domain in Voronoi's reduction theory (see [Val03; Vor08]). These lattices have been characterized in [CS92] by possessing an *obtuse superbasis*, which is a set of vectors $\{b_0, \ldots, b_n\} \subseteq \Lambda$ that generates Λ, and that fulfills the superbasis condition $b_0 + \ldots + b_n = 0$ and the obtuseness condition $b_i^\intercal b_j \leq 0$, for all $i \neq j$. Given an obtuse superbasis, for each Voronoi relevant vector $v \in \Lambda$ there is a strict non-empty subset $S \subseteq \{0, 1, \ldots, n\}$ such that $v = \sum_{i \in S} b_i$.

Let us compare lattices of Voronoi's first kind with lattices possessing a compact basis.

Proposition 3.10.

i) Every lattice of Voronoi's first kind has a compact basis.

ii) Every lattice of rank at most three has a compact basis.

iii) For $n \geq 4$, the checkerboard lattice $D_n = \{x \in \mathbb{Z}^n : \mathbf{1}^\intercal x \in 2\mathbb{Z}\}$ *is not of Voronoi's first kind, but has a compact basis.*

iv) There exists a lattice $\Lambda \subseteq \mathbb{R}^5$ with $c(\Lambda) \geq 2$.

Proof. i): Every obtuse superbasis contains in fact a compact basis. Indeed, using the representation of a Voronoi relevant vector above and writing $b_0 = -\sum_{i=1}^n b_i$, we get $v = \sum_{i \in S} b_i = -\sum_{i \notin S} b_i$. One of the terms does not use b_0.

ii): Every lattice of dimension at most three is of Voronoi's first kind (cf. [CS92]), so part i) applies.

iii): Bost & Künnemann [BK10, Prop. B.2.6] showed that for $n \geq 4$, the lattice D_n is not of Voronoi's first kind. One can easily verify that the set $B = \{b_1, \ldots, b_n\}$ with

$b_1 = e_1 + e_n$, and $b_i = e_i - e_{i-1}$ for $2 \leq i \leq n$, is a basis of D_n. Observing that the vectors $2e_i \pm 2e_j$ are in $2D_n$ for all i, j, a vector v that is the unique (up to sign) shortest vector in the co-set $v + 2\Lambda$, must be of the form $\{\pm(e_i \pm e_j) : 1 \leq i < j \leq n\}$. A routine calculation shows that all these vectors are a $\{-1, 0, 1\}$-combination of the basis B.

iv): This follows from Theorem 3.9 with the lattice $\Lambda_5(3)$. □

We now explore to which extent these initial observations on lattices with compact bases can be generalized.

A *zonotope* Z in $\mathbb{R}^n$ is a Minkowski sum of finitely many line segments, that is, $Z = \sum_{i=1}^{r}[a_i, b_i]$, for some $a_i, b_i \in \mathbb{R}^n$. The vectors $b_1 - a_1, \ldots, b_r - a_r$ are usually called the *generators* of Z. We call a lattice *zonotopal* if its Voronoi cell is a zonotope. A generic zonotopal lattice has typically high combinatorial complexity. An explicit example is the root lattice $A_n^\star$; its zonotopal Voronoi cell is generated by $\binom{n+1}{2}$ vectors and it has exactly the maximum possible $2(2^n - 1)$ facets (cf. [CS99, Ch. 4 & Ch. 21]). However, not every generic lattice is zonotopal. For instance, a perturbation of the E_8 root lattice gives a generic non-zonotopal lattice (cf. [ER94, Sect. 4]).

It turns out that every lattice of Voronoi's first kind is zonotopal, but starting from dimension four, the class of zonotopal lattices is much richer (cf. Vallentin's thesis [Val03, Ch. 2] and [ER94]). In the following, we prove that every zonotopal lattice possesses a compact basis, thus extending Proposition 3.10 *i*) significantly.

Our proof relies on the beautiful work of Erdahl [Erd99] who unraveled an intimate relationship between zonotopal lattices and so-called dicings. A *dicing* $\mathfrak{D}$ in $\mathbb{R}^n$ is an arrangement of hyperplanes consisting of at least n families of infinitely many equally-spaced hyperplanes with the following properties:

i) There are n families with linearly independent normal vectors.

ii) Every vertex of $\mathfrak{D}$ is contained in a hyperplane of each family.

The interesting cases are those with more than n families of hyperplanes.

It turns out that the vertex set of a dicing forms a lattice, denoted by $\Lambda(\mathfrak{D})$. Indeed, the vertex set induced by the n linearly independent families forms a lattice, and because of property (ii) no additional vertices are introduced by the remaining families. A basis of the lattice $\Lambda(\mathfrak{D})$ may be obtained from taking the inverse of the matrix whose rows are n linearly independent normal vectors appropriately scaled (they exist by property (i)).

Erdahl [Erd99, Thm. 3.1] shows that a dicing $\mathfrak{D}$ can be represented by a set $D = \{\pm d_1, \ldots, \pm d_r\}$ of hyperplane normals and a set $E = \{\pm e_1, \ldots, \pm e_s\} \subseteq \Lambda(\mathfrak{D})$ of edge vectors of the arrangement $\mathfrak{D}$ satisfying:

E1) Each pair of edges $\pm e_j \in E$ is contained in a line $d_{i_1}^{\perp} \cap \ldots \cap d_{i_{n-1}}^{\perp}$, for some linearly independent $d_{i_1}, \ldots, d_{i_{n-1}} \in D$, and conversely each such line contains a pair of edges.

E2) For each $1 \leq i \leq r$ and $1 \leq j \leq s$, we have $d_i^{\intercal} e_j \in \{0, \pm 1\}$.

For clarity we denote the dicing by $\mathfrak{D} = \mathfrak{D}(D, E)$.

Theorem 3.11. *Every zonotopal lattice has a compact basis. It can be found among its Voronoi relevant vectors.*

Proof. We start by reviewing the *Delaunay tiling* of a lattice Λ. A sphere $B_c(R) = \{x \in \mathbb{R}^n : \|x - c\|^2 \leq R^2\}$ is called an *empty sphere* of Λ (with center $c \in \mathbb{R}^n$ and radius $R \geq 0$), if every point in $B_c(R) \cap \Lambda$ lies on the boundary of $B_c(R)$. A *Delaunay polytope* of Λ is defined as the convex hull of $B_c(R) \cap \Lambda$, where $B_c(R)$ is an empty sphere. The family of all Delaunay polytopes induces a tiling $\mathcal{D}_\Lambda$ of $\mathbb{R}^n$ which is the Delaunay tiling of Λ. This tiling is in fact dual to the Voronoi tiling.

Erdahl [Erd99, Thm. 2] shows that the Voronoi cell of a lattice is a zonotope if and only if its Delaunay tiling is a dicing. More precisely, the tiling $\mathcal{D}_\Lambda$ induced by the Delaunay polytopes of Λ is equal to the tiling induced by the hyperplane arrangement of a dicing $\mathfrak{D} = \mathfrak{D}(D, E)$ with normals $D = \{\pm d_1, \ldots, \pm d_r\}$ and edge vectors $E = \{\pm e_1, \ldots, \pm e_s\}$. By the duality of the Delaunay and the Voronoi tiling, an edge of $\mathcal{D}_\Lambda$ containing the origin corresponds to a facet normal of the Voronoi cell $\mathcal{V}_\Lambda$. Therefore, the edge vectors E are precisely the Voronoi relevant vectors of Λ.

Now, choosing n linearly independent normal vectors, say $d_1, \ldots, d_n \in D$, the properties E1) and E2) imply the existence of edge vectors, say $e_1, \ldots, e_n \in E$, such that $d_i^{\intercal} e_j = \delta_{ij}$, with δ_{ij} being the Kronecker delta. Moreover, the set $B = \{e_1, \ldots, e_n\}$ is a basis of $\{x \in \mathbb{R}^n : d_i^{\intercal} x \in \mathbb{Z}, 1 \leq i \leq n\}$, which by property E2) equals the whole lattice Λ. Hence, $\{d_1, \ldots, d_n\}$ is the dual basis of B and every Voronoi relevant vector $v \in \mathcal{F}_\Lambda = E$ fulfills $d_i^{\intercal} v \in \{0, \pm 1\}$. In view of Lemma 3.2 *iii*), this means that B is a compact basis of Λ consisting of Voronoi relevant vectors, as desired. □

3.5 Compact bases in small dimensions

We have seen in Proposition 3.10 that every lattice of rank at most three has a compact basis, and that there are five-dimensional lattices without compact bases. In the sequel we complete the picture and show that every four-dimensional lattice admits a compact basis as well.

Our argument uses tools from the theory of parallelotopes which requires to set up the compactness constant in this more general framework. For details and background on

the following definitions and statements on parallelotopes we refer to [Gru07, §32]. A *parallelotope* (also called parallelohedron) is a convex polytope $P \subseteq \mathbb{R}^n$ that admits a facet-to-facet tiling of $\mathbb{R}^n$ by translations. Voronoi cells of lattices are prime examples of parallelotopes. Every parallelotope is centrally symmetric, and we may assume that its center of symmetry is at the origin. The set of translation vectors that constitute the facet-to-facet tiling by copies of P is in fact a lattice, and we denote it by $\Lambda(P)$. Every facet F of P corresponds to a lattice vector $x \in \Lambda(P)$ such that $P \cap (P + x) = F$. Such a lattice vector is called a *facet vector*. More generally, a lattice vector $x \in \Lambda(P)$ such that $P \cap (P + x)$ is a face of both P and $P + x$ is called a *standard vector* of P.

For Voronoi cells the facet vectors and the standard vectors are exactly the strictly and weakly Voronoi relevant vectors, respectively. Therefore, we can extend our notation from the previous sections from Voronoi cells and lattices, to general parallelotopes: We write $\mathcal{F}_P$ and $\mathcal{C}_P$ for the set of facet vectors and standard vectors of P, respectively, and $c(P)$ and $\chi(P)$ for the corresponding compactness constants. For example, $\chi(P)$ is the minimal $\chi > 0$ such that there is a basis $B = \{b_1, \ldots, b_n\}$ of $\Lambda(P)$ with the property that every standard vector $x \in \mathcal{C}_P$ can be written as $x = \sum_{i=1}^{n} \gamma_i b_i$, for some $|\gamma_i| \leq \chi$.

With this notation we prove the crucial fact, that if a parallelotope Q decomposes into the Minkowski sum of another parallelotope P and a (possibly lower-dimensional) zonotope Z, then $\chi(Q) \leq \chi(P)$. We write $Z(U) = \sum_{i=1}^{r} [-u_i, u_i]$ for the zonotope spanned by the set of vectors $U = \{u_1, \ldots, u_r\}$.

Proposition 3.12. *Let $Q \subseteq \mathbb{R}^n$ be a parallelotope that admits a decomposition $Q = P + Z(U)$, for some parallelotope P, and a finite set of vectors $U \subseteq \mathbb{R}^n$. Then, there is a linear map $\varphi : \mathbb{R}^n \to \mathbb{R}^n$ with $\varphi(\Lambda(P)) = \Lambda(Q)$ satisfying*

i) For $x \in \mathcal{C}_Q$, we have $\varphi^{-1}(x) \in \mathcal{C}_P$.

ii) For $x \in \mathcal{F}_P$, we have $\varphi(x) \in \mathcal{F}_Q$.

In particular, $\chi(Q) \leq \chi(P)$.

Proof. First note that if $P + Z(U)$ is a parallelotope, then every vector $u \in U$ is a *free* vector for P, that is, $P + [-u, u]$ is a parallelotope as well (cf. [DGM14]). We thus get a chain of parallelotopes $P = P_0 \subseteq P_1 \subseteq \ldots \subseteq P_r = Q$, where $P_i = P_{i-1} + [-u_i, u_i]$, for $1 \leq i \leq r$, and $U = \{u_1, \ldots, u_r\}$. By induction on r it thus suffices to consider the case $r = 1$.

Hence, let $Q = P + [-u, u]$, for some non-zero vector $u \in \mathbb{R}^n$. Dutour Sikirić et al. [DGM14, Lem. 1 & Lem. 3] give a characterization of the standard vectors of Q in terms of those of P: First, there is a dual lattice vector $e_u \in \Lambda(P)^\star$ such that $\Lambda(Q) = A_u \Lambda(P)$, where

$A_u x = x + 2(e_u^\intercal x)u$, for $x \in \mathbb{R}^n$. Then, $z = A_u w \in \Lambda(Q)$ is a standard vector of Q if and only if w is a standard vector of P, and $e_u^\intercal w \in \{0, \pm 1\}$.

This implies that $\varphi(x) = A_u x$ is a bijection between the lattices $\Lambda(P)$ and $\Lambda(Q)$ satisfying $i)$. Moreover, the proof of [DGM14, Lem. 1] reveals that A_u, and thus φ, sends facet vectors to facet vectors, hence $ii)$ holds as well. For $r \geq 2$, we define φ inductively by setting $\varphi(x) = A_{u_r} \cdot \ldots \cdot A_{u_1} x$.

Finally we show that $\chi(Q) \leq \chi(P)$. As just observed, any basis B of $\Lambda(P)$ is sent to a basis $\varphi(B)$ of $\Lambda(Q)$. Moreover, a standard vector $y = \sum_{i=1}^n \alpha_i \varphi(b_i) \in \mathcal{C}_Q$ represented in the basis $\varphi(B)$ corresponds to a standard vector $\varphi^{-1}(y) = \sum_{i=1}^n \alpha_i b_i \in \mathcal{C}_P$ using the same coefficients when represented in the basis B. Thus, if every vector in $\mathcal{C}_P$ can be represented in B with coefficients bounded by $\chi(P)$, the same holds for all vectors in $\mathcal{C}_Q$ with respect to $\varphi(B)$. □

As a consequence, we get that zonotopal parallelotopes Z allow for a compact representation even of the set $\mathcal{C}_Z$ which strengthens Theorem 3.11. In particular, every three-dimensional parallelotope has this property (cf. [Gru07, §32.2]).

Corollary 3.13. *Let Z be a parallelotope that is a zonotope. Then $\chi(Z) = 1$.*

Proof. There is a set of vectors $U' = \{u_1, \ldots, u_m\}$ such that $Z = Z(U')$. We may assume that $u_1, \ldots, u_n$ are linearly independent, and we write $P = [-u_1, u_1] + \ldots + [-u_n, u_n]$. Then, $Z = P + Z(U)$, for $U = U' \setminus \{u_1, \ldots, u_n\}$, and since P is a parallelepiped, it is actually a parallelotope.

Thus, Proposition 3.12 implies that $\chi(Z) \leq \chi(P)$ and it suffices to show that $\chi(P) = 1$ for every parallelepiped P. The standard vectors of P are exactly those $x \in \Lambda(P) \setminus \{0\}$ such that $P \cap (x + P) \neq \emptyset$. Writing $\pm f_1, \ldots, \pm f_n$ for the n pairs of facet vectors of P, we find that $\{f_1, \ldots, f_n\}$ is a basis of $\Lambda(P)$ in which every standard vector admits a $\{0, \pm 1\}$-representation. □

We now focus again on parallelotopes that are Voronoi cells but work in the more convenient language of quadratic forms. A famous conjecture of Voronoi states that every parallelotope is an affine image of a Voronoi cell of a lattice (cf. [Gru07, §32]). As long as this is not settled we need to make the distinction.

Let $q : \mathbb{R}^n \to \mathbb{R}$ be a positive definite quadratic form defined by $q(x) = x^\intercal A^\intercal A x$, for some invertible matrix $A \in \mathbb{R}^{n \times n}$. We associate the lattice $\Lambda = A\mathbb{Z}^n$ to q. Analogously to the lattice case, the Voronoi cell of q is defined as

$$\mathcal{V}_q = \{x \in \mathbb{R}^n : q(x) \leq q(x - z) \text{ for all } z \in \mathbb{Z}^n\}.$$

This is a linear image of the Voronoi cell of Λ and thus a parallelotope. For the sake of brevity we use the shorter notations $\mathcal{F}_q = \mathcal{F}_{\mathcal{V}_q}$, $\mathcal{C}_q = \mathcal{C}_{\mathcal{V}_q}$, $c(q) = c(\mathcal{V}_q)$, and $\chi(q) = \chi(\mathcal{V}_q)$. The exact correspondences between the various notions in the languages of lattices and quadratic forms are as follows.

Lemma 3.14. *Let $q : \mathbb{R}^n \to \mathbb{R}$ be a positive definite quadratic form defined by $q(x) = x^\intercal A^\intercal A x$, for some invertible matrix $A \in \mathbb{R}^{n\times n}$. Moreover, write $\Lambda = A\mathbb{Z}^n$ for the lattice generated by A. Then,*

i) $\mathcal{V}_\Lambda = A\,\mathcal{V}_q$ *and* $\mathcal{F}_\Lambda = A\,\mathcal{F}_q$,

ii) $c(q) = c(\Lambda)$ *and* $\chi(q) = \chi(\Lambda)$.

Proof. For i), observe that

$$\begin{aligned}\mathcal{V}_q &= \left\{x \in \mathbb{R}^n : \|Ax\|^2 \le \|A(x-z)\|^2 \text{ for all } z \in \mathbb{Z}^n\right\}\\ &= \left\{A^{-1}y \in \mathbb{R}^n : \|y\|^2 \le \|y - Az\|^2 \text{ for all } z \in \mathbb{Z}^n\right\} = A^{-1}\mathcal{V}_\Lambda.\end{aligned}$$

For the second identity, notice that $x \in \mathcal{F}_q$ if and only if $\mathcal{V}_q \cap (\mathcal{V}_q + x)$ is a facet of $\mathcal{V}_q$, which holds if and only if $A\mathcal{V}_q \cap (A\mathcal{V}_q + Ax)$ is a facet of $A\mathcal{V}_q = \mathcal{V}_\Lambda$. Part ii) is a direct consequence of these observations. □

The lattice $D_4 = \{z \in \mathbb{Z}^4 : z_1 + \ldots + z_4 \in 2\mathbb{Z}\}$ plays a crucial role in representing 4-dimensional lattices whose Voronoi cell is not a zonotope, and thus deserves a detailed study.

Lemma 3.15. *Let $y \in \mathcal{C}_{D_4} \setminus \mathcal{F}_{D_4}$. Then there is a basis B of D_4 such that*

$$\mathcal{C}_{D_4} \setminus \{\pm y\} \subseteq \{Bz : ||z||_\infty \le 1\}.$$

Proof. We start by characterizing the sets $\mathcal{F}_{D_4}$ and $\mathcal{C}_{D_4}$. Since $\pm 2e_i \pm 2e_j \in 2D_4 \subseteq 2\mathbb{Z}^4$ for $1 \le i < j \le 4$, it follows that $S = \{z \in \{0, \pm 1\}^4 : ||z||^2 \in \{2, 4\}\}$ contains all vectors $v \neq 0$ that are shortest in their respective co-set $v + 2D_4$. In fact, due to parity of the coefficients, we have $S = \mathcal{C}_{D_4}$. In the proof of Proposition 3.10, we saw that $\mathcal{F}_{D_4} \subseteq \{z \in D_4 : ||z||^2 = 2\}$. For parity reasons of $z \in \mathcal{F}_{D_4}$, the vectors z and $-z$ are the unique shortest vectors in $z + 2D_4$, hence we actually have $\mathcal{F}_{D_4} = \{z \in D_4 : ||z||^2 = 2\}$.

Now, let $y \in \mathcal{C}_{D_4} \setminus \mathcal{F}_{D_4}$ and observe that $D_4^\star = \mathbb{Z}^4 \cup (\frac{1}{2}\mathbf{1} + \mathbb{Z}^4)$. Then

$$\left\{x \in \mathbb{R}^4 : |e_i^\intercal x| \le 1, 1 \le i \le 4\right\} \cap \left\{x \in \mathbb{R}^4 : |y^\intercal x| \le 2\right\}$$

is a facet-description of $Q := \operatorname{conv}\{\mathcal{C}_{D_4} \setminus \{\pm y\}\}$. The inequality $|y^\intercal x| \le 2$ arises since the vectors $y_i e_i + y_j e_j$, $1 \le i < j \le 4$ are contained in $\mathcal{C}_{D_4} \setminus \{\pm y\}$. Taking polars, we

obtain that

$$Q^\star = \operatorname{conv}\left(\{\pm e_i : i = 1, \dots, 4\} \cup \{\pm \tfrac{1}{2} y\}\right),$$

and we see that $Q^\star$ contains the dual lattice basis $B^{-\intercal} = \{\frac{1}{2}y, e_1, e_2, e_3\}$. Hence, in the spirit of Lemma 3.2, every vector in $\mathcal{C}_{D_4} \setminus \{\pm y\}$ is represented with coefficients in $\{\pm 1, 0\}$ in the corresponding primal basis B. □

Observe that Lemma 3.15 is best possible in the sense that $\chi(D_4) = 2$.[1] In order to see this, $\operatorname{conv}(\mathcal{C}_{D_4})^\star$ is the standard crosspolytope, which does not contain a basis of $D_4^\star$ as any such basis has to contain a vector in $\frac{1}{2}\mathbf{1} + \mathbb{Z}^4$. However, after dilating by 2, we find the basis $\{\frac{1}{2}\mathbf{1}, e_2, e_3, e_4\}$ of $D_4^\star$ (cf. Lemma 3.2.)

We now arrive at the desired compactness of four-dimensional lattices.

Corollary 3.16. *Every lattice of rank at most four has a compact basis.*

Proof. We have seen in Proposition 3.10 *ii*), that every lattice of rank at most three has a compact basis. Thus, let $\Lambda = A\mathbb{Z}^4$ be a full-dimensional lattice, and let $q(x) = x^\intercal A^\intercal A x$ be the corresponding quadratic form. In the case that $\mathcal{V}_q$ is a zonotope, we use Lemma 3.14 to get that $\mathcal{V}_\Lambda = A\,\mathcal{V}_q$ is a zonotope as well, and thus Theorem 3.11 implies that $c(\Lambda) = 1$.

If $\mathcal{V}_q$ is not a zonotope, then Voronoi's reduction theory as applied in Vallentin's thesis [Val03, Ch. 3] shows the following: We can write $\mathcal{V}_q = \mathcal{V}_p + Z(U)$, for some positive definite quadratic form p and a set of vectors $U \subseteq \mathbb{R}^4$. Moreover, p is such that $\mathcal{V}_p$ is combinatorially equivalent to the 24-cell. Up to isometries and scalings, the only lattice whose Voronoi cell is combinatorially equivalent to the 24-cell is the root lattice D_4, defined in Proposition 3.10. This is due to the fact that D_4 is what is called a *rigid* lattice. Therefore, any lattice corresponding to p agrees with D_4 up to isometries and scalings.

By Lemmas 3.14 and 3.15, this means that for every vector $y \in \mathcal{C}_p \setminus \mathcal{F}_p$, we can find a basis B of $\Lambda_p := \Lambda(\mathcal{V}_p)$ such that every standard vector of $\mathcal{V}_p$ apart from $\pm y$ can be represented with coefficients in $\{\pm 1, 0\}$. By the first part of Proposition 3.12, there is a linear map φ such that $\varphi(\Lambda_p) = \Lambda_q := \Lambda(\mathcal{V}_q)$, and $\varphi^{-1}(\mathcal{F}_q) =: \mathcal{C}' \subseteq \mathcal{C}_p$. By the second part of Proposition 3.12, we have $\mathcal{F}_p \subseteq \mathcal{C}'$. Since $A\mathcal{V}_q$ is a Voronoi cell, we have $|\mathcal{C}'| = |\mathcal{F}_q| \leq 2(2^4 - 1) = 30$, whereas $|\mathcal{C}_p| = |\mathcal{C}_{D_4}| = 40$ (see the proof of Lemma 3.15). Hence we can choose $y \in \mathcal{C}_p \setminus \mathcal{C}'$, and find a basis B of $\mathcal{V}_p$ so that all vectors in $\mathcal{C}'$ are represented with coefficients in $\{0, \pm 1\}$. This implies that all vectors in $\mathcal{F}_q$ have coefficients in $\{0, \pm 1\}$ when represented in the basis $\varphi(B)$ of $\mathcal{V}_q$. Thus, the lattice Λ has a compact basis as $c(q) = c(\Lambda)$. □

We summarize the results of this section in Table 3.1.

[1] Engel et al. [EMS01] claim that they computed $\chi(D_4) = 1$, which turns out to be wrong.

dimension of Λ	compactness result	reference
$n \leq 3$	$c(\Lambda) = \chi(\Lambda) = 1$	Proposition 3.10 & Corollary 3.13
$n = 4$	$c(\Lambda) = 1$, but $\chi(D_4) = 2$	Corollary 3.16
$n \geq 5$	$c(\Lambda_n) \geq \lceil \frac{n}{4} \rceil$	Theorem 3.9

Table 3.1 – Compactness of lattices in small dimensions.

3.6 Relaxing the basis condition

The compact representation problem for the set of Voronoi relevant vectors does not need B to be a basis of the lattice Λ. In fact, it suffices that we find linearly independent vectors $W = \{w_1, \ldots, w_n\}$ that allow to decompose each Voronoi relevant vector as an integer linear combination with small coefficients. This is due to the fact that, given a basis, membership to a lattice can be checked in polynomial time. Thus, in case that the relaxation improves the compactness of the presentation, this additional check is faster than iterating over the larger set corresponding to a $c(\Lambda)$-compact basis.

Definition 3.17. *Let $\Lambda \subseteq \mathbb{R}^n$ be a lattice. A set of linearly independent vectors $W = \{w_1, \ldots, w_n\} \subseteq \mathbb{R}^n$ is called* c-compact for Λ, *if*

$$\mathcal{F}_\Lambda \subseteq \{w_1 z_1 + \ldots + w_n z_n : z \in \mathbb{Z}^n,\ \|z\|_\infty \leq c\}\,.$$

Moreover, we define

$$\overline{c}(\Lambda) = \min\{c \geq 0 : \textit{there is a } c\textit{-compact set } W \textit{ for } \Lambda\}$$

as the relaxed compactness constant *of* Λ.

If every Voronoi relevant vector is an integral combination of W, then so is *every* lattice vector. That is, a c-compact set W for Λ gives rise to a superlattice $\Gamma = W\mathbb{Z}^n \supseteq \Lambda$. The relaxed compactness constant and $c(\Lambda)$ are related as follows.

Proposition 3.18. *For every lattice Λ in $\mathbb{R}^n$, $n \geq 2$, we have*

$$\overline{c}(\Lambda) = \lambda_n(Q^\star, \Lambda^\star) \qquad \textit{and} \qquad \overline{c}(\Lambda) \leq c(\Lambda) \leq \frac{n}{2}\,\overline{c}(\Lambda),$$

where $Q = \mathrm{conv}(\mathcal{F}_\Lambda)$ as before.

Proof. The identity $\overline{c}(\Lambda) = \lambda_n(Q^\star, \Lambda^\star)$ follows by arguments analogous to those establishing the equivalence of *i)* and *ii)* in Lemma 3.2. The inequality $\overline{c}(\Lambda) \leq c(\Lambda)$ is a direct consequence of the definition of these parameters.

In order to prove that $c(\Lambda) \leq \frac{n}{2}\,\overline{c}(\Lambda)$, we let $v_1, \ldots, v_n \in (\overline{c}(\Lambda) \cdot Q^\star) \cap \Lambda^\star$ be linearly independent, and for $1 \leq k \leq n$, we consider the crosspolytope $C_k = \mathrm{conv}\{\pm v_1, \ldots, \pm v_k\}$.

We show by induction that there are vectors $u_1, \dots, u_n \in \Lambda^\star$ such that (a) $\{u_1, \dots, u_k\}$ is a basis of the lattice $\Lambda^\star \cap \operatorname{lin}\{v_1, \dots, v_k\}$, and (b) $u_k \in \max\{\frac{k}{2}, 1\} \cdot C_k$, for every $1 \le k \le n$. This then implies that $\{u_1, \dots, u_n\}$ is a basis of $\Lambda^\star$ contained in $\frac{n}{2}\, C_n \subseteq \frac{n}{2}\, \overline{c}(\Lambda)\, Q^\star$. Hence, $c(\Lambda) \le \frac{n}{2}\, \overline{c}(\Lambda)$, as desired.

First, at least one of the vectors $v_1, \dots, v_n$ must be primitive, say v_1. Then, setting $u_1 = v_1$ gets the induction started. Now, let us assume that we found $u_1, \dots, u_{k-1}$ satisfying (a) and (b). Let $y \in (\Lambda^\star \cap \operatorname{lin}\{v_1, \dots, v_k\})^\star$ be a primitive vector orthogonal to $\operatorname{lin}\{u_1, \dots, u_{k-1}\}$ and such that $y^\intercal v_k \neq 0$. If $|y^\intercal v_k| = 1$, then $u_k = v_k \in C_k$ complements $\{u_1, \dots, u_{k-1}\}$ to a basis of $\Lambda^\star \cap \operatorname{lin}\{v_1, \dots, v_k\}$. So, we may assume that $|y^\intercal v_k| \ge 2$. Every translate of $\frac{k-1}{2}\, C_{k-1}$ within $\operatorname{lin}\{v_1, \dots, v_{k-1}\}$ contains a point of $\Lambda^\star$. In particular, there is a vector $u_k \in \Lambda^\star$ contained in $\frac{1}{y^\intercal v_k}\, v_k + \frac{k-1}{2}\, C_{k-1}$. By construction, u_k complements $\{u_1, \dots, u_{k-1}\}$ to a basis of $\Lambda^\star \cap \operatorname{lin}\{v_1, \dots, v_k\}$, and since $|\frac{1}{y^\intercal v_k}| \le \frac{1}{2}$, we get that $u_k \in \frac{1}{2}\, C_k + \frac{k-1}{2}\, C_{k-1} \subseteq \frac{k}{2}\, C_k$. □

The relaxation to representing $\mathcal{F}_\Lambda$ by generating sets rather than by lattice bases may reduce the respective compactness constant drastically. In fact, the quadratic upper bound in Theorem 3.4 improves to $\mathcal{O}(n \log n)$. However, there is still a class of lattices that shows that in the worst case the relaxed compactness constant can be linear in the dimension as well. In combination with Theorem 3.9, the second part of the following result moreover shows that the factor $n/2$ in Proposition 3.18 is tight up to a constant.

Proposition 3.19.

i) For every lattice $\Lambda \subseteq \mathbb{R}^n$, we have $\overline{c}(\Lambda) \in \mathcal{O}(n \log n)$.

ii) For $a = \lceil \frac{n}{2} \rceil$, let $\Lambda_n = \Lambda_n(a)$ be the lattice defined in (3.3). For every $n \in \mathbb{N}$, we have $\overline{c}(\Lambda_n) \le 3$, whereas $c(\Lambda_n) \ge \lceil \frac{n}{4} \rceil$, for $n \ge 4$.

iii) There are self-dual lattices $\Lambda \subseteq \mathbb{R}^n$ with relaxed compactness constant $\overline{c}(\Lambda) \in \Omega(n)$.

Proof. i): The polytope $Q = \operatorname{conv}(\mathcal{F}_\Lambda)$ is centrally symmetric, all its vertices are points of Λ, and $\operatorname{int}(Q) \cap \Lambda = \{0\}$. Therefore, we have $\lambda_1(Q, \Lambda) = 1$. Proposition 3.18 and the transference theorem of Banaszczyk [Ban96] thus imply that there is an absolute constant $\gamma > 0$ such that

$$\overline{c}(\Lambda) = \lambda_n(Q^\star, \Lambda^\star) = \lambda_1(Q, \Lambda) \cdot \lambda_n(Q^\star, \Lambda^\star) \le \gamma\, n \log n. \tag{3.7}$$

ii): In view of Proposition 3.18, we have to find n linearly independent points of $\Lambda_n^\star$ in $3\, Q^\star$. To this end, we define $y_i := \frac{1}{a}(e_i - e_n)$, for $1 \le i \le n-1$. Furthermore, let $y_n = \frac{1}{a}\mathbf{1}$, if n is even, and $y_n = \left(\{\frac{1}{a}\}^{n-1}, \frac{2}{a}\right)$, if n is odd. We claim that the vectors $y_1, \dots, y_n$ do the job.

First of all, they are clearly linearly independent, and the description (3.4) shows that all these vectors belong to $\Lambda_n^\star$. Now, recall that $Q^\star = \{y \in \mathbb{R}^n : y^\intercal v \leq 1 \text{ for all } v \in \mathcal{F}_{\Lambda_n}\}$. By Lemma 3.8, a Voronoi relevant vector v of Λ_n either equals $\pm\mathbf{1}$ or is contained in $v \in \{a-\ell, -\ell\}^n$, for some suitable $\ell \in \mathbb{N}$. Consider first the vectors y_i, for $1 \leq i \leq n-1$. We have $\mathbf{1}^\intercal y_i = 0$, and for any $v \in \{a-\ell, -\ell\}^n$ holds $v^\intercal y_i = \frac{1}{a}(v_i - v_n)$ which equals 0, if $v_i = v_n$, and it equals ± 1, if $v_i \neq v_n$. Thus, in fact $y_1, \ldots, y_{n-1} \in Q^\star$.

Regarding the remaining vector y_n, we observe that $\mathbf{1}^\intercal y_n = 2$, independently of the parity of the dimension n. Thus, let $v \in \{a-\ell, -\ell\}^n$, and note that $\ell \in \{\lfloor \frac{ak}{n} \rfloor, \lceil \frac{ak}{n} \rceil\}$, where $k = |\{i : v_i = a-\ell\}|$. Since $-\ell \leq a$ and $a - \ell \leq a$, we have

$$\begin{aligned} y_n^\intercal v &\leq \tfrac{1}{a}\left(k(a-\ell) - (n-k)\ell + a\right) = \tfrac{1}{a}\left(ka - n\ell + a\right) \\ &\leq \tfrac{1}{a}\left(ka - n(\tfrac{ak}{n} - 1) + a\right) = \frac{n+a}{a} \leq 3, \end{aligned}$$

and similarly $y_n^\intercal v \geq -3$. Hence, $y_n \in 3\,Q^\star$, finishing the proof.

iii): Let Λ be a self-dual lattice and let $\mathcal{V}_\Lambda$ be its Voronoi cell. Each Voronoi relevant vector $v \in \mathcal{F}_\Lambda$ provides a facet of $\mathcal{V}_\Lambda$ via the inequality $v^\intercal x \leq \frac{1}{2}\,||v||^2$, as well as a facet of $Q^\star$ via the inequality $v^\intercal x \leq 1$ (this indeed defines a facet, as a vertex v of Q always induces a corresponding facet of the polar $Q^\star$). As $||v|| \geq \lambda_1(B_n, \Lambda)$, for every $c < \lambda_1(B_n, \Lambda)^2$, we have that $c \cdot Q^\star$ is contained in the interior of twice the Voronoi cell of $\Lambda^\star = \Lambda$, and hence contains no non-trivial dual lattice point. Therefore, $\overline{c}(\Lambda) \geq \lambda_1(B_n, \Lambda)^2$.

Conway & Thompson (see [MH73, Ch. 2, §9]) proved that there are self-dual lattices Λ in $\mathbb{R}^n$ with minimal norm

$$\lambda_1(B_n, \Lambda) \geq \left\lfloor \frac{1}{\sqrt{\pi}} \left(\frac{5}{3}\Gamma\left(\frac{n}{2}+1\right)\right)^{\frac{1}{n}} \right\rfloor.$$

Stirling's approximation then gives that $\overline{c}(\Lambda) \in \Omega(n)$. □

Based on the common belief that the best possible upper bound in (3.7) is linear in n, we conjecture the following:

Conjecture 3.20. *The compactness constants are linearly bounded, that is*

$$\overline{c}(\Lambda) \in \mathcal{O}(n) \qquad \textit{and also} \qquad c(\Lambda) \in \mathcal{O}(n),$$

for every lattice $\Lambda \subseteq \mathbb{R}^n$.

3.7 Algorithmic point of view

When it comes to computing a $c(\Lambda)$-compact basis for Λ, not much is known. Lemma 3.2 suggests to take the polar of $\mathrm{conv}(\mathcal{F}_\Lambda)$, and then to look for a dual basis in a suitable dilate thereof. However, in order to do this, we need a description of the Voronoi relevant vectors in the first place. Even if we are only interested in an $(n \cdot c(\Lambda))$-compact basis, it is not clear how to benefit from the allowed slack.

In the following, we rather discuss how to incorporate an already known c-compact basis into the algorithm of Micciancio & Voulgaris [MV13].

The Micciancio-Voulgaris algorithm

The algorithm consists of two main parts. In a preprocessing step, it computes the Voronoi cell $\mathcal{V}_\Lambda$, which can be done in time $2^{\mathcal{O}(n)}$ in a recursive manner. As a c-compact basis already grants a superset of $\mathcal{F}_\Lambda$, we do not recall the details of this first part.

Once the Voronoi cell $\mathcal{V}_\Lambda$ is computed, a vector $p \in \Lambda$ is closest to t if and only if $t - p \in \mathcal{V}_\Lambda$. Bearing this in mind, the idea is to iteratively subtract lattice vectors from t until the condition holds.

But why do we only need $2^{\mathcal{O}(n)}$ iterations? Let us assume for now that t is already rather close to 0, say $t \in 2\,\mathcal{V}_\Lambda$. Let p be a Voronoi relevant vector whose induced facet-defining inequality is violated by t, this means $p^\intercal t > \frac{1}{2}\,||p||^2$. Micciancio & Voulgaris show that $t - p$ is still contained in $2\,\mathcal{V}_\Lambda$, and is strictly shorter than t. Hence, for going from $t \in 2\,\mathcal{V}_\Lambda$ to some $t' = t - w \in \mathcal{V}_\Lambda$, for $w \in \Lambda$, the number of iterations we need is bounded by the number of level sets of the norm function that have a point in $2\,\mathcal{V}_\Lambda \cap (t + \Lambda)$. This number turns out to be at most 2^n.

If t is further away, that is $t \notin 2\,\mathcal{V}_\Lambda$, let k be the smallest integer such that $t \in 2^k \mathcal{V}_\Lambda$. Then, we can apply the above method to the lattice $\Lambda' = 2^{k-1}\Lambda$, and find $w \in \Lambda' \subseteq \Lambda$ such that $t - w \in \mathcal{V}_{\Lambda'} = 2^{k-1}\mathcal{V}_\Lambda$. Doing this recursively yields that after $2^n k$ iterations, we moved t into $\mathcal{V}_\Lambda$. Note that k is polynomial in the input size. More sophisticated arguments allow to limit k in terms of n only, or to decrease the number of iterations to weakly polynomial, as presented in [DB15].

Corollary 3.21. *Assume we are given a c-compact basis B of a lattice $\Lambda \subseteq \mathbb{R}^n$. For any target point $t \in \mathbb{R}^n$, a closest lattice vector to t can be found in time $\mathcal{O}((2c+1)^n\, 2^n\, \mathrm{poly}(n))$ and space polynomial in the input size.*

Proof. Theorem 4.2 and Remark 4.4 in [MV13] state that a closest vector can be found in time $\mathcal{O}(|V| \cdot 2^n\, \mathrm{poly}(n))$, where V is a superset of the Voronoi relevant vectors $\mathcal{F}_\Lambda$. We set $V = \{Bz : z \in \mathbb{Z}^n,\, \|z\|_\infty \leq c\} \supseteq \mathcal{F}_\Lambda$.

The reduction to polynomial space follows from [MV13, Rem. 4.3]: Their algorithm may need exponential space because they store $\mathcal{F}_\Lambda$. As a subset of V it is however described just by the polynomial-size data (B, c). □

The Micciancio-Voulgaris algorithm naturally can be presented as an algorithm for the Closest Vector Problem with Preprocessing (CVPP). In this variant of CVP, we may precompute the lattice for an arbitrary amount of time and store some additional information. Only then the target vector is revealed to us, and we are allowed to use the information we gathered before to speed up the process of finding a closest vector. This is motivated by the fact that in practice, we might have to compute the closest vector for several target vectors, but always on the same lattice. Hence, we happily spend more time for preprocessing, when we are able to vastly benefit from the additional information.

Considered in this setting, our results compress the information after the preprocessing step into polynomial space. However, it is unclear how to compute a $c(\Lambda)$-compact basis *without* computing the Voronoi cell first.

Open Question:
Can we compute a basis B of Λ that attains $c(\Lambda)$ in single-exponential time and polynomial space?

The fact that every zonotopal lattice has a compact basis is especially interesting. McCormick, Peis, Scheidweiler & Vallentin can solve the Closest Vector Problem in polynomial time on a zonotopal lattice, provided it is given in a certain format.[2] Another related result is due to McKilliam, Grant & Clarkson [MGC14], who provide a polynomial time algorithm for lattices of Voronoi's first kind, provided an obtuse superbasis is known. One could wonder whether our representation also allows for solving CVPP faster (measuring only the time after the preprocessing). However, McKilliam et al. use additional combinatorial properties of an obtuse superbasis that are in general not even fulfilled for a 1-compact basis. In fact, Micciancio [Mic01] showed that if CVPP can be solved in polynomial time for arbitrary lattices, then NP $\subseteq$ P/poly and the polynomial hierarchy collapses.

[2] At the time of writing there is no preprint available (personal communication with Frank Vallentin).

4 The closest vector problem with additional information

Introduction

In this chapter we are concerned with requirements on a lattice that allow us to solve the closest vector problem more efficiently. The easiest example is an orthogonal basis B. We can represent $t = By$ with rational coefficients, and a simple application of Pythagoras' theorem allows to conclude that $B\lfloor y \rceil$ is a closest point to t.

A more interesting example are lattices of Voronoi's first kind, lattices that have an *obtuse superbasis*. Essentially, this basis allows us to implement a variant of the algorithm of Micciancio & Voulgaris in polynomial time [MGC14]. We refer to the previous discussion on these lattices in Section 3.4 for more details.

In the first section, we will present two results. For one, we will show that a result of Dadush & Bonifas [DB15] implies that whenever we have a separation oracle for the Voronoi cell, we can solve the closest vector problem in polynomial time. As an application, we will revisit the class of lattices for which the Voronoi cell is a zonotope (cf. Section 3.4). We saw already that the rich structure of these lattices allow for a 1-compact basis. Here we will see that if we are given a set of generators of the Voronoi cell, this implies a separation oracle for the Voronoi cell, and we can solve CVP in polynomial time.

However, in all the cases above, it is not the lattice per se that allows for a more efficient algorithm, but some additional information on the lattice. In the listed examples, this is an orthogonal basis, an obtuse superbasis, or the generators of the zonotopal Voronoi cell. The second part of this chapter is concerned with the question whether we can recognize these special classes of lattices even if the additional information is hidden from us.

We focus on the specific class of lattices that have an orthonormal basis. In particular, we show that deciding whether a given lattice Λ has an orthonormal basis is in the complexity class NP $\cap$ co-NP. While containment in NP is not difficult to show, we

will deploy a strong result from analytic number theory for showing containment in co-NP [Elk95]. On the way, we show that the problem is equivalent to the *unimodular decomposition problem*. Given a positive definite unimodular matrix $G \in \mathbb{Z}^{n\times n}$, decide whether there exists a unimodular matrix U such that $U^\intercal U = G$. We think that this is an interesting problem on its own. Maybe surprisingly, the equivalence (together with known structural results on lattices) implies that such a matrix always exists if $n \leq 7$, and for $n = 8$, there are matrices for which a decomposition $G = U^\intercal U$ does not exist.

4.1 The closest vector problem in polynomial time

For this section, we assume a full-dimensional rational lattice $\Lambda \subseteq \mathbb{Q}^n$, given by a basis $B \in \mathbb{Q}^{n\times n}$, and either a separation oracle or an optimization oracle for the Voronoi cell $\mathcal{V} := \mathcal{V}_\Lambda$. The main result is that such an oracle already suffices for solving the closest vector problem for Λ in polynomial time.

Though we discussed the algorithm of Micciancio & Voulgaris already briefly, we will present another point of view that will be more helpful for this section. We ignore the scaling technique they use for now, but assume that t is rather close to 0, i.e. $||t||_\mathcal{V} \leq n$, where $||\cdot||_\mathcal{V}$ denotes the norm induced by $\mathcal{V}$.

Geometrically, the algorithm of Micciancio & Voulgaris follows a line segment $[0, t]$ for some target t, until it leaves the Voronoi cell $\mathcal{V}$ through a facet. It keeps track of the Voronoi relevant vector v_1 inducing this facet, updates $t' \leftarrow t - v_1$, and iterates, now following the updated line segment $[0, t']$. If $t' \in \mathcal{V}$ after some k iterations, it outputs $\sum_{i=1}^{k} v_k$.

Instead of reducing t' in each iteration and resetting the line segment, we could also continue following the line segment $[0, t]$ through the entered Voronoi cell $v_1 + \mathcal{V}$. Eventually, we will leave $v_1 + \mathcal{V}$, entering the next Voronoi cell $v_2' + \mathcal{V}$ and so forth, finally ending up at a lattice point closest to t. This straight line approach is chosen by Dadush & Bonifas, and depicted in Figure 4.1.

With this approach, it becomes more clear that the Voronoi relevant vectors equip the lattice with a graph structure, and we navigate in this graph towards a vertex closest to t. Two points $u, v \in \Lambda$ are connected by an edge if the vector $u - v$ is Voronoi relevant or, equivalently, the Voronoi cells $u + \mathcal{V}$ and $v + \mathcal{V}$ share a facet. Every time the line segment $[0, t]$ leaves a Voronoi cell, we make one step in the graph. We call this graph the *Voronoi graph* $G_\mathcal{V}$ of Λ. The total length of the traversed path in the Voronoi graph is equal to the number of Voronoi cells the line segment $[0, t]$ intersects (at least if $[0, t]$ does not intersect lower-dimensional faces).

While the approach of Micciancio & Voulgaris induces a path on this graph whose length is only bounded exponentially, Dadush & Bonifas showed that with a slight modification of

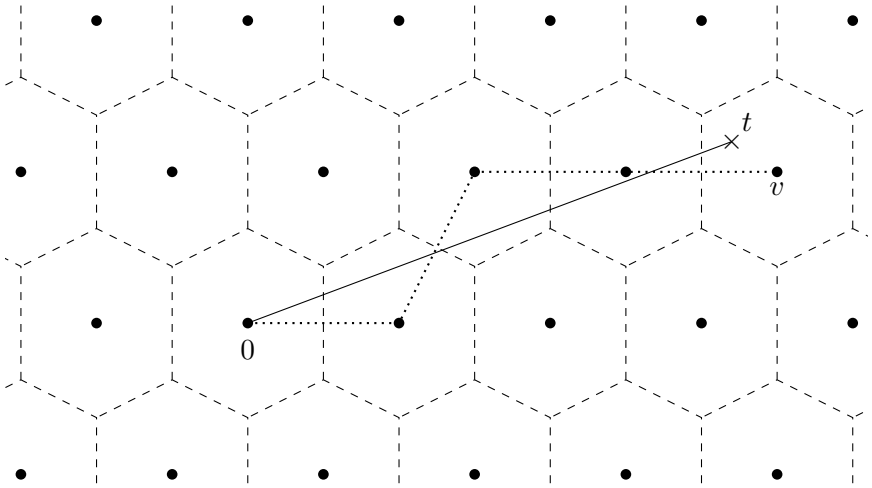

Figure 4.1 – The path on the Voronoi graph induced by following a straight line.

the approach outlined above we can expect to traverse a path of polynomial length [DB15]. This modification is a perturbation of the start and endpoint, allowing for a probabilistic analysis.

More specifically, they sample a point $Z \in \mathcal{V}$, and follow the three line segments $[0, Z]$, $[Z, t + Z]$, and $[t + Z, t]$ one after another. Clearly, as $Z \in \mathcal{V}$, the first line segment does not leave the initial Voronoi cell. The second line segment will cross a number of Voronoi cells that is linear in $n \, ||t||_{\mathcal{V}}$ in expectation.

The last line segment is more difficult to handle. On a high level, the problem is that the closer we come to t on the line segment $[t + Z, t]$, the smaller the effect of the randomness of the sampled point Z becomes. They overcome this problem by showing that either t is on the boundary of a Voronoi cell, or has a minimum distance δ to the boundary, where δ depends on the encoding size. Therefore, they can stop following $[Z + t, t]$ already at distance δ away from t, and must already be in the same Voronoi cell as t. For this truncated line segment, they are again able to limit the expected number of crossings.

Before we give the expected amortized number of crossings for the whole procedure, we formalize it in several steps for a given lattice $\Lambda \subseteq \mathbb{Q}^n$ and target $t \in \mathbb{Q}^n$. We call this procedure the *perturbed straight line procedure*.

1. Find a vector $y \in \Lambda$ such that $t \in y + n\mathcal{V}$. Update $t \leftarrow t - y$ so that $||t||_{\mathcal{V}} \leq n$.

2. Sample $Z \in \mathcal{V}$.

3. Follow the line segments $[0, Z]$, $[Z, t + Z]$, $[t + Z, t + \delta]$ for δ small enough, and navigate on the Voronoi graph accordingly.

We will now provide the expected length of the path we follow in the Voronoi graph. This reformulation is gained by combining the estimates of the Lemmas 3.2 and 3.3 with Theorem 3.2 in [DB15].

Theorem 4.1 (Cf. [DB15, Thm. 3.1 & 3.2])**.** *Given a lattice $\Lambda \subseteq \mathbb{R}^n$ and a target $t \in \mathbb{R}^n$ such that $||t||_{\mathcal{V}} \leq n$. Then the length of the path on the Voronoi graph induced by the perturbed straight line procedure has expected length $\mathcal{O}(n^2(2 + \text{size}(B, t)))$.*

With this estimate at hand, it remains to show (a) how to find an initial point $y \in \Lambda$ with $||y - t||_{\mathcal{V}} \leq n$, (b) how to sample a point $Z \in \mathcal{V}$, and (c) how to perform a single step on the Voronoi graph. As performing a step corresponds to the line segment $[Z, Z + t]$ (or $[Z + t, t]$) crossing a facet, we consider the following, more general problem. Afterwards, we dedicate one subsection to each of the points, assembling everything in the end.

Facet Piercing Problem

INSTANCE: A (possibly implicitly given) polytope $P \subseteq \mathbb{R}^n$, $s \in P$, $t \in \mathbb{R}^n \setminus P$.
TASK: Find a facet normal of a facet F of P s.t. $F \cap [s, t] \neq \emptyset$.

Clearly, the difficulty of this problem drastically depends on the given representation of P. If P is just given in its outer description $\{x \in \mathbb{R}^n \mid Ax \leq b\}$, then the problem is solvable by computing λ such that $a^\intercal(s + \lambda(t - s)) = \beta$ for every inequality, and choose the one yielding the smallest positive λ. This representation is chosen by Dadush & Bonifas, who assume to have a list of Voronoi relevant vectors explicitly given. And while this is polynomial in the input size, in our setting $P = \mathcal{V}$ usually has already $\Theta(2^n)$ facets, and the approach is not polynomial in the dimension.

4.1.1 Solving the facet piercing problem

In this section, we will show that we can solve the facet piercing problem for a Voronoi cell in polynomial time, provided we are given an optimization oracle for it and the starting point s is in the interior. We will also quote the classical result that separation implies optimization (under certain assumptions), leaving us with the result that we can solve the facet piercing problem whenever we have an oracle for either the separation or the optimization problem.

However, the precise statements and proofs might be a bit technical and tedious. Since we are interested in Voronoi cells, we will assume that $\Lambda \subseteq \mathbb{Q}^n$ is a full-dimensional lattice, and restrict our attention to full-dimensional polytopes. If $\Lambda \subseteq \mathbb{Q}^n$ was of lower dimension $d < n$, we could first compute its Gram-Schmidt orthogonalization, and rotate it onto the first d coordinates. Thus, our assumption is without loss of generality.

We start by establishing some bounds on the encoding size of involved quantities.

Lemma 4.2. *For $n \in \mathbb{Z}_{\geq 2}$ let $\Lambda \subseteq \mathbb{Q}^n$ be a full-dimensional lattice given by a basis $B \in \mathbb{Q}^{n\times n}$, $f \in \Lambda$ a Voronoi relevant vector, and $v \in \mathbb{Q}^n$ a vertex of $\mathcal{V}_\Lambda$. Let $s, t \in \mathbb{Q}^n$ be two points such that the intersection $[s,t] \cap \{x \in \mathbb{R}^n \mid f^\intercal x = \frac{||f||^2}{2}\}$ is a unique point p. Then the following estimates hold.*

i) $\operatorname{size}(f) \leq 6n^3 \operatorname{size}(B)$ *and* $\operatorname{size}(||f||_2^2) \leq 6n^2 \operatorname{size}(B)$

ii) $\operatorname{size}(v) \leq 32n^5 \operatorname{size}(B)$

iii) $\operatorname{size}(p) \leq 44n^4 \operatorname{size}(B, s, t)$.

Proof. For now, assume $B \in \mathbb{Z}^{n\times n}$, and hence $\Lambda \subseteq \mathbb{Z}^n$. We will deduct the rational case from there later. Standard estimates for the covering radius of Λ yield $\mu(\Lambda) \leq \frac{1}{2}\sqrt{\sum_{i=1}^n ||b_i||_2^2}$ (cf. [DB15, Lem. A.1]).

Let f be a Voronoi relevant vector. Since the Voronoi cell of Λ is contained in a ball of radius $\mu(\Lambda)$, we have that $||f||_2 \leq 2\mu(\Lambda) \leq \sqrt{\sum_{i=1}^n ||b_i||_2^2}$, implying

$$\begin{aligned} \operatorname{size}(||f||_2^2) &\leq \operatorname{size}(\sum_{i=1}^n b_i^\intercal b_i) \\ &\leq 2\sum_{i,j=1}^n \operatorname{size}(b_{i,j}^2) \\ &\leq 4\sum_{i,j=1}^n \operatorname{size}(b_{i,j}) \\ &\leq 4\operatorname{size}(B) - 4n^2 \end{aligned} \tag{4.1}$$

and since $|f_i| \leq ||f||_2^2$,

$$\operatorname{size}(f) \leq n + 4n\operatorname{size}(B) - 4n^3 \leq 4n\operatorname{size}(B),$$

using integrality and the size estimates of Section 1.5.

Let $v \in \mathbb{Q}^n$ be a vertex of $\mathcal{V}_\Lambda$, $f_1, \ldots, f_n \in \Lambda$ the Voronoi relevant vectors whose induced facets define v, F be the matrix with rows $f_1^\intercal, \ldots, f_n^\intercal$, and $w \in \mathbb{Q}^n$ defined by $w_i = \frac{||f_i||^2}{2}$ so that v is the unique solution to $Fx = w$. Observe that the Estimate (4.1), together with $n \geq 2$ implies $\operatorname{size}(w) \leq n + n(\operatorname{size}(1/2) + 4\operatorname{size}(B) - 4n^2) \leq 4n\operatorname{size}(B)$. By Theorem 1.6, and again using $n \geq 2$, we have

$$\begin{aligned} \operatorname{size}(v) &\leq 4n(\operatorname{size}(F) + \operatorname{size}(w)) \\ &\leq 4n(4n^2\operatorname{size}(B) + 4n\operatorname{size}(B)) \\ &\leq 24n^3\operatorname{size}(B). \end{aligned}$$

For the third part, let $f \in \Lambda$ be a Voronoi relevant vector inducing a facet F of $\mathcal{V}_\Lambda$, and assume in addition to $B \in \mathbb{Z}^{n\times n}$ that $s, t \in \mathbb{Z}^n$, and that $[s,t] \cap F$ contains a single point $p \in \mathbb{Q}^n$. Let k be the index minimizing $|(t-s)_k|$ over all non-zero entries. Let A be a full-rank matrix obtained by taking the canonic unit vectors $\{e_i \mid i \in [n] \setminus \{k\}\}$ as the first $n-1$ columns, and $t-s$ as the last one. Let $u_1, \ldots, u_n$ denote the rows of A^{-1} and observe that $t - s \perp u_1, \ldots, u_{n-1}$. Moreover, any entry of each u_i is the quotient of a $n-1 \times n-1$ subdeterminant of A and the determinant of A by Cramer's rule. Due to the simple structure of A, the determinant of A is $\pm(t-s)_k$, and subdeterminant is either 1, 0 or a single entry of $t-s$. Due to our choice of k, we thus have $\operatorname{size}(u_i) \leq 2\operatorname{size}(t-s)$. Now, p is the unique solution to the system $f^\intercal x = \frac{||f||^2}{2}$, $u_2^\intercal(p-s) = 0, \ldots, u_2^\intercal(p-s) = 0$, and again by Theorem 1.6, the size of p is bounded in terms of the size of this system, culminating in the estimate

$$\begin{aligned}\operatorname{size}(p) &\leq 4n\left(2(n-1)\operatorname{size}(t-s) + \operatorname{size}(f) + (n-1)\operatorname{size}(s) + 2(n-1)\operatorname{size}(t-s)\right)\\ &\leq 34n^2\operatorname{size}(B,s,t).\end{aligned}$$

It remains to consider the case where B is not integral. For the first two cases, let $\overline{q} \in \mathbb{Z}_{\geq 1}$ be the smallest integer such that $B' = \overline{q}B \in \mathbb{Z}^{n\times n}$, and denote $\Lambda' := \overline{q}\Lambda \subseteq \mathbb{Z}^n$. Writing $b_{i,j} = \frac{p^B_{i,j}}{q^B_{i,j}}$ with $p^B_{i,j} \in \mathbb{Z}$, $q^B_{i,j} \in \mathbb{Z}_{\geq 0}$, we have that $\overline{q} \leq \prod_{i=1}^n \prod_{j=1}^n q^B_{i,j}$ and thus $\operatorname{size}(\overline{q}) \leq \operatorname{size}(B) - n^2$. Moreover, $\operatorname{size}(B') = n^2 + \sum_{i,j}(\operatorname{size}(\overline{q}) + \operatorname{size} b_{i,j})) \leq (n^2+1)\operatorname{size}(B) - n^4$.

If $f \in \Lambda$ is Voronoi relevant (for $\mathcal{V}_\Lambda$), then $\overline{q}f \in \Lambda'$ is Voronoi relevant (for $\mathcal{V}_{\Lambda'}$), and with the above we can estimate (again, using $n \geq 2$)

$$\begin{aligned}\operatorname{size}(f) &\leq n\operatorname{size}(\overline{q}^{-1}) + \operatorname{size}(\overline{q}f)\\ &\leq n\operatorname{size}(B) + 4n\operatorname{size}(B')\\ &\leq 6n^3\operatorname{size}(B)\end{aligned}$$

and $\operatorname{size}(||f||_2^2) \leq 6n^2\operatorname{size}(B)$. Similarly, for $v \in \mathbb{Q}^n$ a vertex of $\mathcal{V}_\Lambda$ we estimate

$$\begin{aligned}\operatorname{size}(v) &\leq \operatorname{size} n(\overline{q}^{-1}) + \operatorname{size}(\overline{q}v)\\ &\leq n\operatorname{size}(B) + 24n^3\operatorname{size}(B')\\ &\leq 32n^5\operatorname{size}(B).\end{aligned}$$

For the last point, we have to choose $\tilde{q} \in \mathbb{Z}_{>0}$ large enough so that not only $\tilde{q}B \in \mathbb{Z}^{n\times n}$, but also $\tilde{q}s, \tilde{q}t \in \mathbb{Z}^n$, and thus obtain the estimates

$$\begin{aligned}\operatorname{size}(\tilde{q}) &\leq \operatorname{size}(B,s,t) - n^2,\\ \operatorname{size}(B',s',t') &\leq (n^2+1)\operatorname{size}(B,s,t).\end{aligned}$$

In the same manner as above we estimate

$$\begin{aligned}\operatorname{size}(p) &\leq n \operatorname{size}(\tilde{q}^{-1}) + \operatorname{size}(\tilde{q}p)\\ &\leq n(\operatorname{size}(\tilde{q}) + 2) + 34n^2 \operatorname{size}(B', s', t')\\ &\leq n \operatorname{size}(B, s, t) + 34n^2(n^2+1) \operatorname{size}(B, s, t)\\ &\leq 44n^4 \operatorname{size}(B, s, t).\end{aligned}$$

□

Next, we provide a theorem stating that separation implies optimization under certain assumptions.

Theorem 4.3 (cf. [KV18, Thm. 4.21], [GLS81])**.** *Let $n \in \mathbb{Z}_{\geq 1}$ and $c \in \mathbb{Q}^n$. Let $P \subseteq \mathbb{R}^n$ be a rational polytope, and let $x_0 \in \mathbb{Q}^n$ be a point in the interior of P. Let $T \in \mathbb{Z}_{\geq 1}$ such that $\operatorname{size}(x_0) \leq \log_2(T)$ and $\operatorname{size}(x) \leq \log_2(T)$ for all vertices x of P.*

Given n, c, x_0, T and a polynomial-time oracle for the separation problem for P, a vertex $x^\star$ of P attaining $\max\{c^\intercal x \mid x \in P\}$ can be found in time polynomial in n, $\log_2(T)$ and $\operatorname{size}(c)$.

Though this theorem already provides us with a vertex solution, we briefly address the question how an optimization oracle can be turned into an optimization oracle that outputs an optimum *vertex* solution. Morally, we add a lexicographic order for vertices with the same objective value into the objective. We will refer to this fact later.

Lemma 4.4 ([KV18, Lem. 4.20])**.** *Let $n \in \mathbb{Z}_{\geq 1}$, let $P \subseteq \mathbb{R}^n$ be a rational polytope, and let $x_0 \in \mathbb{Q}^n$ be a point in the interior of P. Let $T \in \mathbb{Z}_{\geq 1}$ such that $\operatorname{size}(x_0) \leq \log_2(T)$ and $\operatorname{size}(x) \leq \log_2(T)$ for all vertices x of P. Then $B(x_0, r) \subseteq P \subseteq B(x_0, R)$, where $r := \frac{1}{n}T^{-379n^2}$ and $R := 2nT$.*

Moreover, let $K := 4T^{2n+1}$. Let $c \in \mathbb{Z}^n$, and define $c' := K^n c + (1, K, \ldots, K^{n-1})^\intercal$. Then $\max\{c'^\intercal x : \ x \in P\}$ is attained by a unique vector $x^\star$, for all other vertices y of P we have $c'^\intercal(x^\star - y) > T^{-2n}$, and $x^\star$ is also an optimum solution of $\max\{c^\intercal x : \ x \in P\}$.

We are now able to show how to solve the facet piercing problem if s is in the interior of P. In Section 4.1.4 we will see how to achieve this requirement. We remark that the statement and proof is essentially taken from [KV18, Thm. 4.23], though we extend the proof by an observation crucial for us.

Theorem 4.5. *Let $n \in \mathbb{Z}_{\geq 1}$ and $t \in \mathbb{Q}^n$. Let $P \subseteq \mathbb{R}^n$ be a rational polytope, and let $s \in \mathbb{Q}^n$ be a point in the interior of P. Let $T \in \mathbb{Z}_{\geq 1}$ such that $\operatorname{size}(t) \leq \log_2(T)$, $\operatorname{size}(s) \leq \log_2(T)$ and $\operatorname{size}(x) \leq \log_2(T)$ for all vertices x of P.*

Given n, t, s, T and an oracle which for any given $c \in \mathbb{Q}^n$ returns a vertex $x^\star$ of P attaining $\max\{c^\intercal x \mid x \in P\}$, we can solve the separation problem for P and y in time polynomial in $n, \log_2(T)$ and $\operatorname{size}(t)$. Indeed, in the case $t \notin P$ we can find an inequality defining a facet F such that $F \cap [s,t] \neq \emptyset$.

Proof. We can shift the setting by $-s$ and replace T with $T2^{\operatorname{size}(s)}$, henceforth we assume $s = 0$. Let

$$F = \{f \in \mathbb{Q}^n \mid f^\intercal x \leq 1\}$$

be the set of normalized facet-defining inequalities of P, and observe that $P^\star = \operatorname{conv}(F)$. Since each facet-defining inequality $v^\intercal x \leq 1$ for $P^\star$ stems from a vertex $v \in P$, we have that each vertex of $P^\star$ is the unique solution to a system

$$\begin{pmatrix} v_1^\intercal \\ \vdots \\ v_n^\intercal \end{pmatrix} x = \mathbf{1},$$

where the v_i are corresponding vertices of P. Therefore, by Theorem 1.6, each vertex z of $P^\star$ satisfies $\operatorname{size}(z) \leq 4n(n \log_2(T) + 3n) \leq 12n^2 \log_2(T)$.

Consider the problem $\max\{t^\intercal x \mid x \in P^\star\}$. If the optimum is at most 1, then $y^\intercal t \leq 1$ for all $y \in P^\star$, and $t \in (P^\star)^\star = P$. If the optimum is larger than 1, let y be a point with $y^\intercal t > 1$. Since $y^\intercal x \leq 1$ is a valid inequality for P, this gives a separating hyperplane. Moreover, if y is a vertex, this hyperplane is facet-defining. Now, observe that whenever $\lambda \leq \frac{1}{y^\intercal t}$ the point λt satisfies the inequality $y^\intercal x \leq 1$. Thus, the vertex $y \in P^\star$ maximizing $t^\intercal x$ yields a facet that gets pierced by $[0, t]$. This means that the facet piercing problem over P is equivalent to the optimization problem over $P^\star$.

Since the size of each vertex of $P^\star$ is bounded by $12n^2 \log_2(T)$ and 0 is in the interior, we can apply Theorem 4.3 and only need to show how to separate over $P^\star$.

But this reduces to optimizing over P. Indeed, for a point $z \in \mathbb{Q}^n$, we have $z \notin P^\star$ if and only if $\max\{z^\intercal y \mid y \in P\} > 1$. Therefore, optimizing over P implies that we can solve the facet piercing problem, if we know a point in the interior. □

We will see later that the requirement for s being in the interior does not harm us. To this end, we establish that for any point $s \in \mathcal{V}_\Lambda$ and any facet F, we either have $s \in F$ or there is a lower bound on the distance. Also Dadush & Bonifas derived a similar bound in their work.

Lemma 4.6. *Let $n \in \mathbb{Z}_{\geq 2}$ and $\Lambda \subseteq \mathbb{Q}^n$ be a full-dimensional rational lattice given by a basis $B \in \mathbb{Q}^{n \times n}$. Let $s \in \mathcal{V}_\Lambda$ and $H \subseteq \mathbb{Q}^n$ a supporting hyperplane inducing a facet. Let $T \in \mathbb{Z}_{\geq 1}$ such that $\operatorname{size}(s) \leq \log_2(T)$ and $\operatorname{size}(B) \leq \log_2(T)$. Then either $s \in H$ or $\operatorname{dist}(s, H) \geq T^{-32n^3}$.*

Proof. Let $2f^\intercal x = ||f||_2^2$ be the equality inducing H. If $s \in H$ there is nothing to show, henceforth assume $s \notin H$.

By Lemma 4.2 we have $\text{size}(f) \leq 6n^3 \,\text{size}(B) \leq 6n^3 \log_2(T)$, $\text{size}(||f||_2^2) \leq 6n^2 \log_2(T)$, and $\text{size}(s) \leq \log_2(T)$ by assumption. Thus, there is an integer $M \leq T^{6n^3+6n^2+1}$ such that Mf, Ms and $M\,||f||_2^2$ are integral. The hyperplane MH is given by the integral equality

$$2Mf^\intercal x = M^2\,||f||_2^2\,,$$

and as $Ms \in \mathbb{Z}^n$ we have that $2Mf^\intercal(Ms) \leq M^2\,||f||_2^2 - 1$, implying that $\text{dist}(MH, Ms) \geq 1/(2M\,||f||_2)$. Scaling back, we obtain

$$\text{dist}(H, s) \geq 1/(2M^2\,||f||_2).$$

Therefore, $\text{dist}(H, s) \geq T^{-32n^3}$. □

By Theorem 4.3, we can replace the optimization oracle in the previous statement with a separation oracle.

4.1.2 Finding a starting vertex close to the target

We adapt a technique that is commonly used. The general idea is to express $t = \sum_{i=1}^n \alpha_i v_i$ as a linear combination of n linearly independent Voronoi relevant vectors, and then take $y = \sum_{i=1}^n \lfloor \alpha_i \rceil v_i$. As $(\alpha_i - \lfloor \alpha_i \rceil)v_i \in \mathcal{V}_\Lambda$ for each i, we have $y - t \in n\mathcal{V}_\Lambda$ by convexity. It is left to show how to find n linearly independent facet vectors when we are only given a separation oracle.

Lemma 4.7. *Given a lattice Λ and a target vector t, we can find a lattice vector y such that $t - y \in n\mathcal{V}_\Lambda$ by solving the facet piercing problem n times for $\mathcal{V}_\Lambda$ with a starting point s in the interior of $\mathcal{V}_\Lambda$.*

Proof. We first give an iterative approach to find n linearly independent Voronoi relevant vectors. Start with $S = \emptyset$. As long as $|S| < n$, pick a vector $x \in \mathbb{R}^n \setminus \{0\}$ orthogonal to $\text{lin}(S)$ (where we assume that any vector is orthogonal to $\text{lin}(\emptyset)$). Then, choose r large enough (e.g. $||rx||_2 \geq \sum_{i=1}^n ||b_i||$ for a given lattice basis B) and determine a facet inequality $f^\intercal x \leq 1$ for a facet that gets pierced by the line segment $[0, rx]$ by Theorem 4.5, where we set T such that $\log_2(T) \geq 35n^5 \,\text{size}(B) + \text{size}(r)$. Set $\alpha = \frac{2}{||f||_2^2}$, $v = \alpha f$, and observe that the inequality $2v^\intercal x \leq ||v||_2^2$ induces the same facet as $f^\intercal x \leq 1$. Therefore, $v \in \Lambda$ is the corresponding Voronoi relevant vector. Update $S \leftarrow S \cup \{v\}$ and repeat. Since our choice of x is always parallel to the induced facets we obtained so far, we add a linearly independent facet vector to the set S in every step.

Now, we express $t = \sum_{v \in S} \alpha_v v$ for some rational values α_v, and set $y = \sum_{v \in S} \lfloor \alpha_v \rceil v$. We

obtain that every summand in $t - y = \sum_{v \in S}(\lfloor \alpha_v \rceil - \alpha_v)v$ is contained in $\mathcal{V}_\Lambda$, and due to convexity, $t - y \in n\mathcal{V}_\Lambda$. □

4.1.3 Sampling a vector in the Voronoi cell

Dadush & Bonifas sample a vector $z \in \mathcal{V}_\Lambda$ almost uniform at random, using the results of Dyer, Frieze & Kannan. The probability distribution Uniform(K) is induced by the function $\mu(x) = 1/\operatorname{vol}(K)$ for $x \in K$ and 0 otherwise.

Theorem 4.8 ([DB15, Thm. A.2], [DFK91])**.** *Let $K \subseteq \mathbb{R}^n$ be a convex body, given by a membership oracle, satisfying $B(0,r) \subseteq K \subseteq B(0,R)$. Then for $\varepsilon > 0$, a realization of a random variable $X \in K$, having total variation distance at most ε from* Uniform(K)*, can be computed using* $\operatorname{poly}(n, \log_2(R/r), \log(1/\varepsilon))$ *arithmetic operations and calls to the membership oracle.*

As a separation oracle immediately realizes a membership oracle, we can use this result in our setting as well, though the access to the list of Voronoi relevant vectors allows Dadush & Bonifas a better grasp on the quantity $\frac{R}{r}$.

Lemma 4.9. *Given a basis B of a lattice $\Lambda \subseteq \mathbb{R}^n$ and a separation oracle for the Voronoi cell, we can compute a* (1/4)*-uniform sample $z \in \mathcal{V}_\Lambda$ in polynomial time.*

Proof. The Voronoi cell satisfies $B(0, \frac{\lambda_1}{2}) \subseteq \mathcal{V}_\Lambda \subseteq B(0, \mu)$, where λ_1 is the length of a shortest lattice vector and μ is the covering radius of Λ. We can compute a 2^n-approximation to λ_1 by using the LLL-algorithm, and $\frac{1}{2}\sum_{i=1}^n ||b_i||_2$ is an upper bound on the covering radius. Therefore, we can compute r, R such that $B(0,r) \subseteq \mathcal{V}_\Lambda \subseteq B(0,R)$ and $\log_2(\frac{R}{r}) \in \operatorname{poly}(\operatorname{size}(B))$.

The membership oracle for $\mathcal{V}_\Lambda$ can be realized by a single call to the separation oracle for $\mathcal{V}_\Lambda$. Choosing $\varepsilon = \frac{1}{4}$ and using the previous theorem, the claim follows. □

4.1.4 The main result and CVP on zonotopal lattices

Theorem 4.10. *Let $\Lambda \subseteq \mathbb{Q}^n$ be a full-dimensional lattice given by a basis B, and $t \in \mathbb{Q}^n$. If we have a separation oracle for the Voronoi cell of Λ, we can solve the closest vector problem in expected polynomial time (polynomial in n and* $\operatorname{size}(B,t)$*).*

Proof. We follow the proof of [DB15, Thm. 3.4] and replace the step of switching to the next Voronoi cell by checking the list of Voronoi relevant vectors with our oracle-base approach.

We first assure $t \in n\mathcal{V}_\Lambda$ with Lemma 4.7. Then we sample a vector $Z \in \mathcal{V}_\Lambda$ by Lemma 4.9.

In the rest of the proof, we will make use of the following claim, shown by Dadush & Bonifas.

[DB15, Claim C.1] With probability 1, the path $[Z, t+Z] \cup [t+Z, t)$ only intersects $\Lambda + \partial\mathcal{V}_\Lambda$ in the relative interior of its facets. Furthermore, with probability 1, the intersection consists of isolated points, and $Z, t+Z \notin \Lambda + \partial\mathcal{V}_\Lambda$.

This allows us to choose T_0 such that $\log_2(T_0) \geq \max\{\text{size}(Z), \text{size}(t+Z), 32n^5 \,\text{size}(B)\}$, and use Theorem 4.5 to find an inequality $f^\intercal x \leq 1$ that induces a facet pierced by $[Z, t+Z]$, or assert that $t+Z \in \mathcal{V}_\Lambda$, in which case 0 is optimal.

Set $\alpha = \frac{2}{||f||_2^2}$, $v_1 = \alpha f$, and observe that the inequality $2v_1^\intercal x \leq ||v_1||_2^2$ induces the same facet as $f^\intercal x \leq 1$, hence $v_1 \in \Lambda$ is the corresponding Voronoi relevant vector. This also allows us to compute the point s_0 where $[Z, t+Z]$ pierces the facet.

Assume we made already k steps and have a point s_k on the facet shared by $v_k + \mathcal{V}_\Lambda$ and $v_{k+1} + \mathcal{V}_\Lambda$. With probability 1, s_k is an isolated point in the relative interior of a facet of $v_k + \mathcal{V}_\Lambda$, and thus there exists $\varepsilon > 0$ such that $s_k + \varepsilon(t-s)$ is in the interior of $v_{k+1} + \mathcal{V}_\Lambda$.

By Lemmas 4.2 and 4.6, we have $\text{size}(s_k) \leq 44n^4 \,\text{size}(B, Z, t+Z)$, and setting $\tilde{T}$ large enough we set $s_k' = s_k + \tilde{T}^{-60n^3} \frac{t-s}{||t-s||_2}$, i.e. $s_k' \in (s_k, Z+t] \subseteq [Z, Z+t]$.

Choosing T such that $\log_2(T) \geq 10(60n^3 \log_2(\tilde{T}) + \text{size}(t, s)$, we can apply Theorem 4.5 on s_k' to find the facet and the point where $[s_k', Z+t]$ exits the Voronoi cell $v_{k+1} + \mathcal{V}_\Lambda$. Since T and $\tilde{T}$ are chosen independent of the iteration, we can iterate the process until we reach $Z+t$. As with probability 1, $Z+t$ is in the interior of a Voronoi cell, we can continue with our procedure for the segment $[Z+t, t)$ until we are close enough to t, cf. [DB15, Lem. 3.2].

In expectation, after $\mathcal{O}(n^2(2+\text{size}(B,t))$ iterations, we found a closest vector. Therefore, we choose some large constant c, and if we did not find a closest vector after $cn^2(2+\text{size}(B,t)$, or if the path does cross a lower-dimensional face, we restart the algorithm by sampling a new Z.

It is shown in [DB15, Thm. 3.4] that we only have to restart the algorithm a constant number of times in expectation. □

Remark 4.11. *Even if a line segment $[s,t]$ pierces the common facet of two neighbouring Voronoi cells $v_1 + \mathcal{V}_\Lambda, v_2 + \mathcal{V}_\Lambda$ in a point p in a lower dimensional face, we could continue with the algorithm instead of restarting. To see this, recall that Theorem 4.4 adds a lexicographic order on all optimal vertices - which in our case are the facets, as we switch to the polar. We can follow the line segment $[v_2, p]$ in $v_2 + \mathcal{V}_\Lambda$ and optimize with respect to different lexicographic orders, providing different facets containing p.*

However, we did not pursue this approach, and recon that we have to take heed of other

issues. For instance, if $\Lambda = \mathbb{Z}^n$ and $t = \mathbf{1}$, the segment $[0, t]$ pierces through the vertex of 2^n distinct Voronoi cells. We do not want to traverse through all of them with this approach, as traversing n cells suffices.

If the Voronoi cell is a zonotope given by its generators, we can implement a separation oracle in polynomial time.

Corollary 4.12. *Let $\Lambda \subseteq \mathbb{Q}^n$ be a full-dimensional lattice whose Voronoi cell $\mathcal{V}_\Lambda$ is a zonotope, and $t \in \mathbb{Q}^n$. Given a basis B of Λ, and the generators $g_1, \ldots, g_m$ of $\mathcal{V}_\Lambda$, we can solve the closest vector problem in time polynomial in n and the input size.*

Proof. Recall that any zonotope with m generators is a projection of an m-dimensional cube. Phrased differently, a zonotope has extension complexity at most $2m$ with extended formulation

$$\left\{ \sum_{i=1}^{m} \alpha_i g_i : -1 \leq \alpha_i \leq 1 \right\},$$

together with the projection $\alpha_i e_i \mapsto \alpha_i g_i$, where e_i is the i-th canonic unit vector. This allows us to optimize over the zonotope, and thus apply Theorem 4.10. □

4.2 The rotated standard lattice problem is in NP ∩ co-NP

Let $B \in \mathbb{R}^{n \times n}$ be a non-singular matrix generating a full-dimensional Euclidean lattice $\Lambda(B) = \{Bz \mid z \in \mathbb{Z}^n\}$. Several problems in the algorithmic theory of lattices, such as the shortest or closest vector problem, or the covering radius problem, become very easy if the columns of B form an orthonormal basis. But surprisingly, if B is any basis and we want to decide whether there exists an orthonormal basis of $\Lambda(B)$, not much is known.

As this is equivalent to $\Lambda(B)$ being a rotation of $\mathbb{Z}^n$, we call this decision problem the *rotated standard lattice problem* (RSLP), which is the main concern of this section. A related problem is the *unimodular decomposition problem*: For a given unimodular and positive definite matrix G, decide whether there exists a unimodular matrix U such that $G = U^\intercal U$. The goal of this section is twofold. For one, we show that RSLP and UDP are equivalent, and that both problems belong to NP ∩ co-NP. We will use a result of Elkies on *characteristic vectors* [Elk95], which appear in analytic number theory. It seems that characteristic vectors are rather unknown in the algorithmic lattice theory. The second attempt of this section is thus to introduce Elkies' result and characteristic vectors as a new possible tool for further algorithmic results in this area.

Related work

Lenstra & Silverberg show that RSLP can be decided in polynomial time, provided that additional information on the automorphism group of the lattice is part of the input [LS17]. However, they do not discuss the complexity of the general problem without additional information. When the input lattice is a construction-A lattice, Chandrasekaran, Gandikota & Grigorescu show that existence of an *orthogonal* basis can be decided in polynomial time [CGG17]. If it exists, they also find one.

RSLP can be viewed as a special case of the *lattice isomorphism problem (LIP)*, which, given two lattices Λ_1, Λ_2, asks whether there is an isomorphism $\varphi : \Lambda_1 \to \Lambda_2$ between the two lattices that preserves the Euclidean structure ($\langle x, y\rangle = \langle \varphi(x), \varphi(y)\rangle$). Here, $\langle x, y\rangle = \sum_{i=1}^{n} x_i y_i$ for $x, y \in \mathbb{R}^n$ is the standard scalar product.

The lattice isomorphism problem was introduced by Plesken & Souvignier [PS97], solving it in small dimension for specific lattices of interest. In [DSV09], Dutour Sikirić, Schürmann & Vallentin show that this problem is at least as hard as the more famous graph isomorphism problem. The best algorithm for the lattice isomorphism problem the author is aware of is due to Haviv & Regev, and has a running time of $n^{\mathcal{O}(n)}$ [HR14]. They solve the problem by computing all orthogonal linear transformations (i.e. all isomorphisms) between the two given lattices Λ_1, Λ_2. On the complexity side, they show that the problem is in the complexity class SZK (statistical zero knowledge), which already suggests that it is not NP-hard.

In the first subsection, we provide the background specific to this problem, including a formal introduction of the problems in consideration and characteristic vectors. In the second subsection, we show that both problems are in NP ∩ co-NP.

4.2.1 The rotated standard lattice problem and the unimodular decomposition problem

Two lattices Λ_1, Λ_2 are *isomorphic*, in symbols $\Lambda_1 \cong \Lambda_2$, if there exists an orthogonal matrix $Q \in \mathbb{R}^{n\times n}$ such that $\Lambda_1 = Q\Lambda_2$. In this case, it follows that B_2 is a basis of Λ_2 if and only if $B_1 := QB_2$ is a basis of Λ_1. We are interested in the following problem, which is a special case of the lattice isomorphism problem.

Rotated Standard Lattice Problem (RSLP)
INSTANCE: A d-dimensional lattice $\Lambda \subseteq \mathbb{R}^n$, given by a basis $B \in \mathbb{Q}^{n\times d}$.
TASK: Decide whether $\Lambda \cong \mathbb{Z}^d$.

While the problem is stated for a d-dimensional lattice in $\mathbb{R}^n$, we will assume $d = n$ for now as we need the more general definition only later. The attentive reader might notice

that an isomorphism refers to an orthogonal matrix Q, whereas a rotation usually refers to an orthogonal matrix with positive determinant, $\det(Q) = 1$. However, in our setting, i.e. one of the lattices in consideration is fixed to $\mathbb{Z}^n$, the terms turn out to be equivalent for the following reason. A matrix B generates $\mathbb{Z}^n$, if and only if B is unimodular. Given an isomorphic basis QB, we can multiply the first column of Q and the first row of B by -1 without changing the basis QB. This flips the sign of $\det(Q)$, while B remains unimodular.

Though a lattice is usually specified by a basis matrix B, we will see that another representation is preferable for this problem. The *Gram matrix* G of a basis B is defined as $G := B^\intercal B$, i.e. $G_{i,j} = \langle b_i, b_j \rangle$. An advantage of the Gram matrix is that it "forgets" the embedding of a lattice Λ into the Euclidean space, and only carries the information of the isomorphism class of Λ:

Lemma 4.13. *Two lattice bases $B_1, B_2 \in \mathbb{R}^{n\times n}$ generate isomorphic lattices $\Lambda(B_1)$, $\Lambda(B_2) \subseteq \mathbb{R}^n$, if and only if there exists a unimodular matrix $U \in \mathbb{Z}^{n\times n}$ such that for the corresponding Gram matrices G_1, G_2 the relation $G_1 = U^\intercal G_2 U$ holds.*

In particular, a lattice $\Lambda(B_1)$ is isomorphic to $\mathbb{Z}^n$, if and only if there exists a unimodular matrix $U \in \mathbb{Z}^{n\times n}$ such that $G_1 = U^\intercal U$.

Proof. If there is an isomorphism $\Lambda(B_1) = Q\Lambda(B_2)$, then QB_2 is a basis of Λ_1. This implies that there is a unimodular matrix $U \in \mathbb{Z}^{n\times n}$ such that $B_1 = QB_2U$, and we obtain $G_1 = U^\intercal G_2 U$.

On the other hand, if $G_1 = U^\intercal G_2 U$, define $Q := B_1 U^{-1} B_2^{-1}$, and verify

$$Q^\intercal Q = B_2^{-\intercal}(U^{-\intercal} B_1^\intercal B_1 U^{-1}) B_2^{-1} = B_2^{-\intercal} G_2 B_2^{-1} = \mathbb{1}.$$

Hence, we find $Q\Lambda(B_2) = \Lambda(QB_2) = \Lambda(B_1U^{-1}) = \Lambda(B_1)$, since U^{-1} is again unimodular.

The second part follows by observing that the identity matrix $B_2 := \mathbb{1}$ is a basis of $\mathbb{Z}^n$, and hence $G_2 = \mathbb{1}$. □

Clearly, a Gram matrix is always symmetric and positive definite. This grants a reduction to the following problem.

Unimodular Decomposition Problem (UDP)
INSTANCE: A symmetric, positive definite, unimodular matrix $G \in \mathbb{Z}^{n\times n}$.
TASK: Decide whether there exists a unimodular matrix $U \in \mathbb{Z}^{n\times n}$ such that $G = U^\intercal U$.

As it turns out, if we allow for a lattice to be embedded in a higher dimension, the

problems are even equivalent. However, the reverse direction requires some more effort, since we need to find a rational lattice basis B such that $B^\intercal B = G$ for a given matrix G. In particular, such a matrix might not exist in the same dimension. This is why we defined the problem for general dimensions $d \leq n$. The following lemma will provide us with the technical statement, so that the equivalence of the two problems follows as a corollary.

Lemma 4.14. *Let $G \in \mathbb{Q}^{n\times n}$ be a symmetric, positive definite matrix, and define $s := \mathrm{size}(G)$. Then, there exists a matrix $B \in \mathbb{Q}^{4n\times n}$ such that $G = B^\intercal B$. Moreover, we can compute such a matrix in expected polynomial (in n and s) time, and we can compute a matrix $B \in \mathbb{Q}^{f_n(s)\times n}$ such that $B^\intercal B = G$, where $f_n(s) \in \mathcal{O}(n\log(ns))$, in deterministic polynomial (in n and s) time.*

Proof. Let us outline the proof first. We show that the Gaussian elimination method can be used to find a rational decomposition $G = T^{-1}DT^{-\intercal}$, where $D \in \mathbb{Z}^{n\times n}$ is an integer diagonal matrix. We will denote a n-by-n diagonal matrix with diagonal entries $d_1, \ldots, d_n$ by $\mathrm{diag}(d_1, \ldots, d_n)$. Lagrange's four-square theorem (see e.g. [HW08, Chap. 20.5]) states that every non-negative integer can be written as the sum of four squares. Hence, we can write every diagonal entry d_k of D as $d_k = v_k^\intercal v_k$, where $v_k \in \mathbb{Z}^{4\times 1}$ is a matrix whose entries are integers $n_1(k), \ldots, n_4(k)$ such that $d_k = \sum_{i=1}^4 n_i(k)^2$. Setting F to be a block-diagonal matrix containing the vectors v_k, we obtain

$$G = T^{-1}DT^{-\intercal} = T^{-1}F^\intercal FT^{-\intercal},$$

and can set $B := FT^{-\intercal}$. For the algorithmic approach, we use the result [PT18] with expected running time $\mathcal{O}(\frac{\log(d_k)^2}{\log(\log(d_k))})$ to obtain the four squares for each diagonal entry. If we want to define a sequence of squares deterministically, we have to increase the number of squares from 4 to $\mathcal{O}(\log(\log(d_k)))$.

Let us discuss the details, starting with the Gaussian algorithm, as outlined in [KV18, Chap. 4.3]. After the k-th step of the Gaussian elimination, our matrix has the shape

$$TG = \begin{pmatrix} R & M \\ 0 & S \end{pmatrix}, \quad R \in \mathbb{Q}^{k\times k}, M \in \mathbb{Q}^{k\times n-k}, S \in \mathbb{Q}^{n-k\times n-k},$$

where R is upper triangular. If we did not swap rows yet, $T \in \mathbb{Q}^{n\times n}$ is a lower triangular matrix whose last $n-k$ columns coincide with the identity. But then symmetry implies that

$$TGT^\intercal = \begin{pmatrix} D' & 0 \\ 0 & S \end{pmatrix},$$

where $D' \in \mathbb{Q}^{k\times k}$ is a diagonal matrix, and S is the same matrix as for the matrix product TG before. This is true because multiplying with $T^\intercal$ does not change the last $n-k$ rows of TG. If $s_{1,1} = 0$, then $e_{k+1}^\intercal TGT^\intercal e_{k+1} = 0$, a contradiction to the positive

definiteness of G. Thus, we also do not have to swap in iteration $k+1$, and by induction we obtain a lower triangular matrix T such that $TGT^\intercal =: D' \in \mathbb{Q}^{n\times n}$ is a diagonal matrix. If the entries of D' are given as $d_k = \frac{p_k}{q_k}$, we set $T' = T\,\mathrm{diag}(\frac{1}{q_1},\dots,\frac{1}{q_n})$ and $D = \mathrm{diag}(p_1q_1,\dots,p_nq_n) \in \mathbb{Z}^{n\times n}$ to obtain the desired decomposition $G = T'^{-1}DT'^{-\intercal}$.

Edmonds [Edm67] observed that the Gaussian elimination can be implemented in polynomial time. A more careful analysis, as in [KV18, Chap. 4.3], reveals that all occurring numbers (i.e. the p_i's and q_i's in particular) are bounded by $\mathcal{O}(n^2 s)$, where s is the encoding size of G. Therefore, the encoding size of the entries of D are bounded by $\mathcal{O}(n^4 s^2)$.

With the discussion in the beginning of the proof, this shows existence and the randomized algorithm.

To show how to obtain the deterministic result, let $a \in \mathbb{N}$ be any number. We define two sequences $(a_i)_{i\in\mathbb{N}}$ and $(n_i)_{i\in\mathbb{N}}$ as follows. Set $a_0 = a$. Then recursively, set $n_i = \lfloor\sqrt{a_{i-1}}\rfloor$ and $a_i = a_{i-1} - n_i^2 \geq 0$. It is clear that there is a finite r such that $n_i \equiv 0$ for $i \geq r$, and we have $\sum_{i=1}^{r} n_i^2 = a_0$. To limit the number r from above, observe that $n_i^2 \leq a_{i-1} < (n_i+1)^2$ implies $a_i = a_{i-1} - n_i^2 < 2n_i + 1$, thus $a_i \leq 2\sqrt{a_{i-1}}$ due to integrality. Resolving the recursion, we obtain $a_k \leq 4a^{(1/2)^k}$. Once $a_k \leq 8$, it is easy to see that we need at most four additional steps. Resolving $4a^{(1/2)^k} \geq 8$ reveals that $r \leq \lceil\log(\log(a_0))\rceil + 4$ numbers suffice such that $\sum_{i=1}^{r} n_i^2 = a$. Clearly, the computations can be done in time polynomial in the input, and the numbers n_i can be found with binary search in $\log(a_{i-1})$ steps, which is also polynomial in the input size.

Turning our attention back to the matrix D from before, we construct a sequence $n_1(k),\dots,n_{r(k)}$ for each diagonal entry d_k, and define a vector consisting of this sequence, $v_i^\intercal = (n_1(k),\dots,n_{r(k)})$, with $r(k) \in \mathcal{O}(\log(\log(a_0)))$, thus $r(k) \in \mathcal{O}(\log(2\log(n)+s))$. This leads to a block diagonal matrix $F = \mathrm{diag}(v_1,\dots,v_n)$ such that $F^\intercal F = D'$. In total, we obtain $G = T^{-1}Q^\intercal F^\intercal FQT^{-\intercal}$.

Thus, we can construct such a sequence for every entry d_k of the matrix D, and write the numbers in a vector $v_k^\intercal = (n_1(k),\dots,n_r(k))$. Inserting our bound on the encoding size, we obtain $r \in \mathcal{O}(\log(ns))$.

Defining $B := FQT^{-\intercal}$ finishes the proof of the deterministic result. □

Corollary 4.15. *The problems UDP and RSLP are polynomial-time equivalent.*

Proof. Let $G \in \mathbb{R}^{d\times d}$ be an instance of UDP. Apply Lemma 4.14 to obtain a matrix $B \in \mathbb{Q}^{n\times d}$ such that $B^\intercal B = G$, where $n \in \mathcal{O}(d\log(ds))$ for the encoding size s. Now, $\Lambda(B) \cong \mathbb{Z}^d \times \{0\}^{n-d}$ if and only if there exists an orthogonal matrix $Q \in \mathbb{R}^{n\times n}$ and a unimodular matrix $U \in \mathbb{Z}^{d\times d}$ such that $QBU = \binom{\mathbb{1}}{0}$. This in turn is equivalent to $U^\intercal GU = \mathbb{1}$, thus $G = U^{-\intercal}U$ is a unimodular decomposition.

For the other direction, let $G = B^\intercal B$. Now, $\Lambda(B) \equiv \mathbb{Z}^n$ if and only if there exists an orthogonal matrix Q and a unimodular matrix U such that $QBU = \mathbb{1}$, which is equivalent to $U^\intercal G U = \mathbb{1}$. □

4.2.2 Self-dual lattices and characteristic vectors

Clearly, UDP is in NP. A certificate is simply given by the matrix U, whose entries are bounded by $\max\{\sqrt{G_{ii}} : 1 \leq i \leq n\}$. In the following, we will discuss a paper of Elkies [Elk95] and apply his results to show that the problem is also in co-NP.

To this end, let us recall some facts for a full-dimensional lattice $\Lambda \subseteq \mathbb{R}^n$. We call Λ *self-dual* if $\Lambda = \Lambda^\star$. Self-dual lattices are also called unimodular lattices for the following reason.

Lemma 4.16 (See e.g. [CS99])**.** *A matrix $B \in \mathbb{R}^{n\times n}$ generates a self-dual lattice $\Lambda(B)$ if and only if the corresponding Gram matrix $G = B^\intercal B$ is unimodular.*

Proof. Let B be the basis of a self-dual lattice. Then $B^{-\intercal}$ is a basis as well, and there exists a unimodular matrix G such that $B^{-\intercal} G = B$, which is equivalent to $G = B^\intercal B$.

Let $G = B^\intercal B$ be unimodular, and $x = Bz_1$ and $y = Bz_2$ be any two lattice vectors. Since $x^\intercal y = z_1^\intercal G z_2$, we have $\Lambda(B) \subseteq \Lambda(B)^\star$. Since $\det(\Lambda(B))^2 = \det(B)^2 = \det(G) = 1$, they have to be the same already. □

Definition 4.17. *Let $\Lambda = \Lambda^\star \subseteq \mathbb{R}^n$ be a self-dual lattice. A vector $w \in \Lambda$ is a* characteristic vector of the lattice Λ*, if*

$$\forall\, v \in \Lambda : \quad \langle v, w\rangle \equiv_2 \langle v, v\rangle\,,$$

where $x \equiv_k y$ is shorthand for $x \equiv y \mod k$.

It is known that for dimensions $n \leq 7$, the lattice $\mathbb{Z}^n$ is the unique self-dual lattice (up to isomorphism). In dimension 8, the lattice

$$E_8 = \left\{ z \in \mathbb{R}^8 \,\middle|\, \sum_{i=1}^{8} z_i \equiv_2 0, z \in \mathbb{Z}^8 \cup \left(\frac{1}{2}\mathbf{1} + \mathbb{Z}^8\right) \right\}, \quad \text{with } \mathbf{1} := (1, \ldots, 1)^\intercal,$$

is self-dual, but not isomorphic to $\mathbb{Z}^8$ (cf. [CS99]). The easiest way to see this is to verify that E_8 does not have any vector of length 1, whereas $\mathbb{Z}^8$ does.

Before we turn our attention to the main result, let us show some basic properties of characteristic vectors. In terms of quadratic forms, the computations carried out in Point *ii*) of the following lemma are already discussed in [Ger04]. Also in terms of quadratic forms, Point *iii*) can be found in [Ser73, Chap. V].

Lemma 4.18. *The following are true for every self-dual lattice $\Lambda = \Lambda^\star \subseteq \mathbb{R}^n$.*

i) *There exists a characteristic vector $w \in \Lambda$.*

ii) *The set of characteristic vectors is precisely a co-set $w + 2\Lambda$, where $w \in \Lambda$ is any characteristic vector.*

iii) *For any two characteristic vectors $u, w \in \Lambda$, we have $||u||^2 - ||w||^2 \equiv_8 0$.*

iv) *If we are given a Gram matrix G of Λ, we can compute in polynomial time a vector $z \in \mathbb{Z}^n$ such that for all $y \in \mathbb{Z}^n$, we have $y^\intercal G z \equiv_2 y^\intercal G y$.*

v) *The shortest characteristic vectors of the lattice $\mathbb{Z}^n$ are the vectors $\{-1, 1\}^n$.*

Proof. *i)* Let $B = (b_1, \ldots, b_n)$ be a basis of Λ, and $D = (d_1, \ldots, d_n) = B^{-\intercal}$ the corresponding dual basis, also spanning Λ. Represented in the primal basis, define a vector $w = \sum_{i=1}^n ||d_i||^2 b_i \in \Lambda$, and let $v = \sum_{i=1}^n \alpha_i d_i \in \Lambda$ be any lattice vector, represented in the dual basis D. Using $x^2 \equiv_2 x$ for $x \in \mathbb{Z}$, we obtain

$$\begin{aligned}\langle v, v\rangle &= \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j d_i^\intercal d_j \equiv_2 \sum_{i=1}^n \alpha_i^2 ||d_i||^2 \\ &\equiv_2 \sum_{i=1}^n \alpha_i ||d_i||^2 = \sum_{i=1}^n \sum_{j=1}^n \alpha_i ||d_j||^2 d_i^\intercal b_j = \langle v, w\rangle .\end{aligned}$$

Thus, w is a characteristic vector and *i)* is shown.

ii) Now let w be as in the first part, and w' be any characteristic vector. Since for any $y \in \Lambda$, we have $\langle w' + 2y, v\rangle = \langle w', v\rangle + 2\langle y, v\rangle \equiv_2 \langle w', v\rangle$, the whole co-set $w' + 2\Lambda$ consists of characteristic vectors. If w' has the representation $w' = \sum_{i=1}^n \gamma_i b_i \in \Lambda$, computing

$$||d_k||^2 = \langle d_k, d_k\rangle \equiv_2 \langle w', d_k\rangle = \gamma_k$$

for every coefficient γ_k shows that $w' \in w + 2\Lambda$, finishing point *ii)*.

iii) Let $w = Bc$ be a characteristic vector. It suffices to show that we have $||w||^2 - ||w + 2b_k||^2 \equiv_8 0$ for $k = 1, \ldots, n$. The claim then follows by repeatedly adding or subtracting twice a basis vector. Let $G = B^\intercal B$, and compute

$$||w + 2b_k||^2 = c^\intercal G c + 4 \underbrace{c^\intercal G e_k}_{\equiv_2 e_k^\intercal G e_k} + 4 e_k^\intercal G e_k \equiv_8 ||w||^2 + 8 G_{kk},$$

where we used that w is characteristic. Since $G_{kk} \in \mathbb{Z}$, we are done.

iv) Observe that $G^{-1} = D^\intercal D$; hence, by the definition of w', we can compute G^{-1} and set $z_k = (G^{-1})_{kk}$ for $k = 1, \ldots, n$.

v) Choosing the identity matrix as lattice basis, it follows that all characteristic vectors of $\mathbb{Z}^n$ must have odd entries only, hence point *v*) follows.

□

The vector z in Point *iv*) can also be seen in a different way. If B is any lattice basis such that $G = B^\intercal B$, then Bz is a characteristic vector of $\Lambda(B)$. Therefore, the vector z can be seen as the coefficient vector of a characteristic vector. Such a "coefficient vector" will be the co-NP certificate for UDP.

4.2.3 Applying the result of Elkies

We are now able to provide the main result of this article. The crucial argument we are using will be Elkies' theorem, which reads as follows.

Theorem 4.19 ([Elk95])**.** *Let Λ be a self-dual lattice in $\mathbb{R}^n$ with no characteristic vector such that $\|w\|^2 < n$. Then $\Lambda \cong \mathbb{Z}^n$.*

As a slight remark, we recall that the shortest characteristic vectors of $\mathbb{Z}^n$ are $\{-1, 1\}^n$ by Lemma 4.18, Point *v*). Hence in this case, $\|w\|^2 = n$.

With this theorem, we can prove that RSLP is in co-NP as follows. If $\Lambda(B) \ncong \mathbb{Z}^n$, then the certificate is a characteristic vector $w \in \Lambda(B)$ with $||w||^2 < n$. By Theorem 4.19, such a vector exists. Hence, we only have to show that we can check the certificate in polynomial time, i.e. we have to ensure that $||w||^2 < n$, and $w^\intercal v \equiv_2 v^\intercal v$ for all $v \in \Lambda(B)$.

However, if we turn out attention to UDP, we are not given a lattice explicitly, but only a Gram matrix G. If $G = B^\intercal B$, our certificate will therefore be the coefficient vector z of a short characteristic vector $w = Bz \in \Lambda(B)$. As it turns out, such a vector z is independent of the particular choice of B for which $B^\intercal B = G$.

We focus on showing UDP $\in$ co-NP rather than RSLP $\in$ co-NP as the reduction from UDP to RSLP increases the dimension by a factor of four.

Lemma 4.20. *Let $G \in \mathbb{Z}^{n\times n}$ be a symmetric, positive definite, and unimodular matrix, and $z \in \mathbb{Z}^n$. We have*

$$e_k^\intercal G e_k \equiv_2 e_k^\intercal G z, \quad \forall\, k = 1, \ldots, n,$$

if and only if for every matrix B with $B^\intercal B = G$, the vector $w = Bz$ is a characteristic vector in the lattice $\Lambda(B)$.

Proof. Let $x = \sum_{i=1}^n \alpha_i b_i \in \Lambda$ be any vector, and $w = Bz \in \Lambda$.

If w is a characteristic vector, we have $e_k^\intercal G e_k = \langle b_k, b_k\rangle \equiv_2 \langle b_k, w\rangle = e_k^\intercal G z$.

For the other direction, we find

$$\begin{aligned}\langle x, w-x\rangle &= \left\langle \sum_{i=1}^{n} \alpha_i b_i, w - \sum_{i=1}^{n} \alpha_i b_i \right\rangle = \left\langle \sum_{i=1}^{n} \alpha_i b_i, v \right\rangle - \sum_i \sum_j \alpha_i \alpha_j \langle b_i, b_j \rangle \\ &\equiv_2 \sum_i \alpha_i (e_i^\intercal G z)) - \sum_i \alpha_i^2 (e_i^\intercal G e_i) \equiv_2 \sum_i (\alpha_i - \alpha_i^2) e_i^\intercal G e_i \\ &\equiv_2 0,\end{aligned}$$

showing that w is indeed a characteristic vector. □

This lemma also shows that we can check whether a vector w is characteristic by only checking $\langle w, b_i\rangle \equiv_2 \langle b_i, b_i\rangle$ on some basis $\{b_1, \ldots, b_n\}$, instead of all lattice vectors. It is straightforward that this can be checked in polynomial time, provided the encoding size of z is bounded in the encoding size of the input.

Lemma 4.21. *Let $G \in \mathbb{Z}^{n\times n}$ be a symmetric, positive definite, and unimodular matrix, and $z \in \mathbb{Z}^n$ such that $z^\intercal G z \leq n$. Then the bit complexity of z is polynomially bounded by the bit complexity of G.*

Proof. Let $M \in \mathbb{Z}$ be a bound on the entries of G, i.e. $|G_{i,j}| \leq M$ for all i, j in range, and let $v_1, \ldots, v_n$ be an orthonormal basis of eigenvectors with eigenvalues $0 < \lambda_1 \leq \cdots \leq \lambda_n \leq nM$. The last inequality holds since $||Gx||_\infty \leq nM\,||x||_\infty$. As $\prod_{i=1}^n \lambda_i = \det(G) = 1$, this in turn implies $\lambda_1 \geq 1/(nM)^{n-1}$. Writing $z = \sum_{i=1}^n \alpha_i v_i$, we estimate $n \geq z^\intercal G z = \sum_{i=1}^n \alpha_i^2 \lambda_i$, and hence $\alpha_i^2 \leq \frac{n}{\lambda_i} \leq (nM)^n$. Thus, $||z||^2 \leq (nM)^{n+1}$, and since $z \in \mathbb{Z}^n$, its bit complexity is bounded by $\mathcal{O}(n^2(\log(n) + \log(M)))$. □

Theorem 4.22. *The unimodular decomposition problem (and thus the rotated standard lattice problem) is in NP ∩ co-NP.*

Proof. For the sake of completeness, we start by showing containment in NP. If G is a YES-instance, the certificate is the unimodular matrix U such that $U^\intercal U = G$. Since $G_{i,i} = U_i^\intercal U_i$, where U_i is the i-th column of U, all entries in U are bounded by $\max\{\sqrt{G_{ii}} \mid i = 1, \ldots, n\}$, hence the encoding size is polynomial in the input, and verifying the certificate clearly takes only polynomial time.

If G is a NO-instance, the certificate is a vector z and we verify

1. $z^\intercal G z < n$, and
2. $e_i^\intercal G e_i \equiv_2 e_i^\intercal G z$, $i = 1, \ldots, n$.

These checks can be done in time polynomial in the encoding size of z and G. If the answer to both is *yes*, then we output that there is no unimodular matrix U with

$G = U^\intercal U$. To show that both tests can be performed in time polynomial in the input, it suffices to observe that the bit complexity of z is polynomially bounded in the bit complexity of G by Lemma 4.21.

It remains to show that such a vector z exists if and only if G is a NO-instance (i.e. the certificate exists and the conclusion is correct). By Lemma 4.20, both points together are equivalent to the fact that in any lattice $\Lambda(B)$ with $B^\intercal B = G$, the vector Bz is a characteristic vector with $||Bz||^2 \leq n - 1$. By Theorem 4.19, this is equivalent to $\Lambda(B) \ncong \mathbb{Z}^n$, which in turn is equivalent to the impossibility of a unimodular decomposition $G = U^\intercal U$ by Lemma 4.13. □

To illustrate the certificates, we consider two examples. First, consider the Gram matrix

$$G = \begin{pmatrix} 4 & -2 & & & & & & 1 & 3 \\ -2 & 2 & -1 & & & & & & -1 \\ & -1 & 2 & -1 & & & & & \\ & & -1 & 2 & -1 & & & & \\ & & & -1 & 2 & -1 & & & \\ & & & & -1 & 2 & -1 & & \\ & & & & & -1 & 2 & & 1 \\ 1 & & & & & & & 2 & 3 \\ 3 & -1 & & & & & 1 & 3 & 7 \end{pmatrix} \in \mathbb{Z}^9$$

and the vector $z = (-1, -1, -1, -1, -1, -1, -1, -1, 1)^\intercal \in \mathbb{Z}^9$ as a certificate. First, $z^\intercal G z = 1 < 9$. Moreover, we have

$$e_i^\intercal G e_i \equiv_2 \begin{cases} 0 & 1 \leq i \leq 8 \\ 1 & i = 9 \end{cases}, \quad \text{and } e_i^\intercal G z \equiv_2 \begin{cases} 0 & 1 \leq i \leq 8 \\ 1 & i = 9 \end{cases}.$$

Thus, there is no unimodular matrix U such that $G = U^\intercal U$. (The matrix G is chosen to correspond to the lattice $E_8 \times \mathbb{Z}$.)

For the second example, let us consider the Gram matrix

$$G = \begin{pmatrix} 34 & 21 \\ 21 & 13 \end{pmatrix}.$$

As discussed before, the first self-dual lattice not isomorphic to $\mathbb{Z}^n$ appears in dimension 8, hence it is no surprise that we can find a decomposition:

$$\begin{pmatrix} 34 & 21 \\ 21 & 13 \end{pmatrix} = U^\intercal U = \begin{pmatrix} 5 & 3 \\ 3 & 2 \end{pmatrix} \begin{pmatrix} 5 & 3 \\ 3 & 2 \end{pmatrix}.$$

The coefficient vector of a shortest characteristic vector is $z = (-1, 2)^\intercal$, yielding $z^\intercal G z = 2$.

In the lattice $\Lambda(U) = \mathbb{Z}^2$, this corresponds to the characteristic vector $(1,1)^\intercal$.

We have seen that characteristic vectors are well suited as a co-NP certificate for RSLP, and their coefficient vectors for UDP. If Λ is given by its basis, it follows from the discussions that the problem at hand can be solved by computing the co-set of characteristic vectors, together with a single call to an oracle for the closest vector problem (CVP), computing the shortest among all characteristic vectors. However, CVP is NP-hard, and the best known running time using this reduction we are aware of is $2^{\mathcal{O}(n)}$ (see e.g. [AS18]).

Another easy approach to RSLP is the following. If a lattice admits an orthogonal basis $\{b_1, \ldots, b_n\}$, then any basis reduced in the sense of Hermite, Korkine & Zolotareff [KZ73; Her50] is an orthogonal basis. Due to the recursive structure of those bases, n calls to an oracle for the shortest vector problem (SVP) are sufficient to find this basis. Hence the best known running time using this reduction is $2^{\mathcal{O}(n)}$ (see e.g. [AS18]), but allows to find general *orthogonal* bases. But as UDP is – from a complexity point of view – supposedly easier than SVP or CVP, it is of great interest to find a smarter algorithm. It seems to be a reappearing phenomenon that if a certain class of lattices allows to solve lattice problems such as SVP or CVP fast, a certain representation is needed. For instance, if Λ is a lattice of Voronoi's first kind, we still need to know an obtuse superbasis before using the polynomial time algorithm in [MGC14].Therefore, being able to find orthonormal, or even orthogonal bases remains an interesting open problem.

We can also ask whether characteristic vectors can help us to decide whether an *orthogonal* basis of a given lattice exists. If we generalize characteristic vectors to integral positive definite Gram matrices, it turns out that there may be several co-sets $v + 2\Lambda$ comprising characteristic vectors. Also the proof of Elkies heavily depends on self-duality, so finding an analogous, more general result will presumably need new techniques.

One might hope that the presented result implies containment of the more general lattice isomorphism problem in co-NP, using the following reduction: Given two lattice bases B_1 and B_2, apply a linear transformation A such that $A\Lambda(B_1) = \mathbb{Z}^n$, for instance $A = B_1^{-1}$. Then, decide whether $\Lambda(AB_2) \cong \mathbb{Z}^n$. For this to hold, we need that two lattices Λ_1, Λ_2 are isomorphic, if and only if the two lattices $A\Lambda_1$ and $A\Lambda_2$ are isomorphic. The following example shows that this is false in general. Consider the hexagonal lattice and a rotation thereof, i.e.

$$B_1 = \begin{pmatrix} 1 & \frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} \end{pmatrix}, \qquad O = \begin{pmatrix} \frac{\sqrt{3}}{2} & \frac{1}{2} \\ -\frac{1}{2} & \frac{\sqrt{3}}{2} \end{pmatrix} \qquad B_2 = OB_1 = \begin{pmatrix} \frac{\sqrt{3}}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{pmatrix}.$$

Clearly, $\Lambda(B_1) \cong \Lambda(B_2)$. Now we consider the linear map A given by

$$A = B_1^{-1} = \begin{pmatrix} 1 & -\frac{1}{\sqrt{3}} \\ 0 & \frac{2}{\sqrt{3}} \end{pmatrix},$$

and compare the two lattices $A\Lambda(B_1) = \mathbb{Z}^2$ and $A\Lambda(B_2)$. We obtain bases

$$B_1' = AB_1 = \mathbb{1}, \qquad B_2' = AB_2 = \begin{pmatrix} 1 & -\frac{1}{\sqrt{3}} \\ 0 & \frac{2}{\sqrt{3}} \end{pmatrix} \begin{pmatrix} \frac{\sqrt{3}}{2} & \frac{\sqrt{3}}{2} \\ -\frac{1}{2} & \frac{1}{2} \end{pmatrix} = \begin{pmatrix} \frac{2}{\sqrt{3}} & \frac{1}{\sqrt{3}} \\ -\frac{1}{\sqrt{3}} & \frac{1}{\sqrt{3}} \end{pmatrix}.$$

An easy way to see that $\Lambda(B_2') \ncong \mathbb{Z}^2 = \Lambda(B_1')$ is to verify that $\Lambda(B_2')$ contains a vector of norm less than 1, namely the second basis vector $\|(1/\sqrt{3}, 1/\sqrt{3})^\intercal\|^2 = 2/3$, whereas $\mathbb{Z}^2$ does of course not.

Bibliography

[AS18] Divesh Aggarwal and Noah Stephens-Davidowitz. "Just take the average! An embarrassingly simple 2^n-time algorithm for SVP (and CVP)". In: *1st Symposium on Simplicity in Algorithms (SOSA 2018)*. Vol. 61. OpenAccess Series in Informatics (OASIcs). Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2018, 12:1–12:19.

[AKS01] Miklós Ajtai, Ravi Kumar, and Dandapani Sivakumar. "A sieve algorithm for the shortest lattice vector problem". In: *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*. ACM, New York, 2001, pp. 601–610.

[AKS02] Miklós Ajtai, Ravi Kumar, and Dandapani Sivakumar. "Sampling short lattice vectors and the closest lattice vector problem". In: *Proceedings of the 17th IEEE Annual Conference on Computational Complexity*. IEEE, 2002, pp. 53–57.

[Ban96] Wojciech Banaszczyk. "Inequalities for convex bodies and polar reciprocal lattices in $\mathbf{R}^n$. II. Application of K-convexity". In: *Discrete Comput. Geom.* 16.3 (1996), pp. 305–311.

[BK10] Jean-Benoît Bost and Klaus Künnemann. "Hermitian vector bundles and extension groups on arithmetic schemes. I. Geometry of numbers". In: *Adv. Math.* 223.3 (2010), pp. 987–1106.

[Cha+19] Timothy Chan, Jacob W. Cooper, Martin Koutecky, Daniel Král', and Kristýna Pekárková. *Optimal matrix tree-depth and a row-invariant parameterized algorithm for integer programming*. 2019. arXiv: 1907.06688v2.

[CGG17] Karthekeyan Chandrasekaran, Venkata Gandikota, and Elena Grigorescu. "Deciding orthogonality in construction-A lattices". In: *SIAM J. Discret. Math.* 31.2 (2017), pp. 1244–1262.

[CM18] Lin Chen and Dániel Marx. "Covering a tree with rooted subtrees—parameterized and approximation algorithms". In: *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, Philadelphia, PA, 2018, pp. 2801–2820.

[Chu16] Sergei Chubanov. "A polynomial-time descent method for separable convex optimization problems with linear constraints". In: *SIAM J. Optim.* 26.1 (2016), pp. 856–889.

[CS92] John H. Conway and Neil J. A. Sloane. "Low-dimensional lattices. VI. Voronoi reduction of three-dimensional lattices". In: *Proc. Roy. Soc. London. Ser. A*. Vol. 436. 1896. 1992, pp. 55–68.

[CS99] John H. Conway and Neil J. A. Sloane. *Sphere packings, lattices and groups*. 3rd ed. Vol. 290. Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]. Springer-Verlag, New York, 1999.

[Cor+09] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to algorithms*. Third. MIT Press, Cambridge, MA, 2009.

[Cyg+15] Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, and Saket Pilipczuk Michałand Saurabh. *Parameterized algorithms*. Springer, Cham, 2015.

[Dad12] Daniel N. Dadush. "Integer programming, lattice algorithms, and deterministic volume estimation". PhD thesis. Georgia Institute of Technology, 2012.

[DB15] Daniel Dadush and Nicolas Bonifas. "Short paths on the Voronoi graph and closest vector problem with preprocessing". In: *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, Philadelphia, PA, 2015, pp. 295–314.

[DHK13] Jesús A. De Loera, Raymond Hemmecke, and Matthias Köppe. *Algebraic and geometric ideas in the theory of discrete optimization*. Vol. 14. MOS-SIAM Series on Optimization. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA; Mathematical Optimization Society, Philadelphia, PA, 2013.

[De +10] Jesús De Loera, Raymond Hemmecke, Matthias Köppe, Jon Lee, Shmuel Onn, and Robert Weismantel. *Report of focused research group: nonlinear discrete optimization*. Tech. rep. Banff International Research Station for Mathematical Innovation and Discovery, July 2010.

[DF13] Rodney G. Downey and Michael R. Fellows. *Fundamentals of parameterized complexity*. Texts in Computer Science. Springer, London, 2013.

[DGM14] Mathieu Dutour Sikirić, Viacheslav Grishukhin, and Alexander Magazinov. "On the sum of a parallelotope and a zonotope". In: *European J. Combin.* 42 (2014), pp. 49–73.

[DSV09] Mathieu Dutour Sikirić, Achill Schürmann, and Frank Vallentin. "Complexity and algorithms for computing Voronoi cells of lattices". In: *Math. Comp.* 78.267 (2009), pp. 1713–1731.

[DFK91] Martin Dyer, Alan Frieze, and Ravi Kannan. "A random polynomial-time algorithm for approximating the volume of convex bodies". In: *J. Assoc. Comput. Mach.* 38.1 (1991), pp. 1–17.

[Edm67] Jack Edmonds. "Systems of distinct representatives and linear algebra". In: *J. Res. Nat. Bur. Standards Sect. B* 71B (1967), pp. 241–245.

[Eib+19] Eduard Eiben, Robert Ganian, Dušan Knop, Sebastian Ordyniak, and Marcin Pilipczuk Michałand Wrochna. "Integer programming and incidence treedepth". In: *Integer programming and combinatorial optimization*. Vol. 11480. Lecture Notes in Comput. Sci. Springer, Cham, 2019, pp. 194–204.

[EHK18] Friedrich Eisenbrand, Christoph Hunkenschröder, and Kim-Manuel Klein. "Faster algorithms for integer programs with block structure". In: *45th International Colloquium on Automata, Languages, and Programming*. Vol. 107. Leibniz Int. Proc. Inform. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2018, Art. No. 49, 13.

[Eis+19] Friedrich Eisenbrand, Christoph Hunkenschröder, Kim-Manuel Klein, Martin Koutecký, Asaf Levin, and Shmuel Onn. *An algorithmic theory of integer programming*. 2019. arXiv: 1904.01361.

[EW18] Friedrich Eisenbrand and Robert Weismantel. "Proximity results and faster algorithms for integer programming using the Steinitz lemma". In: *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, Philadelphia, PA, 2018, pp. 808–816.

[Elk95] Noam D. Elkies. "A characterization of the $\mathbf{Z^n}$ lattice". In: *Math. Res. Lett.* 2.3 (1995), pp. 321–326.

[Eng88] Peter Engel. "Mathematical problems in modern crystallography". In: *Comput. Math. Appl.* 16.5-8 (1988), pp. 425–436.

[EMS01] Peter Engel, Louis Michel, and Marjorie Senechal. "New geometric invariants for Euclidean lattices". In: *Réseaux euclidiens, designs sphériques et formes modulaires*. Vol. 37. Monogr. Enseign. Math. Enseignement Math., Geneva, 2001, pp. 268–272.

[Erd99] Robert M. Erdahl. "Zonotopes, dicings, and Voronoi's conjecture on parallelohedra". In: *European J. Combin.* 20.6 (1999), pp. 527–549.

[ER94] Robert M. Erdahl and Sergeĭ S. Ryshkov. "On lattice dicing". In: *European J. Combin.* 15.5 (1994), pp. 459–481.

[Fom+18] Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Marcin Pilipczuk Michałand Wrochna. "Fully polynomial-time parameterized computations for graphs and matrices of low treewidth". In: *ACM Trans. Algorithms* 14.3 (2018), Art. 34, 45.

[FT87] András Frank and Éva Tardos. "An application of simultaneous Diophantine approximation in combinatorial optimization". In: *Combinatorica* 7.1 (1987), pp. 49–65.

[Fre90] Eugene C Freuder. "Complexity of K-tree structured constraint satisfaction problems". In: *Proceedings of the eighth National conference on Artificial intelligence-Volume 1*. 1990, pp. 4–9.

[GO18] Robert Ganian and Sebastian Ordyniak. "The complexity landscape of decompositional parameters for ILP". In: *Artif. Intell.* 257 (2018), pp. 61–71.

[GOR17] Robert Ganian, Sebastian Ordyniak, and Maadapuzhi S. Ramanujan. "Going beyond primal treewidth for (M)ILP". In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. AAAI Press, 2017, pp. 815–821.

[Ger04] Larry J. Gerstein. "Characteristic elements of unimodular $\mathbb{Z}$-lattices". In: *Linear Multilinear Algebra* 52.5 (2004), pp. 381–383.

[Gra75] Jack E. Graver. "On the foundations of linear and integer linear programming. I". In: *Math. Program.* 9.2 (1975), pp. 207–226.

[GS80] Victor S. Grinberg and Sergey V. Sevast'yanov. "Value of the Steinitz constant". In: *Funct. Anal. its Appl.* 14.2 (1980), pp. 125–126.

[GLS81] Martin Grötschel, László Lovász, and Alexander Schrijver. "The ellipsoid method and its consequences in combinatorial optimization". In: *Combinatorica* 1.2 (1981). Corrigendum: DOI 10.1007/BF02579139, pp. 169–197.

[GLS93] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*. 2nd ed. Vol. 2. Algorithms and Combinatorics. Springer-Verlag, Berlin, 1993.

[Gru07] Peter M. Gruber. *Convex and discrete geometry*. Vol. 336. Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]. Springer, Berlin, 2007.

[HPS11] Guillaume Hanrot, Xavier Pujol, and Damien Stehlé. "Algorithms for the shortest and closest lattice vector problems". In: *Coding and cryptology*. Vol. 6639. Lecture Notes in Computer Science. corrected version at http://perso.ens-lyon.fr/damien.stehle/downloads/SVPCVP.pdf. Springer, Heidelberg, 2011, pp. 159–190.

[HW08] G. Harold Hardy and Edward M. Wright. *An introduction to the theory of numbers*. 6th ed. Oxford University Press, Oxford, 2008.

[HR14] Ishay Haviv and Oded Regev. "On the lattice isomorphism problem". In: *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*. ACM, New York, 2014, pp. 391–404.

[HKW14] Raymond Hemmecke, Matthias Köppe, and Robert Weismantel. "Graver basis and proximity techniques for block-structured separable convex integer minimization problems". In: *Math. Program.* 145.1-2, Ser. A (2014), pp. 1–18.

[HOR13] Raymond Hemmecke, Shmuel Onn, and Lyubov Romanchuk. "n-fold integer programming in cubic time". In: *Math. Program.* 137.1-2, Ser. A (2013), pp. 325–341.

[Her50] Charles Hermite. "Extraits de lettres de M. Ch. Hermite à M. Jacobi sur différents objects de la théorie des nombres. (Continuation)". In: *J. Reine Angew. Math.* 40 (1850), pp. 279–315.

[Hin48] Aleksandr Ya. Hinčin. "A quantitative formulation of the approximation theory of Kronecker". In: *Izvestiya Akad. Nauk SSSR. Ser. Mat.* 12 (1948), pp. 113–122.

[HS90] Dorit S. Hochbaum and J. George Shanthikumar. "Convex separable optimization is not much harder than linear optimization". In: *J. Assoc. Comput. Mach.* 37.4 (1990), pp. 843–862.

[Hun19] Christoph Hunkenschröder. *Deciding whether a lattice has an orthonormal basis is in co-NP*. 2019. arXiv: 1910.03838.

[HRS20] Christoph Hunkenschröder, Gina Reuland, and Matthias Schymura. "On compact representations of Voronoi cells of lattices". In: *Math. Program.* (2020). (published online).

[JK15] Bart M. P. Jansen and Stefan Kratsch. "A structural approach to kernels for ILPs: treewidth and total unimodularity". In: *Algorithms—ESA 2015*. Vol. 9294. Lecture Notes in Comput. Sci. Springer, Heidelberg, 2015, pp. 779–791.

[Jan+19] Klaus Jansen, Kim-Manuel Klein, Marten Maack, and Malin Rau. "Empowering the configuration-IP—new PTAS results for scheduling with setups times". In: *10th Innovations in Theoretical Computer Science*. Vol. 124. Leibniz Int. Proc. Inform. 2019, Art. No. 44, 19.

[JLR19] Klaus Jansen, Alexandra Lassota, and Lars Rohwedder. "Near-linear time algorithm for n-fold ILPs via color coding". In: *46th International Colloquium on Automata, Languages, and Programming*. Vol. 132. Leibniz Int. Proc. Inform. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2019, Art. No. 75, 13.

[JR19] Klaus Jansen and Lars Rohwedder. "On integer programming and convolution". In: *10th Innovations in Theoretical Computer Science*. Vol. 124. Leibniz Int. Proc. Inform. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2019, Art. No. 43, 17.

[Kan87] Ravi Kannan. "Minkowski's convex body theorem and integer programming". In: *Math. Oper. Res.* 12.3 (1987), pp. 415–440.

[Kar72] Richard M. Karp. "Reducibility among combinatorial problems". In: *Complexity of computer computations (Proc. Sympos., IBM Thomas J. Watson Res. Center, Yorktown Heights, N.Y., 1972)*. 1972, pp. 85–103.

[Kha80] Leonid G. Khachiyan. "Polynomial algorithms in linear programming". In: *USSR Computational Mathematics and Mathematical Physics* 20.1 (1980), pp. 53–72.

[KK18] Dušan Knop and Martin Koutecký. "Scheduling meets n-fold integer programming". In: *J. Sched.* 21.5 (2018), pp. 493–503.

[KKM17a] Dušan Knop, Martin Koutecký, and Matthias Mnich. "Combinatorial n-fold integer programming and applications". In: *25th European Symposium on Algorithms*. Vol. 87. Leibniz Int. Proc. Inform. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2017, Art. No. 54, 14.

[KKM17b] Dušan Knop, Martin Koutecký, and Matthias Mnich. "Voting and bribing in single-exponential time". In: *34th Symposium on Theoretical Aspects of Computer Science*. Vol. 66. Leibniz Int. Proc. Inform. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2017, 46:1–46:14.

[KP] Dušan Knop and Marcin Pilipczuk Michałand Wrochna. "Tight complexity lower bounds for integer linear programming with few constraints". In: *36th International Symposium on Theoretical Aspects of Computer Science*. Vol. 126. LIPIcs. Leibniz Int. Proc. Inform. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, Art. No. 44, 15.

[KV00] Phokion G. Kolaitis and Moshe Y. Vardi. "Conjunctive-query containment and constraint satisfaction". In: *J. Comput. System Sci.* 61.2 (2000), pp. 302–332.

[KZ73] Alexander N. Korkine and Jegor I. Zolotareff. "Sur les formes quadratiques". In: *Math. Ann.* 6.3 (1873), pp. 366–389.

[KV18] Bernhard Korte and Jens Vygen. *Combinatorial optimization*. Vol. 21. Algorithms and Combinatorics. Theory and algorithms, 5th ed. Springer, Berlin, 2018.

[KLO18] Martin Koutecký, Asaf Levin, and Shmuel Onn. "A parameterized strongly polynomial algorithm for block structured integer programs". In: *45th International Colloquium on Automata, Languages, and Programming*. Vol. 107. Leibniz Int. Proc. Inform. 2018, 85:1–85:14.

[Kup08] Greg Kuperberg. "From the Mahler conjecture to Gauss linking integrals". In: *Geom. Funct. Anal.* 18.3 (2008), pp. 870–892.

[Lan87] Serge Lang. *Linear algebra*. 3rd. Undergraduate Texts in Mathematics. Springer-Verlag, New York, 1987.

[Lee+12] Jon Lee, Shmuel Onn, Lyubov Romanchuk, and Robert Weismantel. “The quadratic Graver cone, quadratic integer minimization, and extensions”. In: *Math. Program.* 136.2, Ser. B (2012), pp. 301–323.

[Len83] Hendrik W. Lenstra Jr. “Integer programming with a fixed number of variables”. In: *Mathematics of Operations Research* 8.4 (1983), pp. 538–548.

[LS17] Hendrik W. Lenstra and Alice Silverberg. “Lattices with symmetry”. In: *J. Cryptol.* 30.3 (2017), pp. 760–804.

[Mar03] Jacques Martinet. *Perfect lattices in Euclidean spaces*. Vol. 327. Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]. Springer-Verlag, Berlin, 2003.

[MGC14] Robby G. McKilliam, Alex Grant, and I. Vaughan L. Clarkson. “Finding a closest point in a lattice of Voronoi’s first kind”. In: *SIAM J. Discret. Math.* 28.3 (2014), pp. 1405–1422.

[Mic01] Daniele Micciancio. “The hardness of the closest vector problem with preprocessing”. In: *IEEE Trans. Inform. Theory* 47.3 (2001), pp. 1212–1215.

[MG02] Daniele Micciancio and Shafi Goldwasser. *Complexity of lattice problems*. Vol. 671. The Kluwer International Series in Engineering and Computer Science. Kluwer Academic Publishers, Boston, MA, 2002.

[MV13] Daniele Micciancio and Panagiotis Voulgaris. “A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations”. In: *SIAM J. Comput.* 42.3 (2013), pp. 1364–1391.

[MH73] John Milnor and Dale H. Husemöller. *Symmetric bilinear forms*. Ergebnisse der Mathematik und ihrer Grenzgebiete, Band 73. Springer-Verlag, New York-Heidelberg, 1973.

[Min68] Hermann Minkowski. *Geometrie der Zahlen*. Bibliotheca Mathematica Teubneriana, Band 40. Johnson Reprint Corp., New York-London, 1968.

[NO12] Jaroslav Nešetřil and Patrice Ossona de Mendez. *Sparsity*. Vol. 28. Algorithms and Combinatorics. Springer, Heidelberg, 2012.

[Onn10] Shmuel Onn. *Nonlinear discrete optimization*. Zurich Lectures in Advanced Mathematics. An algorithmic theory. European Mathematical Society (EMS), Zürich, 2010.

[Pap81] Christos H. Papadimitriou. “On the complexity of integer programming”. In: *J. Assoc. Comput. Mach.* 28.4 (1981), pp. 765–768.

[PS97] W. Plesken and B. Souvignier. “Computing isometries of lattices”. In: *J. Symb. Comput.* 24.3-4 (1997), pp. 327–334.

[PT18] Paul Pollack and Enrique Treviño. “Finding the four squares in Lagrange’s theorem”. In: *Integers* 18A (2018), Paper No. A15, 16.

[Rei+14] Felix Reidl, Peter Rossmanith, Fernando Sánchez Villaamil, and Somnath Sikdar. "A faster parameterized algorithm for treedepth". In: *Automata, languages, and programming. Part I.* Vol. 8572. Lecture Notes in Comput. Sci. Springer, Heidelberg, 2014, pp. 931–942.

[Reu18] Gina Reuland. *A compact representation of the Voronoi cell.* École Polytechnique Fédérale de Lausanne. master thesis. January 2018.

[Sch86] Alexander Schrijver. *Theory of linear and integer programming.* Wiley-Interscience Series in Discrete Mathematics. John Wiley & Sons, Ltd., Chichester, 1986.

[Ser73] Jean-Pierre Serre. *A course in arithmetic.* Translated from the French, Graduate Texts in Mathematics, No. 7. Springer-Verlag, New York-Heidelberg, 1973.

[Sey99] Martin Seysen. "A measure for the non-orthogonality of a lattice basis". In: *Combin. Probab. Comput.* 8.3 (1999), pp. 281–291.

[SFS09] Naftali Sommer, Meir Feder, and Ofir Shalvi. "Finding the closest lattice point by iterative slicing". In: *SIAM J. Discret. Math.* 23.2 (2009), pp. 715–731.

[Ste16] E. Steinitz. "Bedingt konvergente Reihen und konvexe Systeme". In: *J. Reine Angew. Math.* 146 (1916), pp. 1–52.

[Tar86] Éva Tardos. "A strongly polynomial algorithm to solve combinatorial linear programs". In: *Oper. Res.* 34.2 (1986), pp. 250–256.

[Val03] Frank Vallentin. "Sphere coverings, lattices, and tilings (in low dimensions)". PhD thesis. Technical University Munich, Germany, 2003.

[Vor08] Georges Voronoi. "Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Deuxième mémoire. Recherches sur les parallélloèdres primitifs". In: *J. Reine Angew. Math.* 134 (1908), pp. 198–287.

List of Symbols

$\mathbb{1} \in \mathbb{Z}^{n \times n}$	the identity matrix in suitable dimension, $(\mathbb{1})_{i,j} = \delta_{i,j}$, $1 \leq i, j \leq n$
$\mathbf{1} \in \mathbb{Z}^n$	the all-ones vector in suitable dimension, $(\mathbf{1})_i = 1$, $1 \leq i \leq n$
$\mathbb{N}$	the set of natural numbers
$\mathbb{Z}$	the set of integers
$\mathbb{Q}$	the set of rational numbers
$\mathbb{R}$	the set of real numbers
$\mathbb{X}_{\succ a}$	for $\mathbb{X} \in \{\mathbb{N}, \mathbb{Z}, \mathbb{Q}, \mathbb{R}\}$ and $\succ \in \{>, \geq\}$, the set $\{z \in \mathbb{X} : z \succ a\}$
$[a, b]$	the set $\{a + \lambda(b-a) : t \in \mathbb{R},\ 0 \leq t \leq 1\}$ for $a, b \in \mathbb{R}^n$
$[a : b]$	the set $\{z \in \mathbb{Z} : a \leq z \leq b\}$ for $a, b \in \mathbb{Z}$
$[n]$	the set $\{1, 2, \ldots, n\}$ for $n \in \mathbb{Z}_{\geq 1}$
$\lceil a \rceil$	for $a \in \mathbb{R}$, the smallest $z \in \mathbb{Z}$ with $z \geq a$
$\lfloor a \rfloor$	for $a \in \mathbb{R}$, the largest $z \in \mathbb{Z}$ with $z \leq a$
$\lfloor a \rceil$	for $a \in \mathbb{R}$, the integer $z \in \{\lfloor a \rfloor, \lceil a \rceil\}$ closest to a ($\lfloor a \rceil := \lceil a \rceil$ if $a \in \frac{1}{2}\mathbb{Z}$)
B^A	the set of maps $p : A \to B$ for sets A, B
$\delta_{i,j}$	Kronecker's delta, $\delta_{i,j} = 1$ if $i = j$, and $\delta_{i,j} = 0$ if $i \neq j$
$\mathcal{G}(A)$	the Graver basis of a matrix A (Section 1.6)
height(F)	for a rooted tree F, the number of vertices on a longest root-leaf path
ln	the logarithmus naturalis
log	the logarithm to the base 2
$\log_a$	the logarithm to the base $a \in \mathbb{R}_{>0}$
supp(x)	the support of $x \in \mathbb{R}^n$, i.e. the set $\{i \in [n] : x_i \neq 0\}$
td(F)	the tree-depth of a rooted tree F (Section 1.7)
td$_P(A)$	for $A \in \mathbb{Z}^{m \times n}$, the tree-depth of the primal graph, td$(G_P(A))$
td$_D(A)$	for $A \in \mathbb{Z}^{m \times n}$, the tree-depth of the dual graph, td$(G_D(A))$
tw(F)	the tree-width of a rooted tree F (Section 1.7)
tw$_P(A)$	for $A \in \mathbb{Z}^{m \times n}$, the tree-width of the primal graph, td$(G_P(A))$
tw$_D(A)$	for $A \in \mathbb{Z}^{m \times n}$, the tree-width of the dual graph, td$(G_D(A))$
$\mathcal{V}(\Lambda)$	the Voronoi cell of a lattice Λ (Section 1.4)

Christoph Hunkenschröder

christoph.hunkenschroder@epfl.ch
disopt.epfl.ch/hunkenschroeder
ORCID: 0000-0001-5580-3677

Discrete Optimization Group
EPFL MA C1 573
CH-1015 Lausanne

Research Interests

Integer Programming

Geometry of Numbers and Lattices

Combinatorial and Discrete Optimization

Education

since 2016 PhD student, École Polytechnique Fédérale de Lausanne,
Supervisor: Friedrich Eisenbrand

2015 Master of Science, final grade 1.3
Rheinische Friedrich-Wilhelms-Universität, Bonn, Germany

Thesis title: *Minimizing the Number of Lattice Points in a Polytope*
Supervisor: Nicolai Hähnle

2012 Bachelor of Science, final grade 2.4
Rheinische Friedrich-Wilhelms-Universität, Bonn, Germany

Thesis title: *Approximationsalgorithmen für 2-kantenzusammenängende aufspannende Subgraphen*
Supervisor: Jens Vygen

Teaching

2016 – 2020 Algèbre Linéaire Avancée II, Prof. Friedrich Eisenbrand,
main assistant (multiple times)

2017 Algèbre Linéaire Avancée I, Prof. Daniel Kressner, teaching assistant

2016 Convexity, Prof. Friedrich Eisenbrand, main assistant

2009 – 2015 Algorithmical Mathematics I, teaching assistant (multiple times),

Linear and Integer Optimization, Prof. Stephan Held, teaching assistant

Languages

German	native
English	fluent
French	basic

Publications

[1] Nicolas Bousquet, Yang Cai, Christoph Hunkenschröder, and Adrian Vetta. "On the economic efficiency of the combinatorial clock auction". In: *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*. ACM, New York, 2016, pp. 1407–1423.

[2] Friedrich Eisenbrand, Christoph Hunkenschröder, and Kim-Manuel Klein. "Faster algorithms for integer programs with block structure". In: *45th International Colloquium on Automata, Languages, and Programming*. Vol. 107. Leibniz Int. Proc. Inform. Schloss Dagstuhl. Leibniz-Zent. Inform., Wadern, 2018, Art. No. 49, 13.

[3] Friedrich Eisenbrand, Christoph Hunkenschröder, Kim-Manuel Klein, Martin Koutecký, Asaf Levin, and Shmuel Onn. *An algorithmic theory of Integer Programming*. 2019. arXiv: 1904.01361.

[4] Christoph Hunkenschröder. *Deciding whether a lattice has an orthonormal basis is in co-NP*. 2019. arXiv: 1910.03838.

[5] Christoph Hunkenschröder, Gina Reuland, and Matthias Schymura. "On compact representations of Voronoi cells of lattices". In: *Integer programming and combinatorial optimization*. Vol. 11480. Lecture Notes in Comput. Sci. Springer, Cham, 2019, pp. 261–274.

[6] Christoph Hunkenschröder, Santosh Vempala, and Adrian Vetta. "A 4/3-approximation algorithm for the minimum 2-edge connected subgraph problem". In: *ACM Trans. Algorithms* 15.4 (2019), Art. 55, 28.

[7] Christoph Hunkenschröder, Gina Reuland, and Matthias Schymura. "On compact representations of Voronoi cells of lattices". In: *Math. Program.* (2020).