Computing with Multi-Row Intersection Cuts

by

Álinson Santos Xavier

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Combinatorics and Optimization

Waterloo, Ontario, Canada, 2017

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner	Andrea Lodi Professor, Polytechnique Montréal
Supervisor	Ricardo Fukasawa Professor, University of Waterloo
Internal Member	William J. Cook Professor, University of Waterloo
Internal Member	Levent Tunçel Professor, University of Waterloo
Internal-external Member	Bissan Ghaddar Professor, University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners. I understand that this thesis may be made electronically available to the public.

Statement of Contributions

- Chapter 3 is based on the manuscript *The (not so) Trivial Lifting in Two Dimensions*, by Ricardo Fukasawa, Laurent Poirrier and Álinson S. Xavier, which has been submitted to *Mathematical Programming Computation* and is currently under review. Laurent Poirrier provided the instances for the computational experiments and shared feedback that helped improving the clarity of the paper.

- Chapter 5 is based on the manuscript *Intersection Cuts for Single-Row Corner Relaxations*, by Ricardo Fukasawa, Laurent Poirrier and Álinson S. Xavier, which has also been submitted to *Mathematical Programming Computation* and is currently under review. Laurent Poirrier proved the results in Section 5.3 about the upper bound on the split rank and helped with the implementation of the cut generator.

Abstract

Cutting planes are one of the main techniques currently used to solve large-scale Mixed-Integer Linear Programming (MIP) models. Many important cuts used in practice, such as Gomory Mixed-Integer (GMI) cuts, are obtained by solving the linear relaxation of the MIP, extracting a single row of the simplex tableau, then applying integrality arguments to it. A natural extension, which has received renewed attention, is to consider cuts that can only be generated when considering multiple rows of the simplex tableau simultaneously. Although the theoretical importance of such multi-row cutting planes has been proved in a number of works, their effective use in practice remains a challenge. Since the entire class of multi-row cuts proves challenging to separate, one approach to obtain them is the following. First, the integral non-basic variables are fixed to zero. Then, a lattice-free set, which induces an intersection cut, is generated. Finally, the cut coefficients for the integral non-basic variables are computed by the so-called trivial lifting procedure. In this thesis, we address some computational aspects of this approach, and we make three novel contributions. In our first contribution, we describe a small subset of multi-row intersection cuts based on the infinity norm, which works for relaxations with arbitrary numbers of rows. We present an algorithm to generate them and run extensive computational experiments to evaluate their effectiveness. We conclude that these cuts yield benefits comparable to using the entire class of multi-row cuts, but at a small fraction of the computational cost. In our second contribution, we describe a practical method for performing the trivial lifting step on relaxations with two rows. Unlike previous methods, our method is applicable to intersection cuts derived from any lattice-free set, and, for maximal lattice-free sets, it is guaranteed to run in constant time. Computational experiments confirm that the algorithm is at least two orders of magnitudes faster than current alternatives. In our final contribution, we revisit single-row relaxations containing a single integral non-basic variable, with the goal of obtaining inequalities that are not dominated by GMI cuts. The novelty in our approach is that we use the framework of intersection cuts and trivial lifting, which allows us to obtain a geometric interpretation of our cuts, a fast algorithm for generating them, and an upper bound on their split rank. We run computational experiments and conclude that, for a few instances, they close considerably more gap than GMI cuts alone.

Acknowledgments

First, I would like to express my most sincere gratitude and appreciation to my supervisor, Professor Ricardo Fukasawa, for all his guidance and patience during my graduate studies. More than a very knowledgeable mentor, Ricardo has been a great friend during these years. I could not have hoped for a better supervisor.

To Professor William J. Cook and Professor Levent Tunçel, I would like to thank you for the knowledge you shared during your courses and for your valuable support as committee members. To Professor Bissan Ghaddar and to Professor Andrea Lodi, I would like to thank you for joining my thesis committee, for taking the time to carefully read this document, and for the valuable and constructice feedback you provided.

To Professor Joseph Cheriyan and Professor Ashwin Nayak, thank you for generously donating your time and attending lectures I presented during my graduate studies. The feedback you provided me certainly helped me become a better instructor. To Laurent Poirrier, thank you for being such a great coworker. Your passion for beautiful and efficient code inspired me to improve my own programming skills.

I am also very grateful to all my colleagues and friends at the Department of Combinatorics & Optimization, at University of Waterloo, for their companionship, and for creating such a pleasant and friendly working environment. I would like to send special thanks to Ahmad Abdi, Alan Arroyo, Arman Tavakoli, Atilio Gomes, Cheolwon Heo, Cynthia Rodriguez, Julian Romero, Ningchuan Wang, Saman Lagzi, Venus Lo and Yü Feng.

To my parents and to my sister, thank you for supporting me for so many years, even over long distances, with your unconditional love. Finally, to my beloved wife, Limin Lu, thank you for being such a loving and caring partner, and for making my life brighter every day.

Dedication

This thesis is dedicated to my parents.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

A *mixed-integer linear optimization problem* (MIP) is an optimization problem in which the goal is to minimize a linear objective function subject to linear constraints and to the condition that some variables must be integral. More precisely, a MIP is a problem of the form

$$\begin{aligned} \textbf{minimize} \quad & c^T x \\ \textbf{subject to} \quad & Ax = b \\ & x \geq 0 \\ & x_i \in \mathbb{Z} \qquad \forall i \in I, \end{aligned}$$

where A is a matrix, c and b are column vectors of appropriate dimension, and I is a subset of variable indices.

Mixed-integer linear optimization started being studied as a field during the 1950s, and has become widely used in a number of industries since the late 1990s. Examples of real-world problems where MIPs are particularly useful include vehicle routing, telecommunications network design, facility location, personnel scheduling, among many others. Various important problems in engineering, medicine and biology have also been successfully modelled as MIPs [13].

One of the earliest practical methods for solving MIPs to optimality, known as *branch-and-bound*, was first described by Land and Doig [59], and is still widely used today. The idea is to solve a linear relaxation of the MIP, and, if the solution obtained is fractional, to recursively partition the problem. Early commercial solvers based on this method proved that, despite common belief at the time, enumeration techniques could, in fact, be used to solve real-world MIPs to optimality [25]. Since then, other techniques have been used in conjunction with branch-and-bound in order to improve its performance, and have dramatically improved our ability to solve large-scale problems. One such

technique, which has become one of the main ingredients of modern solvers, are *cutting planes*.

Cutting plane methods work by finding inequalities that make the linear relaxation of the MIP increasingly stronger. They have been used successfully to solve a number of combinatorial problems, including the *travelling salesman* problem [7], the *linear ordering* problem [50], *maximum cut* problems [18] and *packing problems* [51], to name but a few. Cutting planes have also been successfully integrated in most commercial MIP solvers, where they have made significant impact [26]. When the combinatorial problem that the MIP represents is known in advance, cutting planes can be generated by a careful study of the structure of the problem and the geometry of its feasible region. Since the users of commercial solvers do not always have the expert knowledge to derive cutting planes for their particular problems, however, there has also been interest in cutting planes that can be applied to any MIP, regardless of what problem is being modelled. Such cuts are known as *general-purpose* cutting planes.

The first general-purpose cutting planes were proposed by Gomory in the late 1950s [45] and were derived first by solving the linear relaxation of the MIP, extracting a single row of the optimal simplex tableau, and then applying integrality arguments to it. Although, for nearly three decades, they were deemed too numerically unstable and too dense to be usable in practice, computational experiments performed in the 1990s proved that Gomory cuts could, in fact, work very well [14]. Since then, many alternative methods for deriving general-purpose cutting planes from a single row of the simplex tableau have been proposed. Examples include two-step MIR cuts [33], k-cuts [31], interpolated subadditive cuts [43] and knapsack cuts [44].

Despite considerable research effort, the practical impact of these alternative single-row cuts has been somewhat disappointing, as none of the proposed cutting planes could significantly outperform Gomory cuts in practical settings. This led researchers to focus their attention on other classes of general-purpose cutting planes, and, particularly, on cuts that could only be derived by considering multiple rows of the simplex tableau simultaneously.

The theoretical basis for the computation of multi-row cutting planes started being developed in the late 1960s and early 1970s by Gomory and Johnson [46, 47, 48, 56], when they studied the so-called *corner polyhedron*. Given a vector $f \in \mathbb{Q}^n$ and some

vectors $r^1, \ldots, r^m \in \mathbb{Q}^m$, the corner polyhedron is the set of vectors x, s satisfying:

$$\begin{aligned} x &= f + \sum_{j=1}^{m} r^j s_j + \sum_{j=1}^{p} w^j z_j \\ x &\in \mathbb{Z}^n \\ s &\in \mathbb{R}^m_+ \\ z &\in \mathbb{Z}^p_+. \end{aligned} \tag{M}$$

In this model, the decision variables x, s and z correspond, respectively, to the basic variables, the continuous non-basic variables and integral non-basic variables, while f and $r^1, \ldots, r^m, w^1, \ldots, w^p$ correspond to the right-hand side and to the columns of the tableau. An important insight, made early on by Balas [12], was that valid inequalities for the corner polyhedron can be obtained from certain *lattice-free* sets — convex sets that do not contain any integer point in their interior — therefore reducing the problem of generating cutting planes to the geometric problem of finding convex sets that give rise to effective cuts. Valid inequalities for (M) obtained using this paradigm are known as *multi-row intersection cuts*.

After some initial interest in the 1970s, the study of multi-row intersection cuts did not receive much attention until the late 2000s, when Andersen, Louveaux, Weismantel and Wolsey [5] studied a considerably simpler version of the corner polyhedron with exactly two rows, in which the integral non-basic variables are fixed to zero. More precisely, they studied the model

$$\begin{aligned} x &= f + \sum_{j=1}^{m} r^j s_j \\ x &\in \mathbb{Z}^n \\ s &\in \mathbb{R}^m_+, \end{aligned} \tag{C}$$

for the specific case of $n = 2$. This model preserves much of the complexity of the corner polyhedron, but is considerably simpler to analyze. They showed that all inequalities that induce facets for the convex hull of (C) are intersection cuts derived from certain two-dimensional splits, triangles or quadrilaterals. This elegant result sparked a renewed interest in multi-row cutting planes. Since then, researchers have studied many variations of this relaxation [27, 32, 71, 3, 39, 40], have generalized some results for relaxations with more than two rows [27, 11], and have obtained theoretical guarantees on how strong can multi-row cuts be [6, 19, 61, 10].

The theoretical importance of multi-row intersection cuts has been proved in a number of works. Cook, Kannan and Schrijver [30] presented a simple MIP with three variables for which adding single-row split cuts iteratively fails to solve the problem, even after an infinite number of iterations. Andersen et al. [5], on the other hand, observed that the cutting plane needed to immediately solve that problem could be generated by considering two rows of the tableau simultaneously. Basu, Bonami, Cornuéjols and Margot [19] showed more examples of sets where cuts generated from two rows provide a good approximation of the integer hull, while the approximation obtained by single-row split cuts is arbitrarily bad.

On a more practical side, there has also been interest in studying the impact of using valid inequalities for the corner polyhedron as cutting planes during the solution of MIPs arising from real-world applications. Louveaux, Poirrier and Salvagnin [61] developed an exact separator for the entire class of intersection cuts that can be generated from (M), and tested it on instances from the MIPLIB. While the results were encouraging, in the sense that these intersection cuts were able to close significantly more integrality gap than single-row cuts, their exact separator was fairly slow, and therefore could not be used to solve MIPs faster. Their work also highlighted how difficult it is to separate over (M).

Other researchers [42, 64, 34, 35, 19] took a more practical approach to generate valid inequalities for the corner polyhedron, which we describe next. Starting from relaxation (M), the integral non-basic variables z are fixed to zero, and the much simpler model (C) is obtained. Then, a lattice-free set is generated in $\mathbb{R}^n$ and the corresponding intersection cut is computed. Finally, the cut coefficients for the integral non-basic variables, which had been fixed to zero, are computed by the so-called trivial lifting procedure, and a valid inequality for (M) is obtained.

In this thesis, our goal is to study computational aspects of this approach that would be necessary for the effective use of multi-row cutting planes in practice. Our research into this topic started by using the framework of multi-row intersection cuts to generate strong cuts for single-row relaxations. From early experiments it became clear that, in order for these cuts to be useful in practice, two challenges would need to be addressed: the issue of cut selection, and the issue of efficiently performing the lifting step. In this thesis, make three main novel contributions, which we summarize next.

Our first contribution is related to the selection of the lattice-free sets that will be used to generate valid inequalities for (C). Even when restricted to sets that induce facets of the convex hull of (C), the number of suitable maximal lattice-free sets is still very large, specially for instances of large size. An important problem, then, is to select

a small subset of valid inequalities for (C) that can be generated efficiently and that makes a significant impact when used as cutting planes for solving MIPs. In Chapter 4, we introduce a subset of inequalities based on the infinity norm. This subset is very small, works for relaxations having an arbitrary number of rows, and, unlike many subclasses studied in the literature, it takes into account the columns of the simplex tableau. We describe an algorithm for generating these inequalities and run extensive computational experiments in order to evaluate their effectiveness when used as cutting planes for solving real-world MIPs. We conclude that this subset of inequalities yields, in terms of gap closure, around 50% of the benefits of using all valid inequalities for (C) simultaneously, but at a small fraction of the computational cost, and with a very small number of cuts.

In our second contribution, we focus on the lifting step. Since inequalities that are valid for (C) do not take into account the integral non-basic variables, their performance deteriorates for instances where these integer variables play an important role [35]. Following Dey and Wolsey [40], given an inequality that is valid for (C), the usual approach to strengthen it has been to calculate the coefficients for the integral non-basic variables by using the so-called *trivial lifting* function. Despite its name, evaluating this function efficiently is far from trivial. The methods proposed in the literature either have high computational overhead, or are only applicable to intersection cuts coming from lattice-free sets that have very particular structure. In Chapter 3, we develop a more practical method for performing trivial lifting on relaxations with two rows. Unlike the previous alternatives, our method is applicable to intersection cuts derived from any lattice-free set, and it is guaranteed to run in constant time if the sets are maximal. We perform computational experiments, and find that our method is at least two orders of magnitude faster than alternatives.

In our final contribution, we note that, although the framework of multi-row intersection cuts has been traditionally used to study relaxations with two or more rows, it can also be used to generate strong valid inequalities for single-row relaxations. In Chapter 5, we revisit the single-row relaxation, with the goal of generating additional inequalities that are not dominated by Gomory cuts. Although single-row cuts have been extensively studied before, the novelty in our approach is that we use the framework of multi-row intersection cuts to derive our cuts, which allows us to get a geometric interpretation of the inequalities obtained. Starting from a single-row relaxation having one integral non-basic variable, we rewrite it as (C). We then obtain a precise description of the lattice-free sets that induce facets of this set, and we develop a fast algorithm for enumerating them. The algorithm also allows us to obtain an upper bound on the split

closure of (C). Finally, we run computational experiments on the MIPLIB to evaluate the impact of these cuts in practice, and the speed of our cut generator. We conclude that, for a few instances, our single-row cuts close significantly more gap than Gomory cuts alone.

Chapter 2

Background

In this chapter, we present definitions and basic results related to corner relaxations of the simplex tableau, lattice-free sets and lifting that will be used throughout this thesis. We start by introducing, in Section 2.1, the *continuous multi-row relaxation* of the simplex tableau. In Section 2.2, we introduce lattice-free sets and describe how can they be used to generate valid inequalities for the continuous multi-row relaxation. In Section 2.3, we introduce a stronger relaxation of the simplex tableau, known as the *corner polyhedron*, and we show how valid inequalities for the continuous multi-row relaxation can be transformed into strong inequalities for the corner polyhedron by a procedure known as trivial lifting. For a more detailed survey, see Conforti, Cornuéjols and Zambelli [29, Chapter 6].

2.1 Continuous multi-row relaxations

Consider a mixed-integer linear problem (MIP) given by

$$\begin{aligned} \textbf{minimize} \quad & c^T x \\ \textbf{subject to} \quad & Ax = b \\ & x \geq 0 \\ & x_i \in \mathbb{Z} \quad \forall i \in I. \end{aligned} \tag{MIP}$$

Here, and throughout this thesis, we assume that all the data is rational. Although MIPs with irrational data can also be defined, many results on mixed-integer optimization and cutting plane theory assume rationality. Let B be a subset of variable indices, and denote by A_B the submatrix obtained from A by keeping only the columns indexed by B. We recall that B is a *feasible basis* for the linear relaxation of (MIP) if A_B is an invertible matrix, and if $A_B^{-1}b$ is a vector with no negative components. Suppose that B is a

feasible basis, and let N be indices of the remaining variables. The variables x_i, for $i \in B$, are known as the *basic variables*, while the variables x_i, for $i \in N$, are known as the *non-basic variables*. Multiplying both sides of the system $Ax = b$ by A_B^{-1}, it is not hard to see that $Ax = b$ can be rewritten as

$$x_i = \bar{b}_i + \sum_{j \in N} \bar{a}_{ij} x_j \quad \forall i \in B, \tag{TBL}$$

where $\bar{b}_i \geq 0$. The system (TBL) is also known as the *simplex tableau associated with* B, and each of its equalities is commonly referred as a *row* of the simplex tableau. The *basic feasible solution* x^* associated with this tableau is given by setting $x_i^* = \bar{b}_i$, for every $i \in B$, and $x_i^* = 0$, for every $i \in N$. Note that, while x^* is feasible for the linear relaxation of (MIP), it may not be feasible for the original problem, since there may some fractional $\bar{b}_i$, where $i \in I$. In this case, our goal is to find valid inequalities for (MIP) that cut off x^*. Next, we describe one approach, which focuses on a specific relaxation of (TBL).

Starting from the constraints of (MIP), we discard the non-negativity constraints of all the basic variables, in addition to the integrality constraints of all the non-basic variables, to obtain

$$\begin{aligned} x_i &= \bar{b}_i + \sum_{j \in N} \bar{a}_{ij} x_j \quad && \forall i \in B, \\ x_i &\geq 0 && \forall i \in N, \\ x_i &\in \mathbb{Z} && \forall i \in B \cap I. \end{aligned} \tag{2.1}$$

This system can be simplified even further by the observation that the variables x_i, for $i \in B \setminus I$, can assume any real value, and that this value is completely determined by the values of the remaining variables. Indeed, if we have a solution for the following system, which was obtained from (2.1) by dropping the variables x_i, for $i \in B \setminus I$, along with all the equations where these variables appear, then obtaining a solution for (2.1) is trivial:

$$\begin{aligned} x_i &= \bar{b}_i + \sum_{j \in N} \bar{a}_{ij} x_j \quad && \forall i \in B \cap I \\ x_i &\geq 0 && \forall i \in N \\ x_i &\in \mathbb{Z} && \forall i \in B \cap I. \end{aligned} \tag{2.2}$$

Now we rewrite (2.2) in a more clean way. We may assume, without loss of generality,

that $B \cap I = \{1, \ldots, n\}$ and that $N = \{1, \ldots, m\}$. To simplify our notation, let $f = (\bar{b}_1, \ldots, \bar{b}_n)$, and let $r^j = (\bar{a}_{1j}, \ldots, \bar{a}_{nj})$, for $j = 1, \ldots, m$. System (2.2) can be rewritten as

$$\begin{aligned} x &= f + \sum_{j=1}^{m} r^j s_j \\ x &\in \mathbb{Z}^n \\ s &\in \mathbb{R}^m_+. \end{aligned} \tag{C}$$

Relaxation (C) is known as the *finite continuous multi-row relaxation* of the simplex tableau, and it is one of the main models studied throughout this thesis. As shown in the derivation above, the x and s variables correspond, respectively, to the basic and non-basic variables, while f and $r^1, \ldots, r^m$ correspond, respectively, to the right-hand side and to the columns of the tableau. The basic feasible solution associated with the tableau corresponds to the point $(x^*, s^*) = (f, 0)$. Note that, if some component of f is fractional, then (x^*, s^*) is not a feasible solution to (C). This relaxation, therefore, can be used to generate valid inequalities that cut off the fractional basic feasible solution.

The finite continuous multi-row relaxation is a simplified version of the Corner Polyhedron, introduced by Gomory [46] in the late 1960s, and was studied by Andersen, Louveaux, Weismantel and Wolsey [5] and subsequently Cornuéjols and Margot [32] for the two-row case ($n = 2$).

Another relaxation, closely related to (C), is the *infinite continuous multi-row relaxation*. Here, we have a finite number of basic variables but an infinite number of non-basic variables. More precisely, the relaxation is given by

$$\begin{aligned} x &= f + \sum_{r \in \mathbb{Q}^n} r s_r \\ x &\in \mathbb{Z}^n \\ s_r &\in \mathbb{R}_+ \qquad \forall r \in \mathbb{Q}^n \\ s &\text{ has finite support.} \end{aligned} \tag{C$_\infty$}$$

In this relaxation, s is an infinite dimensional vector, and the condition that s has finite support means that only a finite number of its components are allowed to be non-zero. Note that (C) can be obtained from (C$_\infty$) by fixing all, but a finite number of s_r variables to zero. One advantage of the infinite relaxation, when compared to the finite relaxation, is that it only depends on the vector $f \in \mathbb{Q}^n$, and not on the particular values of r^j, and therefore can be more easily studied. Valid inequalities for the infinite multi-row

relaxation are usually described in terms of the so-called valid functions. A function $\psi : \mathbb{Q}^n \to \mathbb{R}$ is *valid* for (C_∞) if the inequality

$$\sum_{r \in \mathbb{Q}^n} \psi(r) s_r \geq 1 \tag{2.3}$$

is satisfied by every vector (x, s) that satisfies (C_∞). Thus, valid functions provide coefficients of valid inequalities for (C_∞). A valid function ψ is *minimal* if there is no valid function ψ' distinct from ψ such that such that $\psi' \leq \psi$. Since the variables s_r are non-negative, minimal functions lead to stronger valid inequalities than non-minimal ones. Borozan and Cornuéjols [27] studied the properties of minimal valid functions. As we will see in the next section, they can be obtained in a geometric way.

2.2 Lattice-free sets and intersection cuts

In this section, we introduce the concepts of *lattice-free sets* and *gauge function*, and we show how they can be used to generate minimal valid functions for the continuous multi-row relaxation.

A convex set is called *lattice-free* if it does not contain any integral points in its interior, and *maximal* lattice-free if it is not properly contained into any other lattice-free set. This concept was introduced by Lovász [62]. In two dimensions, full-dimensional maximal lattice-free sets are well understood, and, following Dey and Wolsey [37], they have been classified as follows.

Proposition 1 ([37, 35])**.** *Every full-dimensional latice-free set in* $\mathbb{R}^2$ *is either:*

(i) A split set $\{(x_1, x_2) : b \leq a_1 x_1 + a_2 x_2 \leq b + 1\}$*, where* a_1 *and* a_2 *are co-prime integers and* b *is an integer;*

(ii) A type-1 triangle — a triangle with integral vertices and exactly one integral point in the relative interior of each edge;

(iii) A type-2 triangle — a triangle with at least one fractional vertex v*, exactly one integral point in the relative interior of the two edges incident to* v*, and at least two integral points on the third edge;*

(iv) A type-3 triangle — a triangle with exactly three integral points on the boundary, one in the relative interior of each edge; or

(v) A quadrilateral containing exactly one integral point in the relative interior of each of its edges.

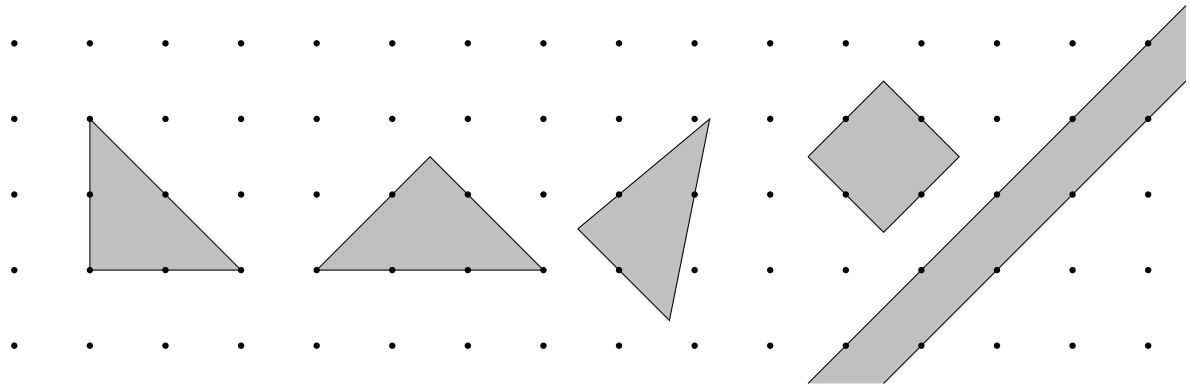


Figure 2.1: Full-dimensional maximal lattice-free sets in $\mathbb{R}^2$.

Figure 2.1 illustrates the maximal lattice-free sets in $\mathbb{R}^2$. In higher dimensions, it is known that maximal lattice-free sets in $\mathbb{R}^n$ are polyhedra with at most 2^n facets [41, 24, 68]. A more precise classification of these sets, however, is still an active research topic [11].

Given a closed convex set $K \subseteq \mathbb{R}^n$ containing the origin in its interior, the *gauge* function $\psi_K : \mathbb{Q}^n \to \mathbb{R}$ of K is defined as

$$\psi_K(r) = \inf \left\{ t > 0 : \frac{r}{t} \in K \right\}.$$

Since the origin belongs to the interior of K, the function ψ_K is finite everywhere. Furthermore, $\psi_K(r) \leq 1$ if and only if $r \in K$. When r belongs to the recession cone of K, that is, when $\frac{r}{t} \in K$, for every $t > 0$, we have $\psi_K(r) = 0$. It is not hard to prove that ψ_K is convex, sublinear and positively homogeneous. When K is a rational polyhedron, we can write

$$K = \left\{ x \in \mathbb{R}^n : a^i x \leq 1, i = 1, \ldots, k \right\},$$

where k is the number of facets of K and $a^1, \ldots, a^k \in \mathbb{Q}^n$ are row vectors. The gauge function of K, in this case, is given by the simple formula

$$\psi_K(r) = \max_{i=1,\ldots,k} a^i r.$$

Since, in this thesis, we only deal with polyhedral lattice-free sets, this is the formula that we will commonly use to compute the gauge function.

Our interest in lattice-free sets and the gauge function comes from the fact that they can be used to generate minimal valid functions for the infinite continuous multi-row relaxation. Indeed, Borozan and Cornuéjols [27] proved that, if B is a full-dimensional lattice free set containing f in its interior, then the gauge function of $B - f$ is a valid

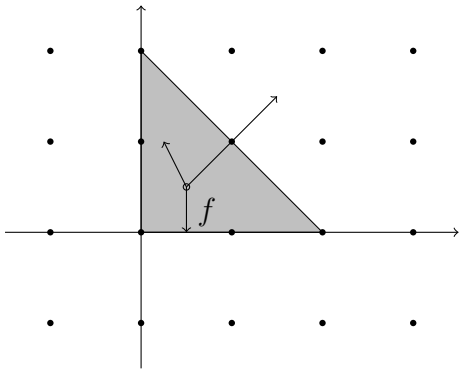


Figure 2.2: Example of intersection cut

function, and therefore

$$\sum_{r \in \mathbb{Q}^n} \psi_{B-f}(r) s_r \geq 1 \tag{2.4}$$

is a valid inequality for (C_∞). Conversely, they also prove that, if $f \in \mathbb{Q}^n \setminus \mathbb{Z}^n$ and $\psi : \mathbb{Q}^n \to \mathbb{R}$ is a minimal valid function, then the closure of the set $\{x \in \mathbb{Q}^n : \psi(x - f) \leq 1\}$ is a full-dimensional maximal lattice-free set containing f.

Using the results above, we can use the following approach to generate valid inequalities for the finite continuous multi-row relaxation that cuts off the fractional point $(f, 0)$. First, we consider the infinite relaxation (C_∞) of (C). Then, we construct a lattice-free set containing f in its interior, preferably maximal, and since (2.4) is a valid inequality for (C_∞), then

$$\sum_{i=1}^{m} \psi_{B-f}(r^i) s_i \geq 1 \tag{2.5}$$

must be valid for (C). The approach of computing cuttings planes through lattice-free sets was first introduced by Balas [12], and cuts obtained in such way are usually called *intersection cuts*. Below we show a concrete example of this approach.

Example 2. *Consider the following continuous two-row relaxation of the simplex tableau.*

$$\begin{aligned} x &= \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \end{pmatrix} s_1 + \begin{pmatrix} 0 \\ -\frac{1}{2} \end{pmatrix} s_2 + \begin{pmatrix} -\frac{1}{6} \\ \frac{1}{2} \end{pmatrix} s_3 \\ x &\in \mathbb{Z}^2 \\ s &\in \mathbb{R}^3_+. \end{aligned} \tag{2.6}$$

We want to obtain a valid inequality for (2.6) that cuts off the fractional point $(x, s) = \left(\frac{1}{2}, \frac{1}{2}, 0, 0, 0\right)$. *Using the intersection cut approach, we can construct a lattice-free set* $B \subseteq \mathbb{R}^2$ *that contains the point* $f = \left(\frac{1}{2}, \frac{1}{2}\right)$ *in its interior, such as the type-1 triangle*

shown in Figure 2.2. In this case,

$$B = \left\{ x \in \mathbb{R}^2 : \begin{array}{r} x_1 \geq 0 \\ x_2 \geq 0 \\ x_1 + x_2 \leq 2 \end{array} \right\}.$$

In order to obtain the gauge function of $B - f$, we rewrite the constraints that define B in the form $a^T(x - f) \leq 1$:

$$B = \left\{ x \in \mathbb{R}^2 : \begin{pmatrix} -2 & 0 \\ 0 & -2 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 - \frac{1}{2} \\ x_2 - \frac{1}{2} \end{pmatrix} \leq 1 \right\}.$$

The gauge function of $B - f$ is given by $\psi_{B-f}(r) = \max\{-2r_1, -2r_2, r_1 + r_2\}$, which implies

$$\psi_{B-f}\begin{pmatrix} 1 \\ 1 \end{pmatrix} = 2 \qquad \psi_{B-f}\begin{pmatrix} 0 \\ -\frac{1}{2} \end{pmatrix} = 1 \qquad \psi_{B-f}\begin{pmatrix} -\frac{1}{6} \\ \frac{1}{2} \end{pmatrix} = \frac{1}{3}.$$

We conclude that the following inequality is valid for (2.6):

$$2s_1 + s_2 + \frac{1}{3}s_3 \geq 1. \qquad \diamond$$

2.3 Mixed-integer multi-row relaxations

Throughout this thesis, we also consider a stronger relaxation of the simplex tableau, where the integrality of some non-basic variables is preserved. More precisely, given $f \in \mathbb{Q}^n \setminus \mathbb{Z}^n$, $r^1, \ldots, r^m \in \mathbb{Q}^n \setminus \{0\}$ and $w^1, \ldots, w^p \in \mathbb{Q}^n \setminus \{0\}$, we consider the *finite mixed-integer multi-row relaxation*, given by

$$\begin{aligned} x &= f + \sum_{j=1}^{m} r^j s_j + \sum_{j=1}^{p} w^j z_j \\ x &\in \mathbb{Z}^n \\ s &\in \mathbb{R}^m_+ \\ z &\in \mathbb{Z}^p_+. \end{aligned} \tag{M}$$

Introduced in the late 1960s by Gomory [46], this relaxation is also known as the *corner polyhedron*[1]. It is considerably stronger than the continuous relaxations, since the integrality of the non-basic variables has been preserved, but also considerably more complex to study.

Similarly to previous sections, an infinite version of this relaxation can also be defined, where we have an infinite number of non-basic variables. Given $f \in \mathbb{Q}^n$, the *infinite mixed-integer multi-row relaxation* is given by

$$\begin{aligned} x &= f + \sum_{r \in \mathbb{Q}^n} r s_r + \sum_{w \in \mathbb{Q}^n} w z_w \\ x &\in \mathbb{Z}^n \\ s_r &\in \mathbb{R}_+ \qquad \forall r \in \mathbb{Q}^n \\ z_w &\in \mathbb{Z}_+ \qquad \forall w \in \mathbb{Q}^n \\ s, z &\text{ have finite support.} \end{aligned} \tag{M_∞}$$

This relaxation was first studied by Johnson [56]. As before, a great advantage of relaxation (M_∞), in comparison to (M), is that it only depends on f. Furthermore, (M) can be obtained from (M_∞) by setting all but a finite number of s and z variables to zero, therefore inequalities that are valid for (M_∞) are also valid for (M).

One strategy for obtaining valid inequalities for (M_∞) is through *lifting*. This technique was first introduced by Padberg [66] for a specific combinatorial problem and later generalized to other polyhedra [69, 72, 17]. In this approach, we start from an inequality

$$\sum_{r \in \mathbb{Q}^n} \psi(r) s_r \geq 1, \tag{2.7}$$

where ψ is a minimal valid function for (C_∞), and we need to find a function $\pi : \mathbb{Q}^n \to \mathbb{R}$ such that

$$\sum_{r \in \mathbb{Q}^n} \psi(r) s_r + \sum_{r \in \mathbb{Q}^n} \pi(r) z_r \geq 1 \tag{2.8}$$

is satisfied by every (x, s, z) in (M_∞). Such a function π is a *lifting* of ψ, and the pair (ψ, π) is known as a *cut-generating pair*. A lifting π is minimal if there does not exist any distinct lifting π' such that $\pi' \leq \pi$. As before, we are mainly interested in minimal liftings of ψ, since these functions lead to valid inequalities with the strongest possible

[1] In the literature, the term *corner polyhedron* has been used to denote different variations of (M). In particular, it has been used to denote both relaxations where all non-basic variables are integer, as well as relaxations where some non-basic variables are allowed to be continuous. For this reason, we prefer the less ambiguous term *mixed-integer multi-row relaxation*.

coefficients. Many authors have studied properties of minimal liftings [56, 9, 20, 22, 28, 39, 40]. We note that minimal liftings are not necesseraly unique. That is, there may exist multiple functions π such that π is a minimal lifting of ψ. Enumerating all such functions remains a computational challenge.

A practical method for obtaining a lifting of ψ (not necesseraly minimal) was proposed by Gomory and Johnson [47] and Balas and Jeroslow [15]. They proved that, if π is defined as

$$\pi(w) = \inf_{k \in \mathbb{Z}^n} \psi(w + k), \tag{2.9}$$

then π is a lifting of ψ. This function is called the *trivial lifting* of ψ. It has been proved that, in some cases, the trivial lifting is a minimal lifting. In other words, in those cases, the computationally cheaper trivial lifting actually yields the best possible cut coefficients. Moreover, even when this is not the case, one important advantage of the trivial lifting is that it is *sequence independent*, which means that the order in which the coefficients are computed is irrelevant. The idea of potentially losing coefficient strength in order to obtain a more computationally tractable sequence-independent lifting has been applied to other valid inequalities for MIPs, like flow cover inequalities [52]. In Chapter 3, we explore the computational aspects of evaluating the trivial lifting function.

Chapter 3

Efficient Trivial Lifting in Two Dimensions

In Section 2.3, we discussed how valid inequalities for the mixed-integer multi-row relaxation can, theoretically, be obtained by modifying inequalities that are valid for the continuous relaxation, via trivial lifting. In this chapter, we focus on the computational aspects of this procedure, and we propose a practical algorithm for performing trivial lifting on relaxations with two rows.

Consider the mixed-integer multi-row relaxation of the simplex tableau, given by

$$\begin{aligned} x &= f + \sum_{j=1}^{m} r^j s_j + \sum_{j=1}^{p} w^j z_j \\ x &\in \mathbb{Z}^n \\ s &\in \mathbb{R}^m_+ \\ z &\in \mathbb{Z}^p_+ \end{aligned} \tag{M}$$

where $f \in \mathbb{Q}^n \setminus \mathbb{Z}^n$ and $r^1, \ldots, r^m, w^1, \ldots, w^p \in \mathbb{Q}^n \setminus \{0\}$. As discussed in Section 2.3, one strategy to obtain a strong valid inequality for (M) is to start from valid inequalities for the continuous multi-row relaxation and strengthen them via trivial lifting. More specifically, we perform the following steps. First, all the z_j variables in (M) are fixed to zero, and we obtain

$$\begin{aligned} x &= f + \sum_{j=1}^{m} r^j s_j \\ x &\in \mathbb{Z}^n \\ s &\in \mathbb{R}^m_+ \end{aligned} \tag{C}$$

which is a continuous multi-row relaxation of the simplex tableau. As discussed in

Section 2.2, strong valid inequalities for (C) can be easily generated from lattice-free sets. The second step, then, is to find a lattice-free set $B \subseteq \mathbb{R}^n$ containing f in its interior. If ψ is the gauge function of $B - f$, then

$$\sum_{j=1}^{m} \psi(r^j) s_j \geq 1 \tag{3.1}$$

is a valid inequality for (C). Note that (3.1) is not necessarily valid for (M), since all the integral non-basic variables z received coefficient zero. An important final step is to modify (3.1), by finding a function $\pi : \mathbb{Q}^n \to \mathbb{Q}$ such that

$$\sum_{j=1}^{m} \psi(r^j) s_j + \sum_{j=1}^{p} \pi(w^j) z_j \geq 1 \tag{3.2}$$

is a valid inequality for (M). As discussed in Section 2.3, one possible choice of π is the so-called *trivial lifting* of ψ, defined as

$$\pi(w) = \inf_{k \in \mathbb{Z}^n} \psi(w + k). \tag{3.3}$$

Despite its name, evaluating the trivial lifting function π efficiently is far from trivial. This is particularly true if put into the context of where the problem arises. It is solved once for every integer variable that needs to be lifted within a cut, so potentially thousands of times per cut. In addition, if one thinks that several cuts are to be generated and that the cutting-plane generation is just one small step in the whole solution process of a MIP, this can quickly become an impractical problem to solve. Next, we discuss some approaches that have been proposed in the literature.

A naive approach to solve (3.3) is to evaluate $\psi(w + k)$ for every $k \in \mathbb{Z}^n$ such that $\|k\|$ is smaller than a fixed constant, chosen before the cut generation procedure starts. For the two-row case, Dey and Wolsey [40] proved that, if the constant is large enough, then this procedure finds the correct answer. The number of points k that must be considered, however, is not bounded by any constant and can be potentially very large.

Instead of evaluating ψ at many points, Espinoza [42] evaluates this function exactly once, at a point selected heuristically, with no guarantee that the exact trivial lifting coefficient is obtained. More specifically, for each ray $w \in \mathbb{Q}^p$, he evaluates $\psi(w + k)$ for

$$k = -(\lfloor w_1 \rfloor, \ldots, \lfloor w_p \rfloor).$$

The motivation for the choice of k comes from the fact that ψ, being a gauge function,

tends to assign smaller values to shorter vectors. In later computational experiments, Musalem [64] used the same strategy. While computationally friendly, this approach is not guaranteed to find the correct answer, and produces inequalities that are not as strong as they could be.

Basu, Bonami, Cornuéjols and Margot [19] focused on the specific case where the lattice-free set B is either a type-1 or type-2 triangle, and derived a closed formula for evaluating the trivial lifting function π. The idea is that, for these lattice-free sets, the infimum on (3.3) must be attained in one of six candidate choices of k. The formula evaluates ψ at these six points and takes the minimum. This result was also used by Dey, Lodi, Tramontani and Wolsey [34, 35] in later computational experiments. The drawback with this approach, however, is that it greatly limits the range of lattice-free sets that can be experimented with.

Finally, it is worth mentioning that, in principle, this problem can be solved in polynomial time for fixed n, since it is a minimization of a convex function over integer points in polyhedra [65], or modeled as a mixed-integer program by adding a continuous variable to handle the convex piecewise linear objective function which leads to another polynomial time algorithm for fixed n [9]. Such approaches, however, rely on the solution of the feasibility problem in fixed dimension as a subroutine to solve the optimization problem. Solving one such MIP for each cut coefficient would likely suffer from significant computational overhead, even if a fast fixed-dimension MIP solver implementation were available.

In this chapter, we develop a more practical method for evaluating the trivial lifting function in two dimensions, which requires significantly fewer queries to the function ψ than the naive procedure and that does not have significant computational overhead. For maximal lattice-free sets, we prove that the algorithm is guaranteed to terminate in constant time, and for the cases where the closed formula described in [19] is applicable, we show that it requires the same number of evaluations to the function ψ as the closed formula. We also obtain an upper bound on the number of evaluations for non-maximal lattice-free sets, which depends on the lattice-width of the set and on its second covering minimum. Finally, we run computational experiments to confirm that the algorithm works well in practice. The proposed method can also easily be adapted to solve the lifting problem in higher dimensions, though in that context we do not have theoretical or computational evaluations of its performance.

3.1 Main algorithm

In this section, we describe an alternative algorithm for the computation of trivial lifting coefficients on two-dimensional lattice-free sets. More specifically, let $B \subseteq \mathbb{R}^2$ be a convex lattice-free set containing $f \in \mathbb{Q}^2 \backslash \mathbb{Z}^2$ in its interior, and suppose $\psi : \mathbb{Q}^2 \to \mathbb{R}$ is the gauge function of $B - f$. For any $w \in \mathbb{Q}^2$, our goal is to solve the minimization problem (3.3).

The following propositions give us the two main ideas behind the algorithm. The first proposition shows that, if one component of k is fixed at any value (without loss of generality, we fix k_2), then the minimization problem becomes significantly easier.

Proposition 3. *Let $\bar{k}_2 \in \mathbb{R}$ and $w \in \mathbb{Q}^2$. If k_1^* is a solution for*

$$\min_{k_1 \in \mathbb{R}} \psi \begin{pmatrix} w_1 + k_1 \\ w_2 + \bar{k}_2 \end{pmatrix} \tag{3.4}$$

then a solution for

$$\min_{k_1 \in \mathbb{Z}} \psi \begin{pmatrix} w_1 + k_1 \\ w_2 + \bar{k}_2 \end{pmatrix} \tag{3.5}$$

is either $\lfloor k_1^* \rfloor$ or $\lceil k_1^* \rceil$.

Proof. This follows directly from the fact that ψ is a univariate convex function. ■

Therefore, given a solution for the continuous problem (3.4), we can easily determine an optimal solution for the integer problem (3.5). Note that (3.4) can be solved efficiently, for example, by modeling it as an LP. Since these ideas will be used throughout, it will be convenient to define the following notation

$$g(\bar{\alpha}_2) := \min_{\alpha_1 \in \mathbb{R}} \psi \begin{pmatrix} \alpha_1 \\ \bar{\alpha}_2 \end{pmatrix}$$

$$h(\bar{k}_2) := \min_{k_1 \in \mathbb{Z}} \psi \begin{pmatrix} w_1 + k_1 \\ w_2 + \bar{k}_2 \end{pmatrix}.$$

The second proposition shows that, if the second component of k is fixed at a number with very large magnitude, either positive of negative, then the optimal value also becomes very large. Therefore, these values of k_2 may be safely ignored. Note that the constants ζ^+ and ζ^- that appear in the statement of the proposition do not depend on w or k, but only on the definition of the function ψ.

Proposition 4. *Let $\bar{k}_2 \in \mathbb{R}$ and $w \in \mathbb{Q}^2$. If $\bar{k}_2$ is a positive integer such that $\bar{k}_2 > |w_2|$,*

then

$$\min_{k_1\in\mathbb{Z}} \psi \begin{pmatrix} w_1+k_1 \\ w_2+\bar{k}_2 \end{pmatrix} \geq \zeta^+(w_2+\bar{k}_2)$$
$$\min_{k_1\in\mathbb{Z}} \psi \begin{pmatrix} w_1+k_1 \\ w_2-\bar{k}_2 \end{pmatrix} \geq \zeta^-(\bar{k}_2-w_2)$$

where $\zeta^+ = \min_{\alpha\in\mathbb{R}} \psi \left(\begin{smallmatrix} \alpha \\ 1 \end{smallmatrix}\right)$ *and* $\zeta^- = \min_{\alpha\in\mathbb{R}} \psi \left(\begin{smallmatrix} \alpha \\ -1 \end{smallmatrix}\right)$.

Proof. First, note that

$$\min_{k_1\in\mathbb{Z}} \psi \begin{pmatrix} w_1+k_1 \\ w_2+\bar{k}_2 \end{pmatrix} \geq \min_{k_1\in\mathbb{R}} \psi \begin{pmatrix} w_1+k_1 \\ w_2+\bar{k}_2 \end{pmatrix} = \min_{\alpha\in\mathbb{R}} \psi \left(\begin{smallmatrix} \alpha \\ w_2+\bar{k}_2 \end{smallmatrix}\right)$$

since the integer problem is a restriction of the continuous one, and we can let $\alpha := w_1 + k_1$. Then, because ψ is positively homogeneous and $w_2 + \bar{k}_2 > 0$, we have

$$\min_{\alpha\in\mathbb{R}} \psi \left(\begin{smallmatrix} \alpha \\ w_2+\bar{k}_2 \end{smallmatrix}\right) = (w_2+\bar{k}_2) \min_{\alpha\in\mathbb{R}} \psi \left(\begin{smallmatrix} \alpha \\ 1 \end{smallmatrix}\right) = \zeta^+(w_2+\bar{k}_2).$$

To obtain the second inequality, we proceed similarly. Since $\bar{k}_2 - w_2 > 0$, we have

$$\min_{k_1\in\mathbb{Z}} \psi \begin{pmatrix} w_1+k_1 \\ w_2-\bar{k}_2 \end{pmatrix} \geq \min_{\alpha\in\mathbb{R}} \psi \left(\begin{smallmatrix} \alpha \\ w_2-\bar{k}_2 \end{smallmatrix}\right) = \zeta^-(\bar{k}_2-w_2).$$

■

Algorithm 5 Trivial Lifting

function TRIVIALLIFTING(B, f, w)
  Let $g(\bar{\alpha}_2) := \min_{\alpha_1\in\mathbb{R}} \psi \left(\begin{smallmatrix} \alpha_1 \\ \bar{\alpha}_2 \end{smallmatrix}\right)$
  Let $h(\bar{k}_2) := \min_{k_1\in\mathbb{Z}} \psi \begin{pmatrix} w_1+k_1 \\ w_2+\bar{k}_2 \end{pmatrix}$
  $\zeta^+, \zeta^- \leftarrow g(1), g(-1)$
  $\eta^* \leftarrow h(0)$
  $\bar{k}_2 \leftarrow 1$
  repeat
    $\eta^* \leftarrow \min\left\{\eta^*, h(\bar{k}_2), h(-\bar{k}_2)\right\}$
    $\bar{k}_2 \leftarrow \bar{k}_2 + 1$
  until $\left(\bar{k}_2 > |w_2|$ **and** $w_2 + \bar{k}_2 > \frac{\eta^*}{\zeta^+}$ **and** $\bar{k}_2 - w_2 > \frac{\eta^*}{\zeta^-}\right)$
  return η^*

Given B, f and w, the function TRIVIALLIFTING described in Algorithm 5 computes the optimum value of (3.3). At each iteration, it solves the two optimization problems

$$\min_{k_1\in\mathbb{Z}} \psi \begin{pmatrix} w_1+k_1 \\ w_2+\bar{k}_2 \end{pmatrix} \text{ and } \min_{k_1\in\mathbb{Z}} \psi \begin{pmatrix} w_1+k_1 \\ w_2-\bar{k}_2 \end{pmatrix},$$

for some fixed value $\bar{k}_2$, starting from zero, and going up. By Proposition 3, these two problems can be easily solved. The algorithm also keeps track of the smallest optimal value found so far, in the variable η^*. It stops when $\bar{k}_2$ is such that three conditions are satisfied:

$$\bar{k}_2 > |w_2| \textbf{ and } w_2 + \bar{k}_2 > \frac{\eta^*}{\zeta^+} \textbf{ and } \bar{k}_2 - w_2 > \frac{\eta^*}{\zeta^-}. \tag{3.6}$$

This is justified by Proposition 4. Indeed, if $\bar{k}_2$ is such that condition (3.6) holds, then

$$\begin{aligned}
\min_{k_1 \in \mathbb{Z}} \psi \begin{pmatrix} w_1 + k_1 \\ w_2 + \bar{k}_2 \end{pmatrix} &\geq \zeta^+ (w_2 + \bar{k}_2) \geq \zeta^+ \frac{\eta^*}{\zeta^+} = \eta^* \\
\min_{k_1 \in \mathbb{Z}} \psi \begin{pmatrix} w_1 + k_1 \\ w_2 - \bar{k}_2 \end{pmatrix} &\geq \zeta^- (\bar{k}_2 - w_2) \geq \zeta^- \frac{\eta^*}{\zeta^-} = \eta^*.
\end{aligned}$$

Therefore, by considering any such $\bar{k}_2$, the incumbent value η^* can never be improved. Also note that, if $\bar{k}_2$ is sufficiently large, then condition (3.6) is automatically satisfied. Indeed, since $\eta^* \geq h(0)$, then

$$\bar{k}_2 > \max \left\{ \frac{h(0)}{\zeta^+} - w_2, \frac{h(0)}{\zeta^-} + w_2, |w_2| \right\}$$

implies that the condition holds. Therefore, the algorithm will always terminate.

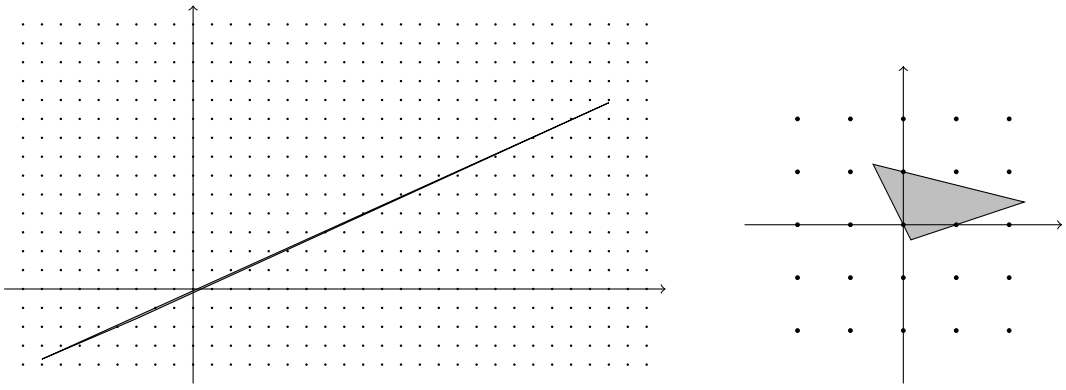
3.2 Preprocessing step

Although finite, the algorithm described in Section 3.1 may require a large number of iterations to terminate. In this subsection, we describe a preprocessing step that, when executed prior to the algorithm, greatly improves its worst-case performance. We start with an example that illustrates this fact.

Example 6. *To illustrate how pre-processing can improve the efficiency of Algorithm 5, let B, w and f be defined as*

$$B = \operatorname{conv} \left\{ \begin{pmatrix} 22 \\ \frac{69}{7} \end{pmatrix}, \begin{pmatrix} -3 \\ -\frac{11}{7} \end{pmatrix}, \begin{pmatrix} -8 \\ -\frac{26}{7} \end{pmatrix} \right\} \quad f = \begin{pmatrix} \frac{2}{3} \\ \frac{1}{6} \end{pmatrix} \quad w = \begin{pmatrix} \frac{2}{3} \\ \frac{1}{3} \end{pmatrix}.$$

The set B, illustrated in Figure 3.1a, is very long and thin, which causes performance problems for both the naive algorithm, as well as the algorithm described previously. In fact, Algorithm 5 requires seven iterations of the main loop to output the optimal value of $\frac{4}{5}$. Before feeding this data into the algorithm however, we could apply to B, w and f



(a) Original set B (b) Modified set $\bar{B}$

Figure 3.1: Example of pre-processing.

the affine unimodular transformation $\tau : \mathbb{R}^2 \to \mathbb{R}^2$ *defined as*

$$\tau(x) = \begin{pmatrix} 1 & -2 \\ -5 & 11 \end{pmatrix} x + \begin{pmatrix} 4 \\ 2 \end{pmatrix}.$$

Let $\bar{B}, \bar{f}, \bar{w}$ *be the set and vectors obtained, namely*

$$\bar{B} = \operatorname{conv}\left\{ \begin{pmatrix} \frac{16}{7} \\ \frac{3}{7} \end{pmatrix}, \begin{pmatrix} \frac{1}{7} \\ -\frac{2}{7} \end{pmatrix}, \begin{pmatrix} -\frac{4}{7} \\ \frac{8}{7} \end{pmatrix} \right\} \quad \bar{f} = \begin{pmatrix} \frac{1}{3} \\ \frac{1}{2} \end{pmatrix} \quad \bar{w} = \begin{pmatrix} \frac{0}{3} \\ \frac{1}{3} \end{pmatrix}.$$

Note that the integral part of $\bar{w}$ *was discarded. The set* $\bar{B}$*, as Figure 3.1b illustrates, is much smaller, and in particular, not very wide in the vertical direction. Feeding this new data into Algorithm 5, we obtain the same optimal value of* $\frac{4}{5}$ *as before, but now after a single iteration of the main loop.* ◇

As illustrated in Example 6, the performance of Algorithm 5 can be improved if, prior to the execution of the algorithm, we apply a certain affine unimodular transformation τ to its input. In the following, we describe exactly what properties we would like this transformation to have, and how such a transformation could be obtained for arbitrary maximal lattice-free sets.

We recall that, if $B \subseteq \mathbb{R}^2$ is a maximal lattice-free set, then B is either a split, a triangle or a quadrilateral. Following Dey and Louveaux [36], we classify maximal lattice-free triangles as follows:

Definition 7 ([36])**.**

(i) A type-1 triangle is a triangle with integral vertices and exactly one integral point in the relative interior of each facet (see Figure 3.2b);

(ii) A type-2 triangle is a triangle with at least one fractional vertex v, exactly one integral point in the relative interior of two facets incident to v, and at least two integral points on the third facet (see Figure 3.2c);

(iii) A type-3 triangle is a triangle with exactly three integral points on the boundary, one in the relative interior of each edge (see Figure 3.2d).

Another important concept, that we make use in the following, is the width of a set along a certain direction.

Definition 8. *Given $d \in \mathbb{R}^2$, we define the* width of B along d *as*

$$\omega_d(B) = \max_{b \in B} d^T b - \min_{b \in B} d^T b.$$

First, we would like τ to be a transformation such that the width of $\tau(B)$ along the vertical direction $\binom{0}{1}$ is small. Algorithm 13 and Algorithm 14 show how to obtain such τ when B is a maximal lattice-free triangle or quadrilateral, respectively. Both algorithms make use of the following lemma, which is implied by the proofs presented by Hurkens [55].

Lemma 9 ([55])**.**

(i) If $B \subseteq \mathbb{R}^2$ is a maximal lattice-free triangle such that $\binom{0}{0}, \binom{0}{1}, \binom{1}{0}$ are in the relative interiors of distinct faces of B, then there exists $d \in \{\binom{0}{1}, \binom{1}{0}, \binom{1}{1}\}$ such that

$$\omega_d(B) \leq 1 + \frac{2}{3}\sqrt{3}.$$

(ii) If $B \subseteq \mathbb{R}^2$ is a maximal lattice-free quadrilateral such that $\binom{0}{0}, \binom{0}{1}, \binom{1}{0}$ are in the relative interiors of distinct faces of B, then there exists $d \in \{\binom{0}{1}, \binom{1}{0}\}$ such that

$$\omega_d(B) \leq 2.$$

The bound given by Lemma 9 can be slightly improved when B is a maximal lattice-free triangle of type 1 or 2, as the next lemma shows.

Lemma 10. *If $B \subseteq \mathbb{R}^2$ is a maximal lattice-free triangle of type 1 or 2 such that $\binom{0}{0}, \binom{0}{1}, \binom{1}{0}$ are in the relative interiors of distinct faces of B, then there exists $d \in \{\binom{0}{1}, \binom{1}{0}, \binom{1}{1}\}$ such that*

$$\omega_d(B) \leq 2.$$

Proof. If B is a type-1 triangle, then B must be the triangle depicted in Figure 3.2b. Clearly, $d = \binom{1}{0}$ satisfies the condition of the lemma. Now suppose that B is a type-2 triangle. We have three subcases, depending on which facet of B contains multiple lattice points.

For the first subcase, suppose that the facet of B that contains multiple lattice points is the one containing $\binom{1}{0}$ in its relative interior. In this case, the vertices of B are

$$(1, \alpha), (1, -\beta), \left(\frac{-1}{\alpha + \beta - 1}, \frac{\beta}{\alpha + \beta - 1} \right),$$

for some $\alpha, \beta \in \mathbb{R}_+$. If $\omega_{(1,0)}(B) \leq 2$, we are done. Suppose $\omega_{(1,0)}(B) > 2$. Then $\alpha + \beta > 2$, and we have

$$\omega_{(0,1)}(B) = 1 + \frac{1}{\alpha + \beta - 1} \leq 1 + \frac{1}{2 - 1} = 2.$$

In any case, there exists $d \in \left\{ \binom{0}{1}, \binom{1}{0} \right\}$ satisfying the condition of the lemma.

For the second subcase, suppose that the facet of B that contains multiple lattice points is the one containing $\binom{0}{1}$. Let $\bar{B}$ be defined as

$$\bar{B} = \left\{ \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} b : b \in B \right\}$$

Clearly, $\bar{B}$ satisfies the conditions of the first subcase. Therefore, there exists $\bar{d} \in \left\{ \binom{0}{1}, \binom{1}{0} \right\}$ such that $\omega_{\bar{d}}(\bar{B}) \leq 2$, which implies that there exists $d \in \left\{ \binom{1}{0}, \binom{0}{1} \right\}$ such that $\omega_d(B) \leq 2$.

Finally, for the third subcase, suppose that the facet of B that contains multiple lattice points is the one containing $\binom{0}{0}$. We proceed similarly to the previous subcase. Let $\bar{B}$ be defined as

$$\bar{B} = \left\{ \begin{bmatrix} 1 & 0 \\ -1 & -1 \end{bmatrix} b + \begin{bmatrix} 0 \\ 1 \end{bmatrix} : b \in B \right\}$$

Note that $\bar{B}$ is a lattice-free triangle, since the transformation is unimodular. Also, the point $\binom{0}{0}$ is mapped to $\binom{0}{1}$. Therefore, $\bar{B}$ satisfies the conditions of the second subcase, and there exists $\bar{d} \in \left\{ \binom{1}{0}, \binom{0}{1} \right\}$ such that $\omega_{\bar{d}}(\bar{B}) \leq 2$. We conclude that there exists $d \in \left\{ \binom{1}{0}, \binom{1}{1} \right\}$ such that $\omega_d(B) \leq 2$. ■

Propositions 11 and 12 show that when Algorithm 13 or 14 is applied to an arbitrary maximal lattice-free triangle or quadrilateral, it produces a transformed set that satisfies the conditions to apply Lemmas 9 and 10. Specifically, our proposed preprocessing step

generates lattice-free sets $\bar{B}$ that have the points $\{\binom{0}{0}, \binom{1}{0}, \binom{0}{1}\}$ in the relative interiors of distinct faces of B, and a small width $\omega_d(\bar{B})$ for $d = \binom{0}{1}$.

Note that in the preprocessing step, we also transform the given vector w (corresponding to the variable to be lifted) in a specific way, resulting in points (ii) of Propositions 11 and 12, which we will exploit in Section 3.3. In this process, we introduce the variable ε, which is the middle point of $\bar{B}$ along the vertical coordinate x_2.

Proposition 11. *Let $B \subseteq \mathbb{R}^2$ be a maximal lattice-free triangle containing $f \in \mathbb{R}^2$ in its interior, and let $w \in \mathbb{R}^2$. Suppose $v^1, v^2, v^3 \in \mathbb{Z}^2$ are lattice points in the relative interiors of three distinct facets of B. If $\bar{B}, \bar{f}$ and $\bar{w}$ are the values returned by Algorithm 13, and if λ is the width of $\bar{B}$ along the vertical direction, then:*

(i) $\lambda \leq 1 + \frac{2}{3}\sqrt{3}$ if B is a type-3 triangle, and $\lambda \leq 2$ otherwise.

(ii) $|\bar{f}_2 + \bar{w}_2 - \bar{b}_2| \leq \frac{\lambda+1}{2}$ for all $\bar{b} \in \bar{B}$.

Proof. (i) It is clear that τ^1 is an affine unimodular function that maps v^1, v^2, v^3 to the points $\binom{0}{0}, \binom{1}{0}, \binom{0}{1}$, respectively. Furthermore, $\tau^1(B)$ satisfies the conditions for item (i) of Lemma 9, so there exists d such that $\omega_d(\tau^1(B)) \leq 1 + \frac{2}{3}$. Additionally, by Lemma 10, if B is a maximal lattice-free triangle of types 1 or 2, then $\omega_d(\tau^1(B)) \leq 2$. If $d = \binom{0}{1}$, then the direction that minimizes the width of $\tau^1(B)$ is already the vertical direction. In that case, $\tau^2 = \tau^1$, and we are done. If $d = \binom{1}{0}$, then the direction that minimizes the width of $\tau^1(B)$ is the horizontal direction. To obtain τ^2, the algorithm composes τ^1 with a transformation that flips the two coordinates. Finally, if $d = \binom{1}{1}$, then the direction that minimizes the width of $\tau^1(B)$ is perpendicular to the line connecting $\binom{1}{0}$ and $\binom{0}{1}$. To obtain τ^2, the algorithm composes τ^1 with a transformation that maps $\binom{0}{0}, \binom{0}{1}, \binom{1}{0}$ to $\binom{0}{1}, \binom{0}{0}, \binom{1}{0}$, respectively. The direction that minimizes the width of $\tau^2(B)$, therefore, is perpendicular to the line that connects $\binom{0}{0}, \binom{1}{0}$, which is the vertical direction, as desired.

(ii) Let $\varepsilon := \frac{\max\{b_2:(b_1,b_2)\in B\}+\min\{b_2:(b_1,b_2)\in B\}}{2}$ be the middle point of B along the vertical direction. By definition, $|\varepsilon - \bar{b}_2| \leq \frac{\lambda}{2}$, for all $\bar{b} \in \bar{B}$. We also claim that $|\bar{f}_2 + \bar{w}_2 - \varepsilon| \leq \frac{1}{2}$. This is true, since, for every $x \in \mathbb{R}$, if we let $\bar{x} \leftarrow x + \left\lfloor \varepsilon + \frac{1}{2} - x \right\rfloor$ then $|\bar{x} - \varepsilon| \leq \frac{1}{2}$, and it is the exact transformation applied to w'_2 in the algorithms. The result then follows, since, for every $\bar{b} \in \bar{B}$, we have:

$$\begin{aligned} |\bar{f}_2 + \bar{w}_2 - \bar{b}_2| &= |\bar{f}_2 + \bar{w}_2 - \varepsilon + \varepsilon - \bar{b}_2| \\ &\leq |\bar{f}_2 + \bar{w}_2 - \varepsilon| + |\varepsilon - \bar{b}_2| \\ &\leq \frac{1}{2} + \frac{\lambda}{2}. \end{aligned}$$

■

Proposition 12. *Let $B \subseteq \mathbb{R}^2$ be a maximal lattice-free quadrilateral with $f \in \mathbb{R}^2$ in its interior, and let $w \in \mathbb{R}^2$. Suppose $v^1, \dots, v^4 \in \mathbb{Z}^2$ are lattice points in the relative interiors of four distinct facets of B. If $\bar{B}, \bar{f}$ and $\bar{w}$ are the values returned by Algorithm 14, then (i) the width of $\bar{B}$ along the vertical direction is at most 2, and (ii)*

$$|\bar{f}_2 + \bar{w}_2 - \bar{b}_2| \leq \frac{3}{2} \qquad \forall \bar{b} \in \bar{B}.$$

Proof. Similarly to the proof of Proposition 11, it is clear that τ^1 is an affine unimodular function that maps v^1, v^2, v^3 to the points $\binom{0}{0}, \binom{1}{0}, \binom{0}{1}$, respectively. Since $\bar{v}^4 \in \mathbb{Z}^2$, since the area of the quadrilateral defined by $\bar{v}^1, \dots, \bar{v}^4$ is one, and since no $\bar{v}^i$ is a convex combination of the others, then $\bar{v}^4$ can only be either $\binom{1}{1}, \binom{-1}{1}$ or $\binom{1}{-1}$. The transformation τ^2 maps $\bar{v}^1, \dots, \bar{v}^4$ to $\binom{0}{0}, \binom{0}{1}, \binom{1}{0}$ and $\binom{1}{1}$. Furthermore, $\tau^2(B)$ satisfies the conditions for item (ii) of Lemma 9, therefore there exists d such that $\omega_d(\tau^1(B)) \leq 2$. To finish, we proceed similarly to the proof of (ii) in Proposition 11, replacing λ by its upper bound 2. ■

Algorithm 13 Preprocessing step for triangles

function PREPROCESS($B, f, w, v^1, \dots, v^3$)
  Let $\tau^1(x) = \begin{bmatrix} v^2 - v^1 & v^3 - v^1 \end{bmatrix}^{-1} (x - v^1)$
  Let $d \in \{\binom{1}{1}, \binom{1}{0}, \binom{0}{1}\}$ such that $\omega_d(\tau^1(B))$ is minimum.
  Let $\tau^2(x) = \begin{cases} \tau^1(x) & \text{if } d = \binom{0}{1} \\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tau^1(x) & \text{if } d = \binom{1}{0} \\ \begin{bmatrix} 1 & 0 \\ -1 & -1 \end{bmatrix} \tau^1(x) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} & \text{if } d = \binom{1}{1} \end{cases}$
  Let $\bar{B} = \{\tau^2(x) : x \in B\}, \bar{f} \leftarrow \tau^2(f), w' \leftarrow \tau^2(w)$
  Let $\varepsilon \in \mathbb{R}$ such that $|\bar{b}_2 - \varepsilon| \leq \frac{1}{2}\omega_{(0,1)}(\bar{B})$ for all $\bar{b} \in \bar{B}$
  Let $\bar{w}_1 \leftarrow w'_1$ and $\bar{w}_2 \leftarrow w'_2 + \left\lfloor \varepsilon + \frac{1}{2} - \bar{f}_2 - w'_2 \right\rfloor$
  Return $\bar{B}, \bar{f}, \bar{w}$

3.3 Complexity analysis

In this section we study the worst case complexity of Algorithm 5. First, in Subsection 3.3.1, we assume only that the convex lattice-free set $B \subseteq \mathbb{R}^2$ is bounded and full-dimensional (i.e. we do not assume maximality at this point). We obtain an upper bound on the number of iterations of the algorithm, which depends on the second

Algorithm 14 Preprocessing step for quadrilaterals

function PREPROCESS($B, f, w, v^1, \dots, v^4$)

Let $\tau^1(x) = \begin{bmatrix} v^2 - v^1 & v^3 - v^1 \end{bmatrix}^{-1} (x - v^1)$

Let $\bar{v}^i \leftarrow \tau^1(v^i)$ for $i = \{1, \dots, 4\}$

Let $\tau^2(x) = \begin{cases} \tau^1(x) & \text{if } \bar{v}^4 = \left(\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right) \\ \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \tau^1(x) & \text{if } \bar{v}^4 = \left(\begin{smallmatrix} 1 \\ -1 \end{smallmatrix}\right) \\ \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \tau^1(x) & \text{if } \bar{v}^4 = \left(\begin{smallmatrix} -1 \\ 1 \end{smallmatrix}\right) \end{cases}$

Let $d \in \{\left(\begin{smallmatrix} 1 \\ 1 \end{smallmatrix}\right), \left(\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right), \left(\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right)\}$ such that $\omega_d(\tau^1(B))$ is minimum.

Let $\tau^3(x) = \begin{cases} \tau^2(x) & \text{if } d = \left(\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right) \\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tau^2(x) & \text{if } d = \left(\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right) \end{cases}$

Let $\bar{B} = \{\tau^2(x) : x \in B\}, \bar{f} \leftarrow \tau^2(f), w' \leftarrow \tau^2(w)$

Let $\varepsilon \in \mathbb{R}$ such that $|\bar{b}_2 - \varepsilon| \leq \frac{1}{2}\omega_{(0,1)}(\bar{B})$ for all $\bar{b} \in \bar{B}$

Let $\bar{w}_1 \leftarrow w'_1$ and $\bar{w}_2 \leftarrow w'_2 + \left\lfloor \varepsilon + \frac{1}{2} - \bar{f}_2 - w'_2 \right\rfloor$

Return $\bar{B}, \bar{f}, \bar{w}$

covering minimum of B and the width of B along the vertical direction. Next, in Subsection 3.3.2, we focus on the case where B is a maximal lattice-free triangle or quadrilateral. Assuming that B has been preprocessed, we prove that Algorithm 5 requires at most a small number of iterations to finish.

3.3.1 Convex lattice-free sets in general

Let $B \subseteq \mathbb{R}^2$ be a bounded and convex lattice-free set containing the point $f \in \mathbb{R}^2$ in its interior. We do not assume that B is maximal. In this section, for any $\gamma > 0$, we denote by γB the set obtained by scaling B by a factor of γ, using f as the origin. That is,

$$\begin{aligned} \gamma B &= \{\gamma(b - f) + f : b \in B\} \\ &= \left\{ x \in \mathbb{R}^2 : \frac{x}{\gamma} + \frac{f(\gamma - 1)}{\gamma} \in B \right\}. \end{aligned}$$

The two following lemmas prove that, if the union of all integer translations of γB cover $\mathbb{R}^2$, then the value of the trivial lifting $\pi_B(w)$ is at most γ, for any $w \in \mathbb{R}^2$.

Lemma 15. *For any $w \in \mathbb{R}^2$ and $\gamma > 0$, if $w + f \in \gamma B$, then $\psi_B(w) \leq \gamma$.*

Proof.

$$\begin{aligned}\psi_B(w) &= \inf\left\{\lambda : \frac{w}{\lambda} + f \in B, \lambda > 0\right\} \\ &= \inf\left\{\lambda : \frac{w+f}{\lambda} + \frac{f(\lambda-1)}{\lambda} \in B, \lambda > 0\right\} \\ &= \inf\{\lambda : w + f \in \lambda B, \lambda > 0\} \\ &\leq \gamma\end{aligned}$$

■

Lemma 16. *If $\gamma B + \mathbb{Z}^2 = \mathbb{R}^2$ for some $\gamma > 0$, then $\pi_B(w) \leq \gamma$ for all $w \in \mathbb{R}^2$.*

Proof. Let $w \in \mathbb{R}^2$. Since $\gamma B + \mathbb{Z}^2 = \mathbb{R}^2$, there exist $b \in \gamma B$ and $k \in \mathbb{Z}^2$ such that $w + f = b + k$. This implies that $w + f - k$ belongs to γB. By the previous lemma, $\pi_B(w - k) \leq \gamma$. But note that $\pi_B(w - k) = \pi_B(w)$, since applying integer translations to the ray does not change the value of the trivial lifting function. We conclude that $\pi_B(w) \leq \gamma$. ■

In the following, let μ be the smallest non-negative number such that $\mu B + \mathbb{Z}^2 = \mathbb{R}^2$. This number is also known as the *second covering minimum* of B [57]. As we recall, in order to evaluate the function

$$\pi_B(w) = \min_{k \in \mathbb{Z}^2} \psi_B(w + k),$$

for a given $w \in \mathbb{R}^2$, we computed $\min_{k_1 \in \mathbb{Z}} \psi_B \begin{pmatrix} w_1 + k_1 \\ w_2 + \bar{k}_2 \end{pmatrix}$ for different values of $\bar{k}_2 \in \mathbb{Z}$. The next lemma shows that, if μB does not intersect the horizontal line at level $f_2 + w_2 + \bar{k}_2$, for a particular $\bar{k}_2 \in \mathbb{Z}$, then such $\bar{k}_2$ can be safely discarded.

Lemma 17. *Let $\bar{k}_2 \in \mathbb{Z}$ and $\mu > 0$. Suppose there does not exist $b \in \mu B$ such that $b_2 = f_2 + w_2 + \bar{k}_2$. Then*

$$\min_{k_1 \in \mathbb{Z}} \psi_B \begin{pmatrix} w_1 + k_1 \\ w_2 + \bar{k}_2 \end{pmatrix} \geq \min_{\alpha \in \mathbb{R}} \psi_B \begin{pmatrix} \alpha \\ w_2 + \bar{k}_2 \end{pmatrix} > \mu.$$

Proof. Note that

$$\min_{k_1 \in \mathbb{Z}} \psi_B \begin{pmatrix} w_1 + k_1 \\ w_2 + \bar{k}_2 \end{pmatrix} \geq \min_{k_1 \in \mathbb{R}} \psi_B \begin{pmatrix} w_1 + k_1 \\ w_2 + \bar{k}_2 \end{pmatrix} = \min_{\alpha \in \mathbb{R}} \psi_B \begin{pmatrix} \alpha \\ w_2 + \bar{k}_2 \end{pmatrix} = \mu \min_{\alpha \in \mathbb{R}} \psi_{\mu B} \begin{pmatrix} \alpha \\ w_2 + \bar{k}_2 \end{pmatrix}.$$

Since we have $f + \begin{pmatrix} \alpha \\ w_2 + \bar{k}_2 \end{pmatrix} \notin \mu B$ for every $\alpha \in \mathbb{R}$, then $\psi_{\mu B} \begin{pmatrix} \alpha \\ w_2 + \bar{k}_2 \end{pmatrix} > 1$ for every $\alpha \in \mathbb{R}$, and the result follows. ■

A consequence of Lemma 17 is that, if μB is not very wide in the vertical direction, then we only need to consider few values of $\bar{k}_2$ when computing the trivial lifting. This motivates the choice of τ in Section 3.2. The next theorem shows that Algorithm 5 does not spend time considering such useless $\bar{k}_2$.

Theorem 18. *Let $B \subseteq \mathbb{R}^2$ be a lattice-free set with $f \in \mathbb{R}^2$ in its interior, with second covering minimum $\mu \in \mathbb{R}_+$. If $w \in \mathbb{R}^2$ is such that*

$$|f_2 + w_2 - b_2| \leq \sigma \qquad \forall b \in \mu B, \tag{3.7}$$

for some $\sigma \geq 1$, then Algorithm 5 stops after at most $\lfloor \sigma \rfloor$ iterations of the main loop upon receiving B, f and w as input.

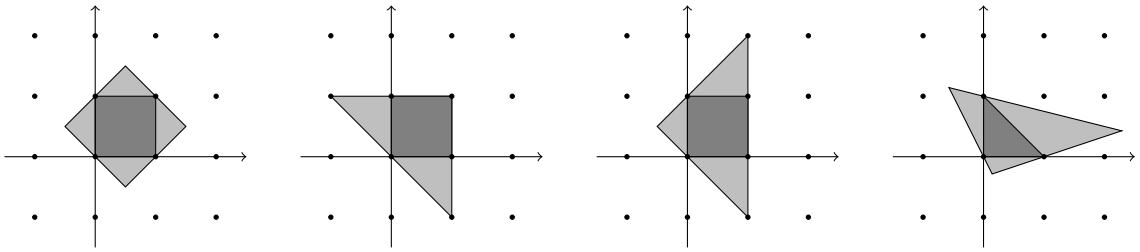
Proof. If there exists $b \in \mu B$ satisfying (3.7) with equality, we can always increase σ very slightly to obtain

$$|f_2 + w_2 - b_2| < \sigma \qquad \forall b \in \mu B,$$

while keeping $\lfloor \sigma \rfloor$ unchanged. We may assume, therefore, that every $b \in \mu B$ satisfies (3.7) strictly. If the algorithm stops before $\lfloor \sigma \rfloor$ iterations, we are done. Suppose, then, that it runs for at least $\lfloor \sigma \rfloor$ iterations. First, we prove that, at the end of iteration $\lfloor \sigma \rfloor$, we have $\eta^* = \pi_B(w) \leq \mu$. Let $\Sigma = \{-\lfloor \sigma \rfloor, \ldots, \lfloor \sigma \rfloor\}$. Clearly, at the end of iteration $\lfloor \sigma \rfloor$, the variable η^* has value $\min_{k_2 \in \Sigma} h(k_2)$. By definition, we also have

$$\pi_B(w) = \min_{k_2 \in \mathbb{Z}} h(k_2) = \min \left\{ \min_{k_2 \in \Sigma} h(k_2), \min_{k_2 \in \mathbb{Z} \setminus \Sigma} h(k_2) \right\} = \min \left\{ \eta^*, \min_{k_2 \in \mathbb{Z} \setminus \Sigma} h(k_2) \right\}.$$

By Lemma 16, $\pi_B(w) \leq \mu$. By Lemma 17, $\min_{k_2 \in \mathbb{Z} \setminus \Sigma} h(k_2) > \mu$. Therefore, $\eta^* = \pi_B(w) \leq \mu$. Now we prove that the algorithm stops at the end of iteration $\lfloor \sigma \rfloor$. First, we prove that $\bar{k}_2 > |w_2|$. At the end of iteration $\lfloor \sigma \rfloor$, the value of $\bar{k}_2$ is $\lfloor \sigma \rfloor + 1$. By assumption, $\sigma > |f_2 + w_2 - b_2|$ for every $b \in \mu B$. Since $f \in \mu B$, we have $\sigma > |w_2|$. Therefore, $\bar{k}_2 = \lfloor \sigma \rfloor + 1 \geq \sigma > |w_2|$. Now we prove that $w_2 + \bar{k}_2 > \frac{\eta^*}{\zeta^+}$. Since $\bar{k}_2$ has value $\lfloor \sigma \rfloor + 1$, there does not exist $b \in \mu B$ such that $b_2 - f_2 - w_2 = \bar{k}_2$. By Lemma 17,



(a) Quadrilateral (b) Type-1 triangle (c) Type-2 triangle (d) Type-3 triangle

Figure 3.2: Proof of Lemma 19.

and since $\bar{k}_2 > |w_2|$, as proved earlier, we have

$$\begin{aligned}
&\min_{\alpha\in\mathbb{R}} \psi_B \left(\begin{smallmatrix} \alpha \\ w_2+\bar{k}_2 \end{smallmatrix} \right) > \mu \geq \eta^* \\
\Rightarrow &(w_2 + \bar{k}_2) \min_{\alpha\in\mathbb{R}} \psi_B \left(\begin{smallmatrix} \alpha \\ 1 \end{smallmatrix} \right) > \eta^* \\
\Rightarrow &(w_2 + \bar{k}_2)\zeta^+ > \eta^* \\
\Rightarrow &w_2 + \bar{k}_2 > \frac{\eta^*}{\zeta^*}.
\end{aligned}$$

We can similarly prove that $\bar{k}_2 - w_2 > \frac{\eta^*}{\zeta^-}$. Therefore, at the end of iteration $\lfloor\sigma\rfloor$, the loop condition is satisfied, and the algorithm stops. ■

3.3.2 Maximal lattice-free sets

Now we suppose that $B \subseteq \mathbb{R}^2$ is a bounded maximal lattice-free set containing f in its interior. In this case, B is either a maximal lattice-free triangle or a maximal lattice-free quadrilateral. Maximal lattice-free sets are interesting in practice, since these are the sets that generate the strongest valid inequalities for the infinite continuous multi-row relaxation. Aside from triangles and quadrilaterals, maximal lattice-free sets include splits, but the latter can be lifted easily by means of a closed formula. We will show that Algorithm 5 requires at most one or four iterations of the main loop to finish, depending on whether B is a type-3 triangle or not.

It is well known that the second covering minimum of any lattice-free set must be at least one. The following lemma shows the second covering minimum of B is also bounded above by a constant. Although this result is probably known as well, to the best of our knowledge, it was not proved explicitly in the literature, and it is included here for completeness.

Lemma 19. *Let $B \subseteq \mathbb{R}^2$ is a full-dimensional maximal lattice-free set. If B is a type-1*

triangle, a type-2 triangle or a quadrilateral, then $\mu \leq 1$. If B is a type-3 triangle, then $\mu \leq 2$.

Proof. For any set $B \subseteq \mathbb{R}^n$, we know that $B + \mathbb{Z}^n = \mathbb{R}^n$ if and only if $\tau(B) + \mathbb{Z}^n = \mathbb{R}^n$, where τ is any unimodular affine transformation. Suppose B is a maximal lattice-free quadrilateral. By applying a unimodular affine transformation, we may assume that the points $\binom{0}{0}, \binom{0}{1}, \binom{1}{0}$ and $\binom{1}{1}$ are in the relative interiors of four distinct facets of B. Therefore, B contains the unit square, and clearly $B + \mathbb{Z}^2 = \mathbb{R}^2$. The same argument applies for type-1 and type-2 triangles. See Figure 3.2 for an illustration. Now suppose B is a type-3 triangle. We may assume that the points $\binom{0}{0}, \binom{0}{1}$ and $\binom{1}{0}$ are in the relative interiors of three distinct facets of B. The set $2B$, therefore, contains a translated type-1 triangle as a subset. By the previous case, $2B + \mathbb{Z}^2$ covers $\mathbb{R}^2$. ■

Now we proceed to obtain upper bounds on the number of iterations of Algorithm 5 for maximal lattice-free sets, by applying Theorem 18. We consider two distinct cases, depending on whether B is a type-3 triangle or not. In every case, we assume that B, f and w have been preprocessed either by Algorithm 13 or by Algorithm 14. First, suppose that B is either a type-1 triangle, or a type-2 triangle, or a quadrilateral. The next theorem shows that the algorithm stops after a single iteration of the main loop.

Theorem 20. *Let $B \subseteq \mathbb{R}^2$ be a maximal lattice-free quadrilateral or triangle of types 1 or 2 containing $f \in \mathbb{R}^2$ in its interior and having vertical width at most 2. If $w \in \mathbb{R}^2$ is such that*

$$|f_2 + w_2 - b_2| \leq \frac{3}{2} \qquad \forall b \in B,$$

then, upon receiving B, f and w as input, Algorithm 5 stops after at most a single iteration of the main loop.

Proof. Let μ be the second covering minimum of B. By Lemma 19, $\mu \leq 1$. Since every lattice-free set has second covering minimum at least one, then $\mu = 1$. Therefore,

$$|f_2 + w_2 - b_2| \leq \frac{3}{2} \qquad \forall b \in \mu B.$$

By Theorem 18, we conclude that Algorithm 5 requires at most $\left\lfloor \frac{3}{2} \right\rfloor = 1$ iteration to finish. ■

This case is closely related to the closed formula presented by Basu, Bonami, Cornuéjols and Margot [19], which can compute trivial lifting coefficients under the assumption

that B is a type-1 or type-2 maximal lattice-free triangle. Although the formula itself is closed, it assumes that B is already in some standard form, and thus requires the application of a pre-processing step. The formula works by evaluating the function ψ_B at six points, in the worst case, and taking the minimum.

Theorem 20 proves that Algorithm 5 requires constant time to finish in the worst case. More interestingly, however, this theorem shows that the algorithm requires at most a single iteration of the main loop, implying that at most six calls to evaluate the function ψ_B are needed. This matches the number of calls made by the closed formula presented by Basu et al. Also note that Algorithm 5 requires the same number of calls when B is a maximal lattice-free quadrilateral, while the aforementioned closed formula does not apply for quadrilaterals.

Now we consider the case where B is a type-3 triangle. In this case, since both the maximum width along the vertical direction, as well as the second covering minimum, can be higher than before, the algorithm may require more iterations to terminate. In the worst case, however, it still requires at most a low, constant number of iterations.

Lemma 21. *Let $B \subseteq \mathbb{R}^2$ be a lattice-free set with $f \in \mathbb{R}^2$ in its interior. Also, let $w \in \mathbb{R}^2$ and $\gamma > 0$ such that*

$$|f_2 + w_2 - b_2| \leq \gamma \qquad \forall b \in B.$$

Then, for any $\mu \geq 1$,

$$|f_2 + w_2 - \bar{b}_2| \leq \gamma(2\mu - 1) \qquad \forall \bar{b} \in \mu B.$$

Proof. Let $\bar{b} \in \mu B$. By definition, there exists $b \in B$ such that $\bar{b} = \mu(b - f) + f$. Also, since $f \in B$ and $|f_2 + w_2 - b_2| \leq \gamma$ for every $b \in B$, we have $|w_2| \leq \gamma$. Therefore,

$$\begin{aligned}
|f_2 + w_2 - \bar{b}_2| &= |f_2 + w_2 - \mu(b_2 - f_2) - f_2| \\
&= |\mu(f_2 + w_2 - b_2) - (\mu - 1)w_2| \\
&\leq \mu|f_2 + w_2 - b_2| + (\mu - 1)|w_2| \\
&\leq \mu\gamma + (\mu - 1)\gamma \\
&= \gamma(2\mu - 1)
\end{aligned}$$

■

Theorem 22. *Let $B \subseteq \mathbb{R}^2$ be a type-3 triangle containing $f \in \mathbb{R}^2$ in its interior and*

having vertical width at most $1 + \frac{2}{3}\sqrt{3}$. *If* $w \in \mathbb{R}^2$ *is such that*

$$|f_2 + w_2 - b_2| \leq 1 + \frac{1}{3}\sqrt{3} \qquad \forall b \in B,$$

then, upon receiving B, f *and* w *as input, Algorithm 5 stops after at most four iterations of the main loop.*

Proof. Let μ be the second covering minimum of B. By Lemma 21,

$$|f_2 + w_2 - b_2| \leq \left(1 + \frac{1}{3}\sqrt{3}\right)(2\mu - 1) \qquad \forall b \in \mu B.$$

Since $\mu \leq 2$ by Lemma 19, we have

$$|f_2 + w_2 - b_2| \leq 3 + \sqrt{3} \qquad \forall b \in \mu B.$$

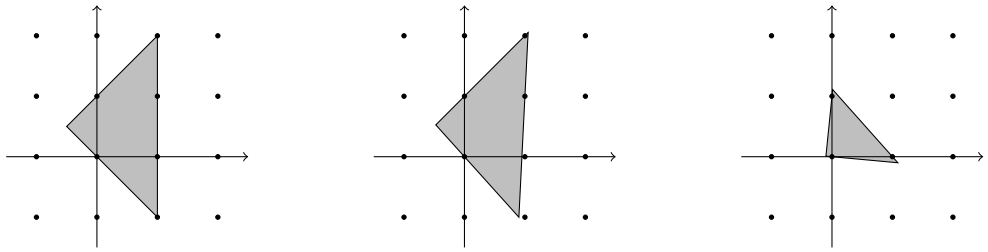
Therefore, by Theorem 18, we conclude that Algorithm 5 requires at most $\lfloor 3 + \sqrt{3} \rfloor = 4$ iterations to finish. ■

It is worth noting that the upper bound on the number of iterations obtained in Theorem 22 is a worst-case upper bound and that the actual run time for some type 3 triangles can be smaller. For instance, the example shown in Figure 3.3b is a type 3 triangle that is very close to being a type 2 triangle (Figure 3.3a), so its trivial lifting coefficient and the runtime of the algorithm are likely similar to the type 2 case. On the other hand, the worst case runtime will be obtained by a type 3 triangle with high second covering minimum, like the one seen in Figure 3.3c.

Another issue worth mentioning is that, computationally, it may be difficult to differentiate between the cases in Figures 3.3a and 3.3b, due to numerical inaccuracy. This highlights a big advantage of the generic trivial lifting algorithm presented in this work, which computes the correct coefficient for any (even non-maximal) lattice-free set, as opposed to one that relies on the particular format of the maximal lattice-free set.

3.4 Computational experiments

In order to evaluate the practical efficiency of Algorithm 5, we implemented it and compared it against two variations of the naive method described in the introduction and against a black-box MIP solver given the formulation of Averkov and Basu [9].



(a) Type-2 triangle (b) "Good" type-3 triangle (c) "Bad" type-3 triangle

Figure 3.3: Illustration of "good" and "bad" type 3 triangles.

3.4.1 Algorithms and variations

Two variations of Algorithm 5 were implemented and tested. The first variation (`bound-orig`) applies the procedure directly to the input data, without performing any kind of preprocessing, while the second variation (`bound-pre`) applies the preprocessing step described in Section 3.2. Both variations were implemented in standard C, and do not make use of any external dependencies. The complete source code has been made available online [67]. To evaluate the function

$$\min_{\alpha_1 \in \mathbb{R}} \psi \begin{pmatrix} \alpha_1 \\ \bar{\alpha}_2 \end{pmatrix}, \tag{3.8}$$

where $\bar{\alpha}_2$ is fixed, an ad-hoc method was used, instead of a generic LP solver, in order to minimize the computational overhead. More precisely, if $B \subseteq \mathbb{R}^2$ is a polyhedron containing $f \in \mathbb{R}^2$ in its interior, then B can be written as

$$B = \{x \in \mathbb{R}^2 : a^i(x - f) \leq 1, i \in \{1, \ldots, t\}\}$$

where $a^1, \ldots, a^t \in \mathbb{R}^2$. Then (3.8) is equivalent to the linear program

$$\begin{aligned} \text{minimize} \quad & \varepsilon \\ \text{subject to} \quad & \varepsilon - a_1^i \alpha_1 \geq a_2^i \bar{\alpha}_2 \qquad\qquad i = 1, \ldots, t \\ & \varepsilon, \alpha_1 \text{ free} \end{aligned}$$

Since, in our experiments, B is always a maximal lattice-free set, this LP has at most a constant and very small number of bases. Instead of calling a generic LP solver, we simply enumerated all these bases, and found the one with best objective value.

We also implemented and tested two variations of the naive trivial lifting algorithm. The first variation (`naive-fixed`) simply evaluates the function $\psi(w + k)$ for all $k \in$

$[-M, M]^2$, where M is a large constant which does not depend on any input data. During our tests, this value was fixed to 50. This variation is the simplest to implement, and probably the most widespread, but does not always produce the correct answers, since, for every fixed M, there is always a lattice-free set $B \subseteq \mathbb{R}^2$ such that M is not large enough for B. The second variation (`naive-bbox`) solves this problem by computing the bounding box for each lattice-free set B, and evaluating all $\psi(w+k)$ for all k such that $f + w + k$ is either inside or reasonably close to the bounding box. In order to avoid exceedingly long running times on some hard instances, the bounding box was intersected with a box $[-M, M]^2$ of fixed size, where M is a large constant which was fixed to 10^4 in our tests. Setting this constant to a smaller value reduces the maximum running time of `naive-bbox`, but increases the probability of generating incorrect answers.

All computations, for all the variations previously described, were performed in floating point arithmetic, due to the observation that small arithmetical errors are not amplified by the algorithms, and therefore have no significant impact in the final cut coefficient. The code used to evaluate the function $\psi(w)$ for a certain $w \in \mathbb{R}^2$ was also exactly the same.

Finally, we also modeled (3.3) as a mixed-integer linear program with a fixed number of variables, as described in [9], and we solved it using IBM ILOG CPLEX, version 12.4. We refer to this implementation as `mip`. Because we use a generic MIP solver (based on branch-and-bound and the simplex method), even in fixed dimension, the worst-case running time is exponential in the encoding size of the problem, while it could be polynomial in theory. This choice was made because, to the best of our knowledge, there is no currently-available MIP solver that is polynomial in fixed-dimension and competitive with CPLEX, for MIPs containing both continuous and integral variables.

3.4.2 Instances

For our computational experiments, an instance of the trivial lifting problem consists of a convex lattice-free set $B \subseteq \mathbb{R}^2$, along with an interior point $f \in \mathbb{R}^2$, and a list of rays $w^1, \ldots, w^k$ that should be lifted. We use a list of rays, instead of a single ray, since, in practice, a cut generator usually needs to lift, for the same intersection cut, multiple rays corresponding to different integer variables. By receiving a list of rays in advance, the cut generator may perform a preprocessing step exactly once, before the trivial lifting computations begin, instead of one time for each ray. The decision of whether to apply some costly preprocessing step could also depend on the number of rays to be lifted, although we limited ourselves to a fixed choice here.

In our experiments, we used two lists of lattice-free sets. The first list was obtained by running the two-row intersection cut generator implemented by Louveaux and Poirrier [60] on the benchmark set of the MIPLIB 2010 [58] and capturing, for each intersection cut generated, the associated lattice-free set. This list was then filtered to exclude splits, since these can be lifted easily, and non-maximal lattice-free sets, since these can be transformed into maximal lattice-free sets. Finally, a sample of 1000 lattice-free sets from this list, picked randomly, was considered for our experiments. The second list of lattice-free sets was obtained by applying a shear transformation to each set on the first list. The precise transformation was given by

$$f(x) = \begin{bmatrix} 51 & 5 \\ 10 & 1 \end{bmatrix} x.$$

This transformed list has sets that resemble the set from Figure 3.1a, and can be seen as a pathological scenario for trivial lifting algorithms.

For each lattice-free set on each list, we also randomly generated a fixed number of rays w^i, uniformly distributed inside the box $[0, 1)^2$. The lists of rays were generated randomly, since we observed that the choice of rays to be lifted had negligible impact in the performance of the algorithms considered. Most of the impact came, instead, from the choice of lattice-free sets. This also allowed us to evaluate the impact of lifting a different number of rays for each lattice-free set.

3.4.3 Results and discussion

First, we focus on the unmodified list of lattice-free sets obtained from the cut-generator. Table 3.1 summarizes the CPU running times that each of the four algorithms took to process this list of lattice-free sets, with 100 randomly generated rays per set. The running time to process one lattice-free set includes any time spent on preprocessing the set, plus the time spent computing the lifted coefficients for all the rays. For each algorithm, the table shows the average, the median and the maximum running times in milliseconds to process each set. The table also shows the percentage of sets for which at least one cut coefficient was calculated incorrectly.

As Table 3.1 shows, algorithm `bound-pre` presented the fastest average running time among all variations for this set of instances, taking only 0.059 ms on average to process each set. This was more than 9, 150, 390 and 2400 times faster on average than `bound-orig`, `naive-bbox`, `naive-fixed` and `mip`, respectively. It performed consistently well on all instances, having a maximum running time of only 0.100 ms.

	bound-pre	bound-orig	naive-bbox	naive-fixed	mip
Average (ms)	0.059	0.565	9.341	23.278	141.938
Median (ms)	0.060	0.068	2.120	23.080	129.200
Maximum (ms)	0.100	33.264	1769.880	30.080	947.600
Failure Rate	0.0 %	0.0 %	0.0 %	0.3 %	0.0 %
Best	83.6 %	37.2 %	0.0 %	0.0 %	0.0 %
Avg Ratio to Best	1.015	9.483	159.449	402.950	2449.938

Table 3.1: Running times statistics: original lattice-free sets, 100 rays per set.

	bound-pre	bound-orig	naive-bbox	naive-fixed	mip
Average (ms)	0.062	4.709	5473.688	23.374	1906.081
Median (ms)	0.064	0.440	21.200	23.160	582.800
Maximum (ms)	0.108	485.760	392259.600	32.120	94620.000
Failure Rate	0.0 %	0.0 %	0.1 %	10.0 %	1.7 %
Best	100.0 %	0.0 %	0.0 %	0.0 %	0.0 %
Avg Ratio to Best	1.000	74.602	87472.010	380.532	31250.644

Table 3.2: Running times statistics: transformed lattice-free sets, 100 rays per set.

Algorithms `bound-orig`, `naive-bbox` and `mip`, on the other hand, presented significant slowdown for some instances, having maximum running times of 33, 1769 ms and 947 ms respectively, which is more than 480, 830 and 7 times their medians. Although algorithm `naive-fixed` was consistent and had a better worst-case running time than `naive-bbox`, we note that it failed to compute some coefficients correctly.

Table 3.1 also shows that, for each instance, the best running time was obtained either by `bound-pre` or `bound-orig`, and never by the other algorithms. Algorithms `bound-pre` and `bound-orig` presented average ratio-to-best of 1.015 and 9.483 respectively. Although `bound-orig` was faster than `bound-pre` for some instances, it was only very slightly so. For instances where `bound-pre` was faster than `bound-orig`, however, the difference in running times was much more significant.

Now we consider the second list of lattice-free sets, obtained by applying a shear transformation to the sets of the first list, with 100 randomly generated rays per set. Table 3.2 summarizes the running times for all algorithms. The transformation had virtually no impact on the performance of algorithm `bound-pre`. Its average, median and maximum running times remained almost unchanged. The median running times for algorithms `bound-orig`, `naive-bbox` and `mip`, on the other hand, became on average

6, 10 and 4 times slower than before, respectively. For some instances, `naive-bbox` and `mip` required minutes of processing time, while `bound-pre` required only a fraction of millisecond. The table also shows that failure rate of algorithm `naive-fixed` increased considerably, to 10%. Algorithm `mip` calculated some coefficients incorrectly, due to insufficient numerical precision. Because of our restriction on the maximum size of the bounding boxes, algorithm `naive-bbox` also produced some incorrect coefficients. Algorithm `bound-pre` presented the best performance for every instance in this set.

Chapter 4

Intersection Cuts from the Infinity Norm

As discussed in Section 2.2, valid inequalities for the continuous multi-row relaxation can be easily generated from maximal lattice-free sets, and these inequalities can be used as cutting planes for solving general MIPs. One important computational question is to decide which inequalities to use. In this chapter, we propose a new subset of intersection cuts, derived based on the infinity norm, and we run computational experiments to measure their effectiveness.

Consider the continuous multi-row relaxation of the simplex tableau, given by

$$\begin{aligned} x &= f + \sum_{j=1}^{m} r^j s_j \\ x &\in \mathbb{Z}^n \\ s &\in \mathbb{R}^m_+, \end{aligned} \tag{C}$$

where $f \in \mathbb{Q}^n \setminus \{0\}$ and $r^1, \ldots, r^j \in \mathbb{Q}^n \setminus \{0\}$. As discussed in Section 2.2, valid inequalities for (C) can be constructed from lattice-free sets. More specifically, if $B \subseteq \mathbb{R}^n$ is a lattice-free set containing f in its interior, and if ψ is the gauge function of $B - f$, then the inequality

$$\sum_{j=1}^{m} \psi(r^j) s_j \geq 1, \tag{4.1}$$

is valid for (C). In the following, we will refer to (4.1) as *the intersection cut obtained from* B, and, for conciseness, when (4.1) is facet-defining for the convex hull of (C), we will say that B itself is facet-defining.

We recall that our interest in valid inequalities for (C) comes from the fact that they can be used as cutting planes to solve general mixed-integer optimization problems

(MIPs). From this perspective, we are interested in inequalities from (C) that cut off the current fractional basic solution $(x, s) = (f, 0)$. It can be easily verified, however, that any intersection cut generated as described above satisfies this requirement. An important practical question, then, is to decide which intersection cuts from (C) to use in a cutting plane algorithm. Various factors must be considered, including the speed in which these cuts can be generated and the total number of cuts used. In the following, we present some cut selection strategies that have been proposed in the literature, as well as their impact on solving MIPs.

In the first extensive computational experiments on the impact of using multi-row intersection cuts, Espinoza [42] considers intersection cuts generated from three simple families of bounded, maximal lattice-free polyhedra. The lattice-free sets considered have fixed shape, and, in particular, do not change according to the values r^j. The impact of each family of cuts, for a different number of tableau rows, was measured individually on instances from MIPLIB 3.0 and MIPLIB 2003. The cuts improved both the LP bound obtained at the root node of the branch-and-bound tree, as well as the total running time of the algorithm. The best configurations, compared against CPLEX 11.0 defaults, increased the gap closed at the root node by 2.5 percentage points and reduced the overall running time by 5%. Interestingly, the family of intersection cuts that presented the best results was never facet-defining for (C). The paper concludes that even simple subclasses of intersection cuts from (C) can have positive impact on the performance of branch-and-cut algorithms, and points to identifying additional important subclasses of such cuts, along with good computational implementation choices, as an interesting research question.

Musalem [64] focuses on generating multi-row intersection cuts $\pi^T s \geq 1$ that minimize the 1-norm and the 2-norm of the vector π. The problem of finding these cuts is modeled as a MIP and solved using column generation. Computational experiments are conducted on instances from MIPLIB 3.0, and the results are compared against the split cut separator implemented by Balas and Saxena [16]. Results are negative, as the 1-norm cut separator was only able to close 34% of the integrality gap at the root node, on average, under a two-hour time limit, while the split cut separator is able to close 86% of the gap, under a 10-hour time limit. Yamangil [70] also performs computational experiments on cuts minimizing the 1-norm, for relaxations with 2 to 5 rows. Compared against CPLEX defaults, these cuts closed, on average, 60% of the integrality gap at the root node, while CPLEX closed 52%.

Dey, Lodi, Tramontani and Wolsey [34, 35] focus on the specific case where (C) is a two-row relaxation. They evaluate the impact of cuts generated from maximal

lattice-free triangles and from two-row splits, compared against, and in conjunction with, Gomory Mixed-Integer (GMI) cuts. Because the number of suitable triangles, even when restricted to facet-defining ones, is very large, they develop a heuristic procedure to select a small, but effective subset of triangles. Experiments are conducted both on a set of randomly generated instances and on a subset of MIPLIB 3.0 instances. While both families of two-row intersection cuts, used together, proved useful for all sets of instances, most of the improvement for real-world instances came from two-row split cuts.

Basu, Bonami, Cornuéjols and Margot [19] also focus on the two-row case. They generate cuts from relaxations where one component of f is integral and the other is fractional. This typically occurs when (C) comes from a tableau whose basis is degenerate. They measure the impact of a restricted subset of lattice-free triangles, carefully chosen so that the separation and trivial lifting procedure can be done through simple closed formulas. To measure the strength of these cuts, they use the method *diving towards a feasible solution*, described by Margot [63]. The conclusion is that the family of two-row cuts tested is not competitive with GMI cuts.

Louveaux and Poirrier [60] also focus on the two-row case, but instead of considering lattice-free sets with specific shapes, they develop an exact separator that, given a point $(\bar{x}, \bar{s})$, either finds an intersection cut from (C) that cuts off this point, or proves that no such intersection cut exists. In this way, they can evaluate the impact of using all facet-defining intersection cuts for (C) simultaneously, without explicitly enumerating them. By using techniques to reduce the complexity of the polar set of (C), they are able to obtain a method that works well in practice, despite having no guarantee of polynomial running time. They conclude that the gap closed by two-row cuts, on top of single-row cuts, is considerable, although much of that closure can be achieved from split cuts. In later experiments, Louveaux, Poirrier and Salvagnin [61] develop another exact separator that, although much slower than the former, can separate over several variations of (C), and can handle relaxations with arbitrary numbers of rows.

Following the direction proposed by Espinoza [42], our goal in this chapter is to identify additional subclasses of intersection cuts generated from (C) that can be easily computed in practice and that, when used as cutting planes for solving MIPs, provide benefits comparable to using all the facet-defining inequalities for (C). While many strategies have been proposed for two-row relaxations and triangles, other subclasses of intersection cuts, specially for relaxations with more than two rows, have received much less attention.

In this chapter, we introduce a new subset of intersection cuts from (C), based on

the infinity norm, which has many interesting properties. First, it is very small, with exactly one cut per continuous multi-row relaxation. Its cuts are minimal, which implies that they are not very easily dominated. Unlike the classes introduced by [34, 35] and [19], it works for relaxations with arbitrary numbers of rows, and unlike the subclasses introduced by [42], it takes into account the values of the variables r^j. Finally, these cuts have an interesting geometrical interpretation, which we exploit in order to compute them more efficiently.

We start in Section 4.1, by giving a precise definition of these cuts and proving some of their basic properties. In Section 4.2 we describe an algorithm for generating them. Finally, in Section 4.3, we run extensive computational experiments to evaluate their strength. We conclude that these cuts yield, in terms of gap closure, about 50% of the benefits of using the exact separator, but at a small fraction of the computational cost, and with a significantly smaller number of cuts.

4.1 Definition of infinity cuts

In this chapter, we consider the finite continuous multi-row relaxation (C). We assume that this relaxation has at least one feasible solution, in which case it can be proved that every valid inequality that cuts off the point $(f, 0)$ can be written as

$$\sum_{i=1}^{m} \pi_i s_i \geq 1,$$

where $\pi_1, \ldots, \pi_m \geq 0$. Since the variables s_i are non-negative, we are generally interested in cuts that have small π_i coefficients. A natural idea, therefore, is to consider cuts that minimize $\|\pi\|$ for different norms. Musalem [64] and Yamangil [70] performed computational experiments with cuts minimizing the 1-norm and the 2-norm. In this chapter, we study intersection cuts that minimize the infinity norm.

Given a vector $\pi \in \mathbb{R}^m$, we recall that the infinity norm of π is given by

$$\|\pi\|_\infty = \max\{|\pi_1|, \ldots, |\pi_m|\}.$$

Since this norm only takes into account the coefficient with largest magnitude, cuts that minimize it, in a sense, try to assign, with equal priority, good coefficients to all the variables. Next, we describe a simple geometric way of obtaining a cut that minimizes

the infinity norm. Consider the inequality

$$\sum_{i=1}^{m} \varepsilon s_i \geq 1, \tag{4.2}$$

where $\varepsilon > 0$. Using the framework of intersection cuts, as introduced in Section 2.2, we can prove that inequality (4.2) is valid for (C) by verifying that a certain convex set is lattice-free, as the next lemma shows.

Lemma 23. *For every $\varepsilon > 0$, if the convex set*

$$B(\varepsilon) = \operatorname{conv}\left(\{f\} \cup \left\{f + \frac{1}{\varepsilon} r^i : i \in \{1, \ldots, m\}\right\}\right).$$

is a lattice-free set, then (4.2) is valid for (C).

Proof. Let ψ be the gauge function of $B(\varepsilon) - f$. We recall that, by definition,

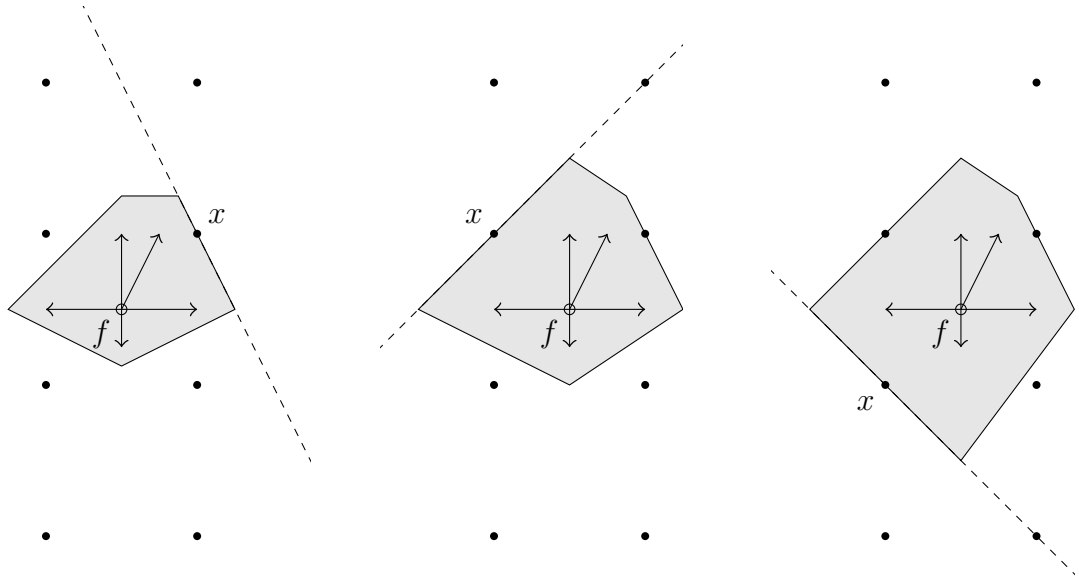
$$\psi(r) = \inf\left\{\varepsilon > 0 : \frac{r}{\varepsilon} + f \in B(\varepsilon)\right\} \qquad \forall r \in \mathbb{Q}^n.$$

For every $i \in \{1, \ldots, m\}$, since $f + \frac{1}{\varepsilon} r^i \in B(\varepsilon)$, then $\psi(r^i) \leq \varepsilon$. Using the results from Section 2.2, we know that $\sum_{i=1}^{m} \psi(r^i) s_i \geq 1$ is a valid inequality for (C). Since this inequality is at least as strong as (4.2), we conclude that (4.2) is also a valid inequality for (C). ■

From the previous lemma and from the fact that $f \notin \mathbb{Z}^n$, it is clear that, for a large enough ε, the set $B(\varepsilon)$ is lattice-free, and therefore (4.2) is valid. One strategy for obtaining a cut that minimizes the infinity norm, therefore, is to start with a very large value of ε and slowly decrease it, while making sure that the set $B(\varepsilon)$ is lattice-free. We stop when an integer point touches the boundary of $B(\varepsilon)$. Although a cut that minimizes the infinity norm can be easily obtained in this way, we note that such cut is not necessarily minimal, and, in most cases, can be further strengthened. This is illustrated in the next example.

Example 24. *Consider the continuous relaxation*

$$\begin{aligned} x &= \begin{pmatrix} \frac{1}{2} \\ \frac{1}{2} \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{1}{2} \end{pmatrix} s_1 + \begin{pmatrix} \frac{1}{4} \\ \frac{1}{2} \end{pmatrix} s_2 + \begin{pmatrix} \frac{1}{2} \\ 0 \end{pmatrix} s_3 + \begin{pmatrix} 0 \\ -\frac{1}{4} \end{pmatrix} s_4 + \begin{pmatrix} -\frac{1}{2} \\ 0 \end{pmatrix} s_5 \\ x &\in \mathbb{Z}^2 \\ s &\in \mathbb{R}^5_+ \end{aligned} \tag{4.3}$$



(a) Level 1 (b) Level 2 (c) Level 3

Figure 4.1: Example of infinity cut for a two-row relaxation with 5 rays.

The set $B\left(\frac{2}{3}\right)$ *is illustrated in Figure 4.1a. Since this set is lattice-free and the point* $x = (1, 1)$ *belongs to its boundary, we conclude that*

$$\frac{2}{3}s_1 + \frac{2}{3}s_2 + \frac{2}{3}s_3 + \frac{2}{3}s_4 + \frac{2}{3}s_5 \geq 1$$

is an intersection cut that minimizes the infinity norm. Note, however, that this cut is not minimal. Indeed, it is dominated by the cut

$$\frac{1}{2}s_1 + \frac{2}{3}s_2 + \frac{2}{3}s_3 + \frac{1}{4}s_4 + \frac{1}{2}s_5 \geq 1,$$

which can be obtained from the lattice-free set illustrated in Figure 4.1c. ◇

As the previous example illustrates, although the cuts obtained using the previous strategy minimize the infinity norm, they are not necessarily minimal. In the following, we present a more restricted subset of cuts that are both minimal and that minimize the infinity norm. We start with the definition of a *tight* ray. Intuitively, a ray r is tight in a cut if its cut coefficient cannot be decreased without generating an invalid inequality.

Definition 25. *Given an inequality* $\sum_{i=1}^{m} \pi_i s_i \geq 1$ *that is valid for* (C) *and a ray* r^k*, where* $k \in \{1, \ldots, m\}$*, we say that* r^k *is* tight *with respect to this inequality if, for any*

$\bar{\pi}_k < \pi_k$, the inequality

$$\sum_{\substack{i=1 \\ i \neq k}}^{m} \pi_i s_i + \bar{\pi}_k s_k \geq 1$$

is not valid for (C).

Next, we present the precise definition of the subset of cuts we will study in this chapter. The definition is recursive, and leads naturally to an algorithm for obtaining these cuts. The idea is to decrease the cut coefficients of all the variables simultaneously, until some rays become tight and their coefficients cannot be decreased any further. The coefficients for these tight rays are then fixed, and the remaining coefficients are then lowered simultaneously. The process repeats, until all the rays become tight.

Definition 26.

(i) The inequality

$$\sum_{i=1}^{m} \varepsilon s_i \geq 1 \tag{4.4}$$

is an infinity cut (level 1) if $\varepsilon \geq 0$ is the smallest number such that (4.4) *is valid.*

(ii) Suppose

$$\sum_{i=1}^{m} \pi_i s_i \geq 1 \tag{4.5}$$

is an infinity cut (level k), for some $k \in \{1, 2, \ldots\}$ and some $\pi \in \mathbb{R}^m_+$. Let $T \subseteq \{1, \ldots, m\}$ be the set of indices i such that r^i is tight with respect to (4.5)*, and let $N = \{1, \ldots, m\} \setminus T$ be the set of the remaining indices. The inequality*

$$\sum_{i \in T} \pi_i s_i + \sum_{i \in N} \varepsilon s_i \geq 1 \tag{4.6}$$

is an infinity cut (level $k+1$) if $\varepsilon \geq 0$ is the smallest number such that (4.6) *is valid.*

Figure 4.1 shows the infinity cuts (levels 1, 2 and 3) for the relaxation in Example 24. The next proposition shows the infinity cut (level m) is indeed minimal, in the sense that no coefficient can be made individually stronger without generating an invalid cut.

Proposition 27. *If $\sum_{i=1}^{m} \pi_i s_i \geq 1$ is an infinity cut (level m), then this inequality is minimal.*

Proof. It can be easily proved by induction that at least k rays are tight with respect to the infinity cut (level k), for any $k \in \{1, \dots, m\}$. It follows that all the rays are tight with respect to the infinity cut (level m), and therefore the cut is minimal. ■

We end this section with the observation that, from Definition 26, it is also clear that the infinity cut (level m) is unique for each continuous corner relaxation. This subset of intersection cuts, therefore, is very small, which makes them attractive computationally.

4.2 Computation of infinity cuts

In this section, we present an algorithm for computing the infinity cut (level m) for a given finite continuous multi-row relaxation. As in the previous section, we take a geometric approach, and focus on scaling up a certain lattice-free set.

First observe that, if (4.6) is an infinity cut (level k), then it can also be written as

$$\sum_{i=1}^{m} \max\{\beta_i, \varepsilon\} s_i \geq 1, \tag{4.7}$$

for some $\beta_1, \dots, \beta_m \geq 0$. Indeed, one valid choice of β is $\beta_i = \pi_i$ for every $i \in T$, and $\beta_i = 0$ for every $i \in N$. This form is more convenient than (4.6) since we do not need to deal with the sets T and N; all we need to characterize the cut are the values β_i. Similarly to the previous section we can prove that (4.7) is a valid inequality by verifying that a certain convex set is lattice-free, as shown by the next lemma.

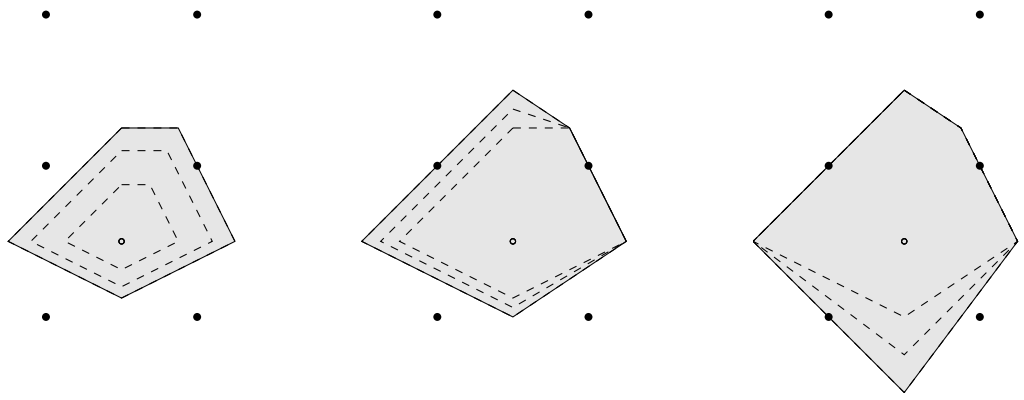
Lemma 28. *For every $\varepsilon > 0$ and $\beta_1, \dots, \beta_m \geq 0$, if the convex set*

$$B(\varepsilon, \beta) = \operatorname{conv}\left(\{f\} \cup \left\{f + \frac{1}{\max\{\varepsilon, \beta_i\}} r^i : i \in \{1, \dots, m\}\right\}\right)$$

is lattice-free, then inequality (4.7) is valid for (C).

Proof. Similar to the proof of Lemma 23. ■

A brief outline of the algorithm is as follows. First, all the variables β_i are set to zero. Then, the algorithm tries to find the smallest number $\varepsilon > 0$ such that $B(\varepsilon, \beta)$ is lattice-free. If $B(\varepsilon, \beta)$ is lattice-free for every $\varepsilon > 0$, then ε is set to zero. At this point, inequality (4.7) corresponds to the infinity cut (level 1). Next, for every ray r^i that has become tight with respect to (4.7), the algorithm updates β_i to ε. The process then repeats, and the algorithm tries to find, once again, the smallest $\varepsilon > 0$ such that $B(\varepsilon, \beta)$ is lattice-free. The algorithm stops after m iterations, since it is guaranteed, by then,



(a) $\varepsilon \geq \frac{2}{3}$ (b) $\frac{2}{3} \geq \varepsilon \geq \frac{1}{2}$ (c) $\frac{1}{2} \geq \varepsilon \geq \frac{1}{4}$

Figure 4.2: Example of $B(\varepsilon, \beta)$ for different values of ε

that every ray has become tight. We show the algorithm in a small example, before presenting its precise description.

Example 29. *Consider the continuous relaxation presented in Example 24. The algorithm starts by settings β to $(0,0,0,0,0)$. Figure 4.2a shows the set $B(\varepsilon, \beta)$ for values $\varepsilon = \frac{4}{3}, \frac{4}{5}, \frac{2}{3}$. Note that, as we decrease ε, the set $B(\varepsilon, \beta)$ grows. It is clear that, for any $\varepsilon < \frac{2}{3}$, the point $(1,1)$ belongs to the interior of $B(\varepsilon, \beta)$, therefore the desired ε for the first iteration is $\frac{2}{3}$. The algorithm then sets β_2 and β_3 to $\frac{2}{3}$, and continues to decrease ε. Figure 4.2b shows $B(\varepsilon, \beta)$ for values $\varepsilon = \frac{2}{3}, \frac{4}{7}, \frac{1}{2}$. Note that, even as ε decreases, the vertices corresponding to rays r^2 and r^3 remain fixed. Since any $\varepsilon < \frac{1}{2}$ causes the point $(0,1)$ to fall in the interior of $B(\varepsilon, \beta)$, the desired ε for the second iteration is $\frac{1}{2}$. The algorithm sets β_1 and β_5 to $\frac{1}{2}$ and continues decreasing ε. Figure 4.2c shows $B(\varepsilon, \beta)$ for values $\varepsilon = \frac{1}{2}, \frac{1}{3}, \frac{1}{4}$. Since $(0,0)$ belongs to the interior of $B(\varepsilon, \beta)$ for any $\varepsilon < \frac{1}{4}$, the desired ε for the third iteration is $\frac{1}{4}$. The algorithm sets β_4 to $\frac{1}{4}$. In the following iterations, the set $B(\varepsilon, \beta)$ is lattice-free for any choice of ε, and no new rays become tight. The algorithm terminates then with $\beta = (\frac{1}{2}, \frac{2}{3}, \frac{2}{3}, \frac{1}{4}, \frac{1}{2})$, which gives us the coefficients for the infinity cut (level 5).* ◇

A precise description of our method is presented in Algorithm 30. The algorithm relies on a function Bound, which, given $x \in \mathbb{Z}^n$ and $\beta_1, \ldots, \beta_m \geq 0$, either returns the largest $\varepsilon_x > 0$ such that $B(\varepsilon_x, \beta)$ contains x, given that this number exists, or returns zero in case it does not. In the following, we prove that InfinityCut correctly computes the infinity cut (level m). First, we prove technical lemma, which shows that, at the end of each iteration of the loop in line 3, the value of the variable ε^k decreases.

Algorithm 30

function InfinityCut
  $\beta_i^0 \leftarrow 0$, for every $i \in \{1, \dots, m\}$
  for $k \in \{1, \dots, m\}$ **do**
    $\varepsilon^k \leftarrow 0$
    for $x \in \mathbb{Z}^n$ **do**
      $\varepsilon_x^k \leftarrow \text{Bound}(\beta_1^{k-1}, \dots, \beta_m^{k-1}, x)$
      $\varepsilon^k \leftarrow \max\{\varepsilon^k, \varepsilon_x^k\}$
    $T^k \leftarrow \{i \in \{1, \dots, m\} : r^i \text{ is tight for } \sum_{i=1}^m \max\{\beta_i^{k-1}, \varepsilon^k\} s_i \geq 1\}$
    $\beta_i^k \leftarrow \max\{\beta_i^{k-1}, \varepsilon^k\}$, for every $i \in T^k$
    $\beta_i^k \leftarrow \beta_i^{k-1}$, for every $i \in \{1, \dots, m\} \setminus T^k$
  return $\beta_1^m, \dots, \beta_m^m$

Lemma 31. *If $\varepsilon^1, \dots, \varepsilon^m$ are defined as in Algorithm 30, then*

$$\varepsilon^1 \geq \varepsilon^2 \dots \geq \varepsilon^m.$$

Proof. Let $k \in \{2, \dots, m\}$. Suppose, by contradiction, that $\varepsilon^{k-1} < \varepsilon^k$. Since $\varepsilon^{k-1} \geq 0$, we must have $\varepsilon^k > 0$. This implies that $\varepsilon^k = \varepsilon_x^k = \text{Bound}(\beta_1^{k-1}, \dots, \beta_m^{k-1}, x)$, for some $x \in \mathbb{Z}^n$. By the definition of Bound, this implies $x \in B(\varepsilon^k, \beta_1^{k-1}, \dots, \beta_m^{k-1})$. Next, we prove that $B(\varepsilon^k, \beta_1^{k-1}, \dots, \beta_m^{k-1}) = B(\varepsilon^k, \beta_1^{k-2}, \dots, \beta_m^{k-2})$. Indeed,

$$\begin{aligned}
&B(\varepsilon^k, \beta_1^{k-1}, \dots, \beta_m^{k-1}) \\
&\quad = \text{conv}\left(\{f\} \cup \left\{f + \frac{1}{\max\{\varepsilon^k, \beta_i^{k-1}\}} r^i\right\}_{i=1}^m\right) \\
&\quad = \text{conv}\left(\{f\} \cup \left\{f + \frac{1}{\max\{\varepsilon^k, \varepsilon^{k-1}, \beta_i^{k-1}\}} r^i\right\}_{i \in T^{k-1}} \cup \left\{f + \frac{1}{\max\{\varepsilon^k, \beta_i^{k-2}\}} r^i\right\}_{i \in \{1,\dots,m\} \setminus T^{k-1}}\right) \\
&\quad = \text{conv}\left(\{f\} \cup \left\{f + \frac{1}{\max\{\varepsilon^k, \beta_i^{k-2}\}} r^i\right\}_{i=1}^m\right) \\
&\quad = B(\varepsilon^k, \beta_1^{k-2}, \dots, \beta_m^{k-2}).
\end{aligned}$$

This implies that $x \in B(\varepsilon^k, \beta_1^{k-2}, \dots, \beta_m^{k-2})$. Therefore, if $\varepsilon_x^{k-1} = \text{Bound}(\beta_1^{k-2}, \dots, \beta_m^{k-2}, x)$, then we must have $\varepsilon_x^{k-1} \geq \varepsilon^k$. On the other hand, we know that $\varepsilon^{k-1} \geq \varepsilon_x^{k-1}$. A contradiction. ■

Theorem 32. *If $k \in \{1, \dots, m\}$ and if $\beta_1^k, \dots, \beta_m^k$ and ε^k are defined as in Algorithm 30, then the inequality*

$$\sum_{i=1}^m \max\{\beta_i^{k-1}, \varepsilon^k\} s_i \geq 1 \tag{4.8}$$

is the infinity cut (level k).

Proof. First, we prove that the result holds at the end of the first iteration. In this case, equation (4.8) is given by

$$\sum_{i=1}^{m} \max\{\beta_i^0, \varepsilon^1\} s_i = \sum_{i=1}^{m} \varepsilon^1 s_i \geq 1.$$

By Lemma 28, in order to prove that this inequality is valid, it is sufficient to show that $B(\varepsilon^1, \beta^0)$ is lattice-free. Let $x \in \mathbb{Z}^n$ be an arbitrary lattice point and let $\varepsilon_x^1 = \textsc{Bound}(\beta^0, x)$. We prove that x is not in the interior of $B(\varepsilon^1, \beta^0)$. If $\varepsilon_x^1 = 0$, then there does not exist $\bar{\varepsilon} > 0$ such that $x \in B(\bar{\varepsilon}, \beta^0)$. In particular, $x \notin B(\varepsilon^1, \beta^0)$, and we are done. Suppose, on the other hand, that $\varepsilon_x^1 > 0$. By definition of Bound, we known that $B(\varepsilon_x^1, \beta^0)$ does not contain x in its interior. Also, from the description of the algorithm, have $\varepsilon^1 \geq \varepsilon_x^1$. Therefore, $B(\varepsilon^1, \beta^0)$ does not contain x in its interior either. Since the choice of x was arbitrary, we conclude that $B(\varepsilon^1, \beta^0)$ is lattice-free. Now we prove that ε^1 is as small as possible. If $\varepsilon^1 = 0$, there is nothing to prove. If $\varepsilon^1 > 0$, then, by the description of the algorithm, there exists $x \in \mathbb{Z}^n$ such that $\varepsilon^1 = \varepsilon_x^1 = \textsc{Bound}(\beta^0, x)$. By definition of Bound, for any $\bar{\varepsilon}$ such that $0 < \bar{\varepsilon} < \varepsilon^1$, the set $B(\bar{\varepsilon}, \beta^0)$ contains x in its interior. Therefore, ε^1 cannot be decreased without generating an invalid cut.

Now let $k \in \{2, \ldots, m\}$ and suppose

$$\sum_{i=1}^{m} \max\{\beta_i^{k-2}, \varepsilon^{k-1}\} s_i \geq 1 \tag{4.9}$$

is the infinity cut (level k). We prove that

$$\sum_{i=1}^{m} \max\{\beta_i^{k-1}, \varepsilon^{k}\} s_i \geq 1 \tag{4.10}$$

is the infinity cut (level $k+1$). By Lemma 31, $\varepsilon^k \geq \varepsilon^{k+1}$. Consider a ray r^i such that $i \in T^{k-1}$. By definition, $\beta_i^{k-1} = \max\{\beta_i^{k-2}, \varepsilon^{k-1}\}$, which implies that $\max\{\beta_i^{k-1}, \varepsilon^k\} = \max\{\beta_i^{k-2}, \varepsilon^{k-1}, \varepsilon^k\} = \max\{\beta_i^{k-2}, \varepsilon^k\}$. Suppose, on the other hand, that r^i is such that $i \notin T^{k-1}$. Since the set of tight rays only increases after each iteration, $i \notin T^1, \ldots, T^{k-2}$ and we have $\beta_i^{k-1} = 0$. Therefore, $\max\{\beta_i^{k-1}, \varepsilon^k\} = \varepsilon^k$. Equation (4.10) can be rewritten as

$$\sum_{i \in T^{k-1}} \max\{\beta_i^{k-2}, \varepsilon^{k-1}\} s_i + \sum_{\substack{i=1 \\ i \notin T^{k-1}}}^{m} \varepsilon^k s_i \geq 1. \tag{4.11}$$

This form matches item (ii) of Definition 26. More precisely, notice that the coefficients corresponding to the tight rays T^{k-1} have been kept unchanged when going from (4.9) to (4.11). Also, all the coefficients for the remaining rays are equal to some $\varepsilon^k \in \mathbb{R}_+$. It remains to show that (4.11) is valid and that ε^k cannot be made any smaller. The proof is very similar to the base case, and has been omitted. ■

Before we continue, we make a small modification to Algorithm 30 in order to make it finite. Note that the algorithm currently never terminates because of the loop in Line 5. If ε^k is initialized to a very small but strictly positive constant η, instead of zero, then this loop can be rewritten as

$$\textbf{for } x \in \mathbb{Z}^n \cap B(\varepsilon^k, \beta^{k-1})$$

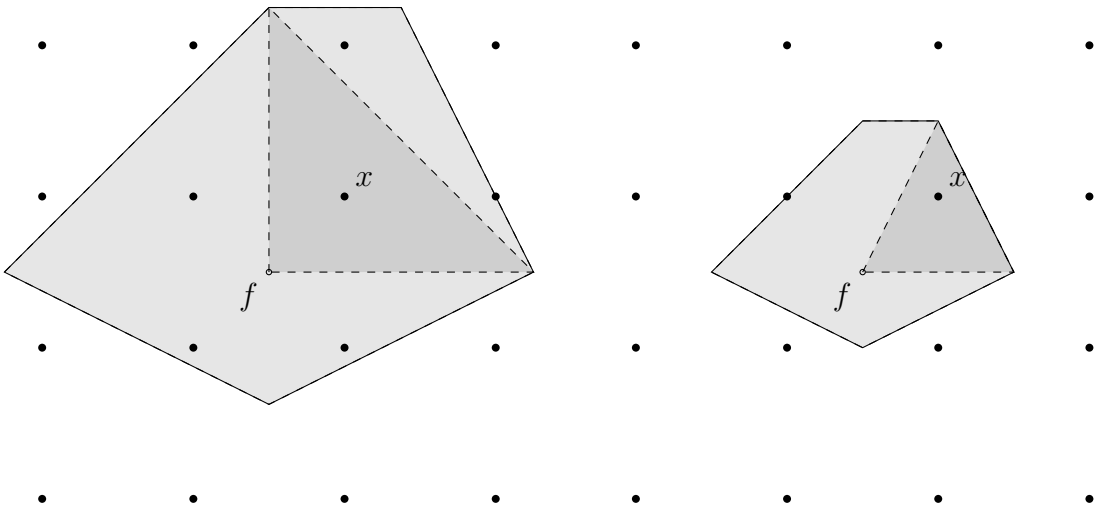
Since $B(\varepsilon^k, \beta^{k-1})$ is bounded, there are a finite number of lattice points contained in it, and the loop now terminates in finite time. Another advantage is that, whenever ε^k gets updated inside of the loop, the set $B(\varepsilon^k, \beta^{k-1})$ shrinks, causing the loop to terminate earlier. We note that this modification does change the output of the algorithm, and causes the algorithm to produce a slightly weaker cut whenever the infinity cut for the relaxation turns out to be a split. If the constant η is small enough, however, the practical impact is likely negligible. In our experiments, this constant was set to 10^{-3}.

Next, we present an algorithm for evaluating the function Bound. With our modification, this function now behaves as follows. Given $x \in \mathbb{Z}^n$ and $\beta_1, \ldots, \beta^m \geq 0$, if there exists $\varepsilon \geq \eta$ such that $B(\varepsilon, \beta)$ contains x, the function returns the largest of such ε. If no such ε exists, the function returns η. An outline of the algorithm is as follows. We start by setting $\varepsilon = \eta$ and verifying whether $B(\beta, \varepsilon)$ contains x in its interior. If so, there must exist a subset of linearly independent rays $r^{i_1}, \ldots, r^{i_s}$, where $s \leq n$, such that x belongs to the relative interior of the set

$$S(r^{i_1}, \ldots, r^{i_s}, \beta_{i_1}, \ldots, \beta_{i_s}, \varepsilon) = \operatorname{conv}\left(\{f\} \cup \left\{f + \frac{1}{\max\{\varepsilon, \beta_{i_j}\}} r^{i_j} : j \in \{1, \ldots, s\}\right\}\right).$$

Note that this set was obtained from $B(\beta, \varepsilon)$ by dropping some of the rays r^i, and therefore has a simpler structure. The algorithm then computes ε such that x belongs to the boundary of this simplified set, which can be done easily. Next, the algorithm verifies whether x still belongs to the interior of the original set $B(\beta, \varepsilon)$, and, if so, the process repeats. The algorithm stops when x is no longer in the interior of $B(\varepsilon, \beta)$. We show the execution of this algorithm in a small example.

Example 33. *Consider, once again, the continuous relaxation presented in Example 24.*



(a) First iteration (b) Second iteration

Figure 4.3: Examples for BOUND

Let $\beta = (0,0,0,0,0), x = (1,1)$ *and suppose we want to compute* BOUND(β, x)*. The algorithm starts by setting* ε *to some very small* η*. The set* $B(\varepsilon, \beta)$ *is shown in Figure 4.3a. Clearly,* x *belongs to the interior of* $B(\varepsilon, \beta)$*. Notice that* x *also belongs to the relative interior of* $S(r^1, r^3, \beta_1, \beta_3, \varepsilon)$ *for any* $\varepsilon < \frac{1}{2}$*. The algorithm then sets* ε *to* $\frac{1}{2}$ *and verifies that* x *still belongs to the interior of* $B(\varepsilon, \beta)$*, as shown in Figure 4.3b. Furthermore,* x *belongs to the interior of* $S(r^2, r^3, \beta_2, \beta_3, \varepsilon)$ *for any* $\varepsilon < \frac{2}{3}$*. After setting* ε *to* $\frac{2}{3}$*, it can be verified that* x *no longer belongs to the interior of* $B(\varepsilon, \beta)$*, and the algorithm stops.* $\diamond$

A precise description of BOUND is given in Algorithm 34. In order to find the desired subset of linearly independent rays $r^{i_1}, \ldots, r^{i_s}$, or to prove that no such subset exists, BOUND calls the function FINDVIOLATEDCONE, which is described precisely in Algorithm 35, and in order to find ε such that x belongs to the boundary of $S(r^{i_1}, \ldots, r^{i_s}, \beta^{i_1}, \ldots, \beta^{i_s}, \varepsilon)$, BOUND calls the function CONEBOUND, which is described precisely in Algorithm 36. In the following, we prove that these algorithms indeed work as intended. We start with FINDVIOLATEDCONE.

Proposition 37. *Let* $\varepsilon > 0$*. If* FINDVIOLATEDCONE *returns a non-empty set* $\{i_1, \ldots, i_s\} \subseteq \{1, \ldots, m\}$*, then*

(i) $r^{i_1}, \ldots, r^{i_s}$ *are linearly independent*

(ii) x *belongs to the relative interior of* $S(r^{i_1}, \ldots, r^{i_s}, \beta_{i_1}, \ldots, \beta_{i_s}, \varepsilon)$

If FINDVIOLATEDCONE *returns* $\emptyset$*, then no subset* $\{i_1, \ldots, i_s\} \subseteq \{1, \ldots, m\}$ *satisfies (i) and (ii).*

Algorithm 34

function BOUND($\beta_1, \ldots, \beta_m, x$)
  $\varepsilon \leftarrow \eta$
  loop
    $Q \leftarrow$ FINDVIOLATEDCONE($\beta_1, \ldots, \beta_m, \varepsilon, x$)
    if $Q = \emptyset$ **then**
      break
    else
      Let $Q = \{i_1, \ldots, i_s\}$, where $\beta_{i_1} \leq \ldots \leq \beta_{i_s}$
      $\varepsilon \leftarrow \max\{\varepsilon,$ CONEBOUND($r^{i_1}, \ldots, r^{i_s}, \beta_{i_1}, \ldots, \beta_{i_s}, x$)$\}$
  return ε

Algorithm 35

function FINDVIOLATEDCONE($\beta_1, \ldots, \beta_m, \varepsilon, x$)
  Let λ^* be an optimal basic solution for

$$\begin{aligned} \textbf{minimize} \quad & \sum_{j=1}^{m} \lambda_j \\ \textbf{subject to} \quad & x = f + \sum_{j=1}^{m} \frac{\lambda_j}{\max\{\varepsilon, \beta_j\}} r^j \\ & \lambda_r \geq 0 \end{aligned}$$

  if $\sum_{j=1}^{m} \lambda_j^* \geq 1$ **then return** $\emptyset$
  else return $\{i \in \{1, \ldots, m\} : \lambda_i^* > 0\}$

Algorithm 36

function CONEBOUND($q^1, \ldots, q^s, \beta^1, \ldots, \beta^s, x$)
  Let $\lambda \in \mathbb{R}^s$ such that $x = f + \sum_{i=1}^{s} \lambda_i q^i$
  for $t = 0$ **to** $s - 1$ **do**
    Let $\varepsilon_t \leftarrow \dfrac{1 - \sum_{i=1}^{t} \lambda_i \beta_i}{\sum_{i=t+1}^{s} \lambda_i}$
    if $\varepsilon_t > \beta_{t+1}$ **then return** ε_t
  return 0

Proof. Suppose FindViolatedCone returns a non-empty subset $\{i_1, \ldots, i_s\} \subseteq \{1, \ldots, m\}$, and let λ^* be defined as in the algorithm. Property (i) follows immediately from the fact that λ^* is a basic feasible solution. Now we prove that x belongs to the relative interior of $S := S(r^{i_1}, \ldots, r^{i_s}, \beta_{i_1}, \ldots, \beta_{i_s}, \varepsilon)$. If we denote by v^j the point $f + \frac{1}{\max\{\beta_j, \varepsilon\}} r^j$, for $j \in \{1, \ldots, m\}$, then the vertices of S are $f, v^{i_1}, \ldots, v^{i_s}$. It suffices to prove that x is a strict convex combination of these vertices. Recall that $\sum_{j=1}^m \lambda_j^* < 1$. Since each $\lambda_j^* > 0$, this also implies $\lambda_j^* < 1$, for every $j \in \{1, \ldots, m\}$. Furthermore, if we define $\lambda_0^* = 1 - \sum_{j=1}^m \lambda_j^*$, then $0 > \lambda_0^* > 1$. We have

$$\begin{aligned} x &= f + \sum_{j=1}^{m} \frac{\lambda_j^*}{\max\{\varepsilon, \beta_j\}} r^j \\ &= f + \sum_{j=1}^{s} \lambda_{i_j}^* \left(v^{i_j} - f\right) \\ &= \lambda_0^* f + \lambda_{i_1}^* v^{i_1} + \ldots + \lambda_{i_s}^* v^{i_s} \end{aligned}$$

We conclude that (ii) is satisfied.

Now suppose FindViolatedCone returns the empty set. In this case, $\sum_{j=1}^m \lambda_j^* \geq 1$. Also suppose, by contradiction, that there exists a non-empty set $\{i_1, \ldots, i_s\} \subseteq \{1, \ldots, m\}$ satisfying (i) and (ii). Let the points v^j be as defined previously. We know there exist $0 > \lambda_0, \lambda_{i_1}, \ldots, \lambda_{i_s} > 1$ such that

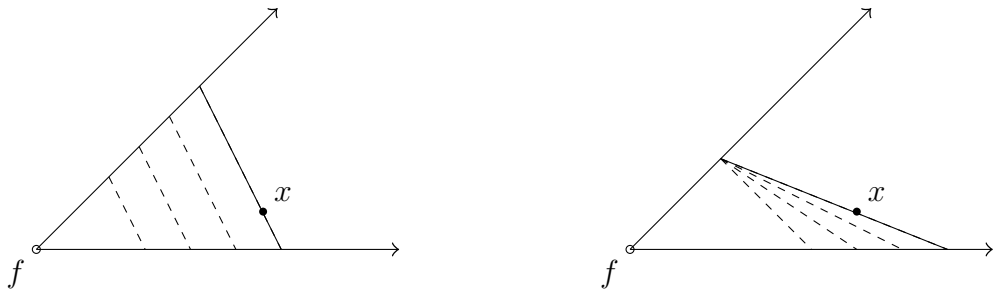
$$\begin{cases} x = \lambda_0 f + \lambda_{i_1} v^{i_1} + \ldots + \lambda_{i_s} v^{i_s} \\ \lambda_0 + \lambda_{i_1} + \ldots + \lambda_{i_s} = 1 \end{cases}$$

Let $\lambda_j = 0$ for every $j \in \{1, \ldots, m\} \setminus \{i_1, \ldots, i_s\}$. It is not hard to verify that

$$x = f + \sum_{j=1}^{m} \frac{\lambda_j}{\max\{\varepsilon, \beta_j\}} r^j.$$

Therefore, $\lambda_1, \ldots, \lambda_m$ is a feasible solution to the LP in the algorithm. Furthermore, $\sum_{j=1}^m \lambda_j = 1 - \lambda_0 < 1 \leq \sum_{j=1}^m \lambda_j^*$. This implies that λ^* is not an optimal solution, a contradiction. We conclude that no subset $\{i_1, \ldots, i_s\} \subseteq \{1, \ldots, m\}$ satisfying (i) and (ii) exists. ■

Next, we focus on the function ConeBound. First, we present a technical lemma which justifies the formula for ε_t that appears in the algorithm, and then a proposition which proves that ConeBound finds the correct answer.



(a) Computation of ε_0 (b) Computation of ε_1

Figure 4.4: Illustration for ConeBound

Lemma 38. *Let $x \in \mathbb{Z}^n$, let $\beta_1, \dots, \beta_s \in \mathbb{R}$ such that $\beta_1 \geq \dots \geq \beta_s \geq 0$, let $q^1, \dots, q^s \in \mathbb{R}^n$ be linearly independent vectors such that x belongs to the relative interior of $f + \operatorname{cone}(q^1, \dots, q^s)$, and let $\lambda_1, \dots, \lambda_s > 0$ such that $x = f + \sum_{i=1}^s \lambda_i q^i$. Let $t \in \{0, \dots, s-1\}$, and define*

$$\varepsilon_t = \frac{1 - \sum_{i=1}^t \lambda_i \beta_i}{\sum_{i=t+1}^s \lambda_i}.$$

If $\beta_1, \dots, \beta_t > 0$ and $\varepsilon_t > 0$, then ε_t is the largest $\varepsilon \in \mathbb{R}$ such that

$$x \in \operatorname{conv}\left(\{f\} \cup \left\{f + \frac{1}{\beta_i} q^i\right\}_{i=1}^t \cup \left\{f + \frac{1}{\varepsilon} q^i\right\}_{i=t+1}^s\right) =: S_t(\varepsilon)$$

Proof. Let $v^i = f + \frac{1}{\beta_i} q^i$ for $i \in \{1, \dots, t\}$ and $v^i = f + \frac{1}{\varepsilon_t} q^i$ for $i \in \{t+1, \dots, s\}$. We prove that x is a convex combination of $v^1, \dots, v^s$. This implies that $x \in S_t(\varepsilon_t)$, and it is not hard to see that, for any $\varepsilon > \varepsilon_t$, we have $x \notin S_t(\varepsilon)$. To prove this claim, let

$$\gamma_i = \begin{cases} \lambda_i \beta_i & \text{for } i \in \{1, \dots, t\} \\ \lambda_i \varepsilon_t & \text{for } i \in \{t+1, \dots, s\} \end{cases}$$

By hypothesis, we have $\lambda_1, \dots, \lambda_s > 0$ and $\beta_1, \dots, \beta_t, \varepsilon_t > 0$. This implies that $\gamma_1, \dots, \gamma_t > 0$. Furthermore, the sum of $\gamma_1, \dots, \gamma_t$ equals one, since

$$\sum_{i=1}^s \gamma_i = \sum_{i=1}^t \lambda_i \beta_i + \varepsilon_t \left(\sum_{i=t+1}^s \lambda_i\right) = \sum_{i=1}^t \lambda_i \beta_i + 1 - \sum_{i=1}^t \lambda_i \beta_i = 1.$$

It only remains to prove that $x = \sum_{i=1}^s \gamma_i v^i$. We have:

$$\sum_{i=1}^s \gamma_i v^i = \sum_{i=1}^s \gamma_i f + \sum_{i=1}^t \lambda_i \beta_i \frac{1}{\beta_i} q^i + \sum_{i=t+1}^s \lambda_i \varepsilon_t \frac{1}{\varepsilon_t} q^i = f + \sum_{i=1}^s \lambda_i q^i.$$

■

Proposition 39. *Let $f, x, \beta_1, \ldots, \beta_s$ and $q^1, \ldots, q^s$ be defined as in Lemma 38. If there exists $\varepsilon > 0$ such that x belongs to the interior of $S(q^1, \ldots, q^s, \beta_1, \ldots, \beta_s, \varepsilon) := S(\varepsilon)$, then* ConeBound *returns the largest $\varepsilon \in \mathbb{R}$ such that $x \in S(\varepsilon)$. Otherwise,* ConeBound *returns zero.*

Proof. Suppose there exists $\tilde{\varepsilon} > 0$ such that x belongs to the interior of $S(\tilde{\varepsilon})$. First, we prove that there exists $\varepsilon^* > \beta_s$ such that $x \in S(\varepsilon^*)$. If $\tilde{\varepsilon} > \beta_s$, there is nothing to prove, so suppose $\tilde{\varepsilon} \leq \beta_s$. Since $\beta_s \leq \ldots \leq \beta_1$, this implies $S(\tilde{\varepsilon}) = S(\beta_s)$. Therefore, x belongs to the interior of $S(\beta_s)$. It is not hard to see that, for a very small $\theta > 0$, we still have $x \in S(\beta_s + \theta)$. Therefore, there exists $\varepsilon^* > \beta_s$ such that $x \in S(\beta_s + \theta)$. Furthermore, we may assume that ε^* is the maximum value satisfying this property. Indeed, if such maximum does not exist, then $x \in S(\varepsilon)$ for every $\varepsilon > \varepsilon^*$, which implies $f = x$, a contradiction. Next, we prove that ConeBound returns ε^*. Let $\beta_0 = \infty$. There exists $k \in \{0, \ldots, s-1\}$ such that $\beta_{k+1} < \varepsilon^* \leq \beta_k$. By Lemma 38, we know that $\varepsilon^* = \varepsilon_k$. We prove that ConeBound returns ε_k. Since $\varepsilon_k = \varepsilon^* > \beta_{k+1}$, it is sufficient to show that the loop does not finish before considering ε_k. That is, that ConeBound does not return $\varepsilon_0, \ldots, \varepsilon_{k-1}$. Suppose, by contradiction, that ConeBound returns ε_l, for some $l \in \{0, \ldots, k-1\}$. Then $\varepsilon_l > \beta_{l+1} \geq \beta_k \geq \varepsilon^*$. Furthermore, $x \in S(\varepsilon_l)$, contradicting the maximality of ε^*. We conclude that the algorithm returns ε^*.

Now, suppose that ConeBound returns a non-zero value. Clearly, the returned value must be ε_k, for some $k \in \{0, \ldots, s-1\}$. Furthermore, $x \in S(\varepsilon_k)$ and $\varepsilon_k > \beta_{k+1} \geq \beta_s$. Let $\tilde{\varepsilon} = \frac{\varepsilon_k + \beta_s}{2}$. It is not hard to see that $\tilde{\varepsilon} > 0$ and that x belongs to the interior of $S(\tilde{\varepsilon})$. We conclude that there exists $\varepsilon > 0$ such that x belongs to the interior of $S(\varepsilon)$.

■

We end this section with a proof of correctness for the function Bound.

Proposition 40. *Let $x \in \mathbb{Z}^n$, let $\beta_1, \ldots, \beta_m \geq 0$ and let $\eta > 0$. If there exists $\varepsilon \geq \eta$ such that x belongs to the interior of $B(\beta_1, \ldots, \beta_m, \varepsilon) =: B(\varepsilon)$, then* Bound *returns the largest $\varepsilon \geq \eta$ such that $x \in B(\varepsilon)$. Otherwise,* Bound *returns η.*

Proof. First, suppose there does not exist $\varepsilon \geq \eta$ such that x belongs to the interior of $B(\varepsilon)$. Then, clearly, x does not belong to the interior of $B(\eta)$ and, by Proposition 37, the function FindViolatedCone returns the empty set. The loop then breaks during the first iteration. Therefore, Bound returns η.

Now suppose there exists $\varepsilon \geq \eta$ such that x belongs to the interior of $B(\varepsilon)$. Since $B(\varepsilon) \subseteq B(\eta)$, then x also belongs to the interior of $B(\eta)$. This implies that the value ε^*

returned by Bound equals the value returned by ConeBound, for some subset of rays $r^{i_1}, \ldots, r^{i_s}$. By Proposition 39, we have $x \in B(\varepsilon^*)$. Now suppose, by contradiction, that there exists $\tilde{\varepsilon} > \varepsilon^*$ such that $x \in B(\tilde{\varepsilon})$. This implies that x belongs to the interior of $B(\varepsilon^*)$, a contradiction. We conclude that ε^* is the largest $\varepsilon \geq \eta$ such that $x \in B(\varepsilon)$. ■

4.3 Computational experiments

In order to evaluate the strength of infinity cuts, we implemented a cut generator and tested it with problems from MIPLIB. Our main performance indicator was the integrality gap closed at the root node of the branch-and-bound tree. We compared our cuts against Gomory Mixed-Integer (GMI) cuts, and also against an exact separator for continuous multi-row intersection cuts implemented by Louveaux, Poirrier and Salvagnin [60, 61]. Our main goal was to determine how close to the maximum theoretical gap closure would infinity cuts bring us. Another side goal was to evaluate the efficiency our cut generator.

The cut generator was implemented in C and compiled with GNU GCC 5.4.0. The generator makes use of IBM ILOG CPLEX 12.4.0 as a LP/MIP solver. The complete source code has been made available online [67]. We conducted our experiments on an Intel Xeon E5-2630v2 2.6 GhZ with 64GB of memory, with a time limit of 15 CPU minutes.

4.3.1 Generating infinity cuts from the tableau

When generating a multi-row intersection cut from the simplex tableau, the first step is to select a suitable combination of tableau rows that will be used to derive the cut. Suppose that the simplex tableau is given by

$$x_i + \sum_{j \in N} \bar{a}_{ij} x_j = b_i, \qquad i \in B$$

where x_i, for $i \in B$, are the basic variables, and x_j, for $j \in N$, are the non-basic variables. In principle, any subset of tableau rows $I \subseteq B$ can be used, as long as two conditions are satisfied: first, only rows corresponding to integral basic variables are selected; and second, at least one row with a fractional right-hand side is included in the subset. In practice, however, the number of subsets satisfying these two conditions is excessively large for all but very small instances. Our cut generator, therefore, filtered down the list of row subsets even further, as described next.

As a first step, the rows of the simplex tableau were sorted according to the fractionality of their right-hand side. More specifically, the rows were sorted increasingly according to $\left|\widehat{b_i} - \frac{1}{2}\right|$, where $\widehat{b_i}$ is the fractional part of b_i. In this way, rows having right-hand sides closer to $\frac{1}{2}$ received higher priority, while rows having right-sides closer to 0 or 1 were given lower priority. Then, only 300 tableau rows with the most fractional right-hand sides were kept, while the remaining rows were discarded.

Next, for each pair of remaining tableau rows, we computed how similar was their support. More specifically, given two tableau rows $x_p + \sum_{j \in N} \bar{a}_{pj} x_j = b_p$ and $x_q + \sum_{j \in N} \bar{a}_{qj} x_j = b_q$, their *affinity* is given by the formula

$$\textsc{Affinity}(p, q) = \frac{2 \times |\{j : \bar{a}_{pj} \neq 0 \text{ and } \bar{a}_{qj} \neq 0\}|}{|\{j : \bar{a}_{pj} \neq 0\}| + |\{j : \bar{a}_{qj} \neq 0\}|.}$$

Note that, for every pair of tableau rows, this formula returns one if $\bar{a}_p$ and $\bar{a}_q$ have exactly the same support and zero if their supports are completely disjoint. This heuristic was also used by Louveaux, Poirrier and Salvagnin [61]. Pairs of tableau rows with affinity less than a certain threshold were considered incompatible. The generator derived one infinity cut for each subset of tableau rows containing only pairwise compatible rows.

After selecting a subset $I = \{i_1, \ldots, i_k\} \subseteq B$ to generate the cut, another issue was to decide what set of rays to use when generating the infinity cut. On one hand, we wanted the set of rays to be as close as possible to the original columns of the simplex tableau. On the other hand, the running time of the algorithm described in the previous section is highly sensitive on the number of rays, and, for instances having a large number of variables, it proved to be too slow to be useful in practice. Our separator, therefore, performed the following heuristic to reduce the number of rays considered. For each non-basic variable x_j, a ray $r^j = (\bar{a}_{ij} : i \in B)$ was created. Rays having zeros in all components were discarded. Then, for each pair of rays that were very similar to each other, only one was kept. More specifically, for every pair of rays r^1, r^2, we computed $\|r^1 - r^2\|$, and, whenever this value was smaller than a certain threshold, one of these rays was discarded. The threshold was initially set to infinity and then slowly decreased until at most 100 rays were selected.

4.3.2 Cut generating procedure

Our cut generator performed the following steps. First, the linear relaxation of the problem was solved, and a certain basic solution with value z_{LP} with value z_{LP} was obtained.

The optimal simplex tableau was stored. Although this relaxation was later strengthened and solved again, only this first optimal simplex tableau was used to generated all subsequent cutting planes, hence we only obtain rank-1 cuts. Next, one GMI cut was generated from each suitable tableau row, and added to the relaxation. The relaxation was solved again, and a certain solution with value z_{GMI} was found. Then, one infinity cut was generated at a time, for each suitable subset of tableau rows, as described in the previous subsection. If the infinity cut did not cut the current basic optimal solution, the cut was discarded. Otherwise, the infinity cut was added to the relaxation, the relaxation was solved once again, and the current basic solution was updated. At the end of the procedure, a new solution with value z_{INF} was obtained. Finally, we also ran, independently from our generator, the exact separator for continuous multi-row cuts, and obtained a solution with value z_{EXACT}. Since different bases cause slightly different results, we were careful to use the same basis for all cut generators.

The set of instances was the MIPLIB 3.0, which contains 65 instances coming from real-world applications. We chose this version of the MIPLIB so that the exact separator could produce more reliable bounds. Out of the 65 instances, we eliminated those that were infeasible, that had no known optimal solution, or whose LP relaxation had the same objective value as the original problem. Three instances were also eliminated because of their size. For each of the remaining 56 instances, we computed:

- G-GMI, the gap closed by the inclusion of GMI cuts:

$$\text{G-GMI} = \frac{z_{GMI} - z_{LP}}{z_{OPT} - z_{LP}},$$

- G-INF, the gap closed by the inclusion of GMI cuts, in addition to the infinity cuts:

$$\text{G-INF} = \frac{z_{INF} - z_{LP}}{z_{OPT} - z_{LP}},$$

- G-EXACT, the gap closed by using the exact separator for continuous multi-row cuts:

$$\text{G-EXACT} = \frac{z_{EXACT} - z_{LP}}{z_{OPT} - z_{LP}},$$

- REL, a relative measure of strength, in terms of gap closure, when compared to the exact separator, given precisely by:

$$\text{REL} = \frac{z_{INF} - z_{GMI}}{z_{EXACT} - z_{GMI}}.$$

- T-TIME, the total CPU running time spent by the infinity cut generator (min:sec),
- C-TIME, the CPU running time that the infinity cut generator, on average, took to generate a single cut, in milliseconds,
- N-CUTS, the number of infinity cuts added to the relaxation.

4.3.3 Results and discussion

Table 4.1 shows the computational results for infinity cuts generated from two rows of the simplex tableau. We show both the strength of non-lifted infinity cuts, where no attempt was made to strengthen the coefficients of the non-basic variables, and for lifted infinity cuts, where the coefficients for the integral non-basic variables were strengthened using the trivial lifting procedure described in Section 2.3.

For the set of 56 instances processed, the average gap closed by GMI cuts, by non-lifted infinity cuts, by lifted infinity cuts, and by the exact separator was, respectively, 29.0%, 31.3%, 31.8% and 33.9%. Therefore, in terms of gap closure, non-lifted infinity cuts provided, on average, 47.8% of the benefits of adding the entire class of continuous 2-row intersection cuts, while lifted infinity cuts provided, on average 57.8%.

The total CPU running time necessary to generate the non-lifted infinity cuts was on average very low, at 20 seconds, while the median was below 1 second. After lifting, the average running time increased to 2 minutes and 27 seconds. In comparison, the average running time for the exact separator was 1 hour and 26 minutes, with many instances reaching the time limit of 4 hours. Because of this time limit, the infinity cut generator was able to obtain a better gap closure than the exact separator in several instances. The infinity cut generator added, on average, 5.6 non-lifted cuts and 7.4 lifted cuts per instance. In comparison, the exact separator added, on average 42 cuts per instance.

For 21 out of the 56 instances processed, the exact separator was not able to improve upon the gap closed by the GMI cuts, while the non-lifted infinity cuts were not able to improve upon the gap for 32 instances. We note however that, for these instances, the total running time to generate these cuts was only 22 seconds and the number of cuts added was only 0.3, on average. Therefore, even for instances where non-lifted infinity cuts were not helpful, they would also probably not significantly affect the total running time of the branch-and-bound algorithm. Lifted infinity cuts were not able to improve the gap for 25 instances. The total running time for these instances was 3 minutes and 26 seconds, while the number of added cuts was 0.5, on average.

We also ran computational experiments for lifted and non-lifted infinity cuts from 3 rows of the simplex tableau. Table 4.2 shows these results. Here, we compare against

the exact separator for continuous 3-row intersection cuts. For the set of 56 instances processed, the average gap closed by GMI, by non-lifted infinity cuts, by lifted infinity cuts, and by the exact separator was, respectively, 29.0%, 32.2%, 32.7% and 36.1%. Non-lifted infinity cuts provided, on average, 44.9% of the benefits of the exact separator, while lifted infinity cuts provided 52.2%. This suggests that the relative strength of infinity cuts is preserved for rows with more relaxations.

The total CPU running time per instance was 3 minutes and 52 seconds for the non-lifted infinity cuts, and 6 minutes and 6 seconds for the lifted ones, on average. In comparison, the exact separator spent 2 hours and 27 minutes, on average. The infinity cut generator added, on average, 7.8 non-lifted cuts and 10.0 lifted cuts — numbers considerably lower than the 103 cuts added on average by the exact separator.

For illustration purposes, we finish this chapter with Figure 4.5, which shows some of the lattice-free sets produced by the infinity cut generator for a particular instance. The generator mostly produced sets that are approximately quadrilaterals, and also a few truncated triangles. While we can verify that all of these cuts are minimal, we can observe that the corresponding lattice-free sets are seldom maximal. While this has no impact on the strength of non-lifted infinity cuts, it does negatively impact the effectiveness of the lifting procedure.

			Non-Lifted 2-Row					Lifted 2-Row				
Instance	G-GMI	G-EXACT	G-INF	REL (%)	T-TIME (m:s)	C-TIME (ms)	N-CUTS	G-INF	REL (%)	T-TIME (m:s)	C-TIME (ms)	N-CUTS
10teams	57.1	57.1	57.1	—	04:18	52	0	57.1	—	15:00	527	0
air03	100.0	100.0	100.0	—	01:53	51	0	100.0	—	15:00	1410	0
bell3a	39.0	52.5	48.1	66.9	04:36	882	6	49.0	74.0	00:33	104	8
bell5	14.5	17.8	14.6	1.2	00:00	2	2	14.6	1.2	00:03	10	2
blend2	16.0	18.3	16.0	0.0	00:00	5	0	16.1	1.7	00:00	16	1
cap6000	41.6	41.6	41.6	—	00:00	651	0	41.6	—	00:02	2633	0
danoint	1.7	1.7	1.7	—	00:06	14	3	1.7	—	00:09	8	1
dcmulti	48.1	50.2	49.1	47.4	00:01	3	8	49.3	58.8	00:03	5	9
egout	65.1	82.3	75.9	63.0	00:00	0	2	75.9	63.0	00:00	2	2
fiber	70.3	71.6	71.6	99.7	00:02	1	3	73.9	274.9	00:36	26	12
fixnet6	22.6	26.6	27.3	119.2	00:00	2	9	30.9	209.1	00:01	20	16
flugpl	10.8	13.7	10.8	0.0	00:00	0	0	10.8	0.0	00:00	1	0
gen	1.3	6.8	8.1	123.3	00:00	1	10	6.2	88.9	00:00	8	15
gesa2	27.7	45.0	29.2	8.9	00:00	1	22	29.0	7.6	00:02	4	19
gesa2_o	29.2	56.8	34.8	20.2	00:01	1	30	34.8	20.4	00:06	5	28
gesa3	19.5	46.4	30.8	42.2	01:01	18	15	30.5	40.7	00:33	10	16
gesa3_o	59.1	72.4	64.4	40.0	00:05	1	15	65.0	44.3	00:56	13	14
gt2	59.9	100.0	89.7	74.4	00:00	1	2	90.6	76.7	00:07	29	2
harp2	24.1	26.8	24.2	1.9	00:02	5	3	24.2	2.0	01:14	155	6
khb05250	74.3	79.9	74.3	0.0	00:00	19	0	74.3	0.0	00:00	16	0
l152lav	12.9	12.9	12.9	—	01:10	21	0	13.0	—	15:00	518	6
lseu	37.0	37.3	37.0	0.0	00:01	6	0	39.0	732.1	00:05	21	4
markshare1	0.0	0.0	0.0	—	00:00	2	0	0.0	—	00:00	14	1
markshare2	0.0	0.0	0.0	—	00:00	2	0	0.0	—	00:00	27	0
mas74	6.7	6.7	6.7	0.0	00:00	4	0	7.0	675.8	00:04	67	10
misc03	15.0	15.0	15.0	—	00:02	2	0	15.0	—	00:55	29	0
misc06	31.4	37.6	37.6	99.7	00:00	4	5	41.3	157.9	00:00	4	5
misc07	0.7	0.7	0.7	—	00:09	2	0	0.7	—	03:19	46	0

Table 4.1: Strength of 2-row infinity cuts

			Non-Lifted 2-Row					Lifted 2-Row				
Instance	G-GMI	G-EXACT	G-INF	REL (%)	T-TIME (m:s)	C-TIME (ms)	N-CUTS	G-INF	REL (%)	T-TIME (m:s)	C-TIME (ms)	N-CUTS
mitre	84.0	84.0	84.0	—	00:02	2	1	84.0	—	01:30	94	3
mkc	8.7	10.6	8.7	0.0	00:02	11	0	8.7	0.0	00:41	216	1
mod008	21.6	21.6	21.6	—	00:00	8	0	21.6	—	00:02	167	0
mod010	100.0	100.0	100.0	—	00:45	15	0	100.0	—	15:00	486	0
mod011	33.0	34.6	33.0	0.0	00:06	13	0	33.0	0.0	00:12	26	0
modglob	17.3	26.5	17.5	2.4	00:01	6	5	17.5	2.7	00:02	12	5
nw04	66.1	66.1	66.1	—	00:57	350	0	66.1	—	15:09	11968	0
p0033	34.4	35.0	34.4	0.0	00:00	0	0	34.8	70.6	00:00	7	2
p0201	6.5	6.5	6.5	—	00:02	1	0	6.5	—	01:03	27	0
p0282	3.2	8.2	6.2	60.8	00:11	6	4	6.2	60.8	00:40	21	10
p0548	61.7	63.2	61.7	0.0	00:01	5	0	61.7	0.3	00:06	31	2
p2756	58.0	58.0	58.0	—	00:02	14	3	58.0	—	00:08	40	1
pk1	0.0	0.0	0.0	—	00:00	3	0	0.0	—	00:01	17	0
pp08a	54.9	66.7	67.1	103.3	00:00	2	26	67.0	103.2	00:00	3	26
pp08aCUTS	33.8	39.3	37.7	70.0	00:03	6	11	37.8	73.0	00:04	7	12
qiu	1.4	1.7	1.7	95.8	00:05	7	9	1.7	94.2	00:06	7	11
qnet1	11.7	14.6	12.0	8.4	00:53	11	5	14.7	106.5	15:00	222	14
qnet1_o	19.8	25.4	22.3	45.2	00:04	5	17	23.6	67.6	01:20	95	27
rentacar	19.8	20.3	19.8	0.0	00:00	14	0	19.8	0.0	00:01	21	0
rgn	12.0	12.0	12.0	—	00:00	1	0	15.1	—	00:00	9	4
rout	0.3	5.2	0.3	0.0	00:34	34	0	0.3	0.0	01:43	103	3
set1ch	29.4	55.6	44.0	55.7	00:00	2	87	44.0	55.7	00:00	4	87
seymour	6.6	6.6	6.6	—	01:13	15	0	6.6	—	14:59	209	0
stein27	0.0	0.0	0.0	—	00:02	1	0	0.0	—	00:26	5	0
stein45	0.0	0.0	0.0	—	00:04	1	0	0.0	—	00:47	9	0
swath	8.7	10.7	8.7	0.0	00:46	21	4	10.6	93.3	15:00	792	5
vpm1	29.9	31.3	30.7	54.8	00:00	1	5	33.8	280.9	00:00	5	5
vpm2	13.9	19.4	15.6	31.2	00:00	1	5	18.6	83.9	00:00	5	17
	29.0	33.9	31.3	47.8	00:20	41.2	5.7	31.8	57.8	02:27	363.1	7.4

Table 4.1: Strength of 2-row infinity cuts (continued)

			Non-Lifted 3-Row					Lifted 3-Row				
Instance	G-GMI	G-EXACT	G-INF	REL (%)	T-TIME (m:s)	C-TIME (ms)	N-CUTS	G-INF	REL (%)	T-TIME (m:s)	C-TIME (ms)	N-CUTS
10teams	57.1	57.1	57.1	—	15:00	116	0	57.1	—	15:00	557	0
air03	100.0	100.0	100.0	—	15:00	212	0	100.0	—	15:00	1465	0
bell3a	39.0	56.0	48.2	53.8	05:23	87	9	49.0	58.7	03:29	57	8
bell5	14.5	18.0	14.6	1.2	01:01	27	2	14.6	1.2	02:40	70	2
blend2	16.0	18.8	16.0	0.0	00:02	42	0	16.1	1.4	00:06	98	1
cap6000	41.6	41.6	41.6	—	00:10	10416	0	41.6	—	00:11	11880	0
danoint	1.7	1.7	1.7	—	01:50	70	3	1.7	—	03:33	47	1
dcmulti	48.1	52.5	49.1	23.4	01:56	23	12	49.6	36.2	03:34	43	31
egout	65.1	84.0	76.3	59.2	00:00	11	4	76.3	59.2	00:00	17	4
fiber	70.3	71.6	71.6	99.7	03:13	42	4	74.1	295.2	13:24	176	19
fixnet6	22.6	26.8	27.8	124.3	00:09	31	22	30.9	200.0	00:48	166	20
flugpl	10.8	62.7	39.1	54.6	00:02	30	3	39.1	54.6	00:03	36	3
gen	1.3	18.1	18.6	103.1	00:43	92	21	18.6	103.1	01:08	147	31
gesa2	27.7	45.0	30.1	14.0	01:03	19	36	30.3	14.9	02:30	45	40
gesa2_o	29.2	56.8	35.0	21.0	02:14	23	36	35.1	21.3	05:12	54	41
gesa3	19.5	46.4	31.6	44.9	02:07	21	19	31.6	45.0	02:13	21	20
gesa3_o	59.1	72.4	64.4	40.0	01:19	10	15	65.0	44.3	06:03	46	14
gt2	59.9	100.0	89.7	74.4	01:25	56	2	90.6	76.7	04:53	192	2
harp2	24.1	26.9	24.2	1.8	05:15	97	5	24.2	1.8	15:00	853	6
khb05250	74.3	80.4	74.3	0.0	00:01	17	0	74.3	0.0	00:01	19	0
l152lav	12.9	12.9	12.9	—	15:00	143	0	12.9	—	15:00	557	2
lseu	37.0	37.6	37.3	49.9	00:57	42	4	39.0	349.7	03:33	157	5
markshare1	0.0	0.0	0.0	—	00:02	81	0	0.0	—	00:06	178	1
markshare2	0.0	0.0	0.0	—	00:06	118	0	0.0	—	00:13	247	0
mas74	6.7	6.8	6.7	0.0	00:52	185	0	7.1	327.8	02:20	490	17
misc03	15.0	15.0	15.0	—	02:28	40	0	15.0	—	07:49	128	0
misc06	31.4	38.2	37.6	91.9	00:28	47	5	41.3	145.6	00:37	61	5
misc07	0.7	0.7	0.7	—	07:23	61	0	0.7	—	15:00	163	2

Table 4.2: Strength of 3-row infinity cuts

			Non-Lifted 3-Row					Lifted 3-Row				
Instance	G-GMI	G-EXACT	G-INF	REL (%)	T-TIME (m:s)	C-TIME (ms)	N-CUTS	G-INF	REL (%)	T-TIME (m:s)	C-TIME (ms)	N-CUTS
mitre	84.0	84.0	84.0	—	01:07	27	1	84.0	—	15:00	559	4
mkc	8.7	10.6	8.7	0.0	01:06	89	0	8.7	0.0	15:00	1336	1
mod008	21.6	21.6	21.6	—	00:05	164	0	21.6	—	00:27	779	0
mod010	100.0	100.0	100.0	—	06:30	84	0	100.0	—	15:00	503	0
mod011	33.0	35.1	33.0	0.0	02:50	34	0	33.0	0.0	04:20	52	0
modglob	17.3	33.6	17.5	1.4	00:15	19	5	17.5	1.5	00:24	30	5
nw04	66.1	66.1	66.1	—	15:00	802	0	66.1	—	15:09	13373	0
p0033	34.4	35.2	34.5	8.7	00:11	28	1	35.2	105.4	00:26	65	6
p0201	6.5	6.5	6.5	—	06:51	63	0	6.5	—	15:00	155	4
p0282	3.2	9.9	6.2	45.3	03:31	35	4	6.2	45.3	08:32	84	10
p0548	61.7	74.2	61.7	0.0	00:19	42	0	61.7	0.1	01:15	164	3
p2756	58.0	58.9	58.0	0.5	00:14	30	3	58.0	0.0	00:50	104	1
pk1	0.0	0.0	0.0	—	01:31	163	0	0.0	—	02:48	301	0
pp08a	54.9	66.7	68.1	112.4	00:05	18	37	68.1	112.3	00:07	22	36
pp08aCUTS	33.8	40.7	38.0	60.8	04:35	62	23	38.0	61.3	06:29	88	21
qiu	1.4	1.7	1.7	104.5	08:39	110	11	1.7	102.6	10:21	132	13
qnet1	11.7	16.8	13.8	41.0	15:00	121	13	14.7	59.9	14:59	250	13
qnet1_o	19.8	30.5	23.8	36.9	12:46	133	27	24.8	45.9	15:00	651	32
rentacar	19.8	20.3	19.8	0.0	00:02	21	0	19.8	0.0	00:03	23	0
rgn	12.0	12.0	12.0	—	00:15	45	0	16.2	—	00:29	88	7
rout	0.3	5.2	0.3	0.0	12:56	153	0	0.3	0.0	15:00	460	3
set1ch	29.4	72.2	44.0	34.1	00:08	19	87	44.0	34.1	00:14	33	87
seymour	6.6	6.6	6.6	—	14:59	108	0	6.6	—	14:59	209	0
stein27	0.0	0.0	0.0	—	01:27	15	0	0.0	—	02:08	22	0
stein45	0.0	0.0	0.0	—	07:38	46	0	0.0	—	13:07	79	0
swath	8.7	10.7	8.7	0.1	13:41	124	7	10.6	93.3	14:59	837	5
vpm1	29.9	33.6	30.9	25.4	00:04	27	8	33.9	105.8	00:09	58	8
vpm2	13.9	23.5	15.7	19.5	00:14	33	8	19.0	53.6	00:22	54	26
	29.0	36.1	32.2	44.8	03:52	263.8	7.8	32.7	52.2	06:06	687.2	10.0

Table 4.2: Strength of 3-row infinity cuts (continued)

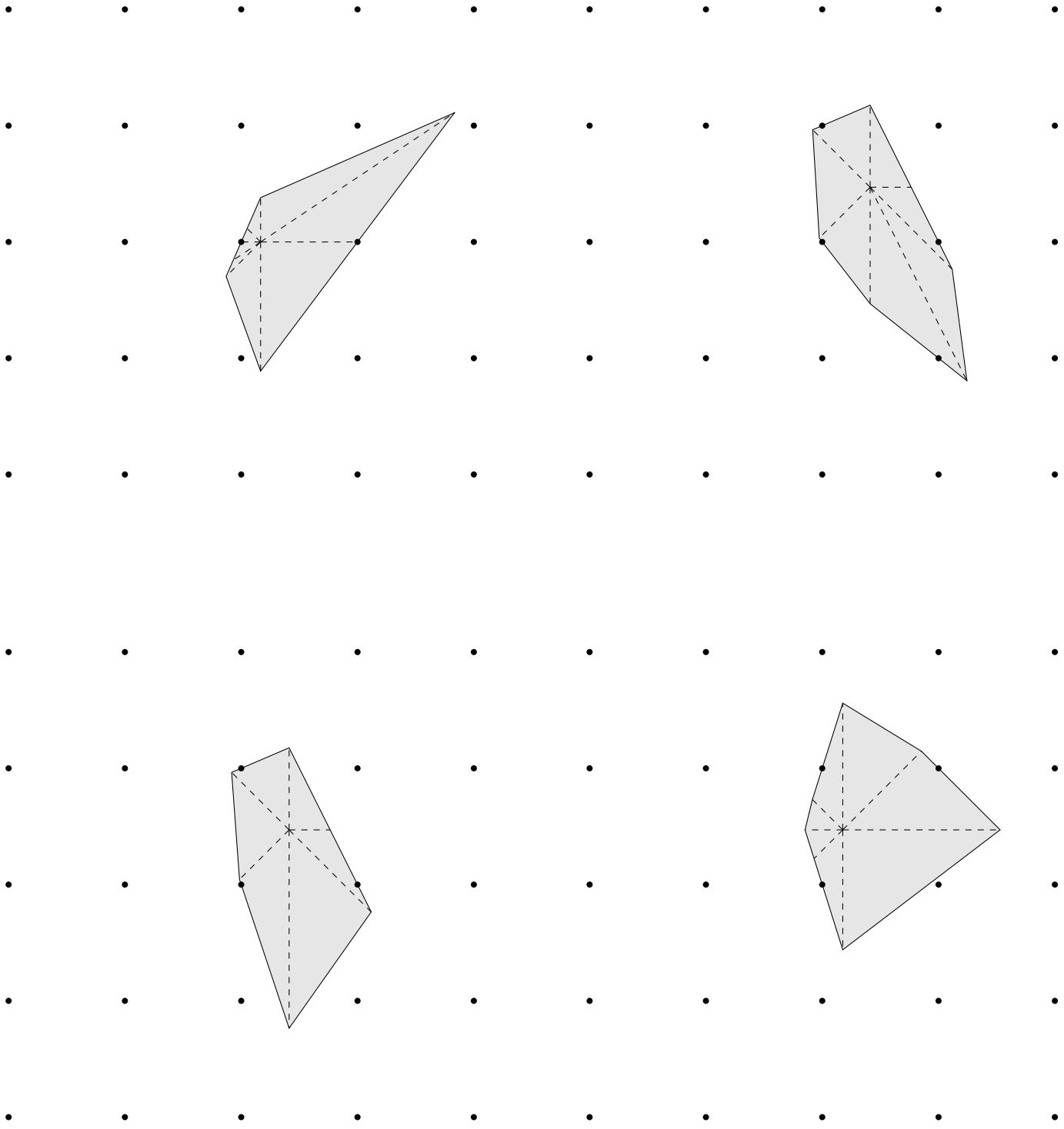


Figure 4.5: Infinity cuts generated from instance `gen`

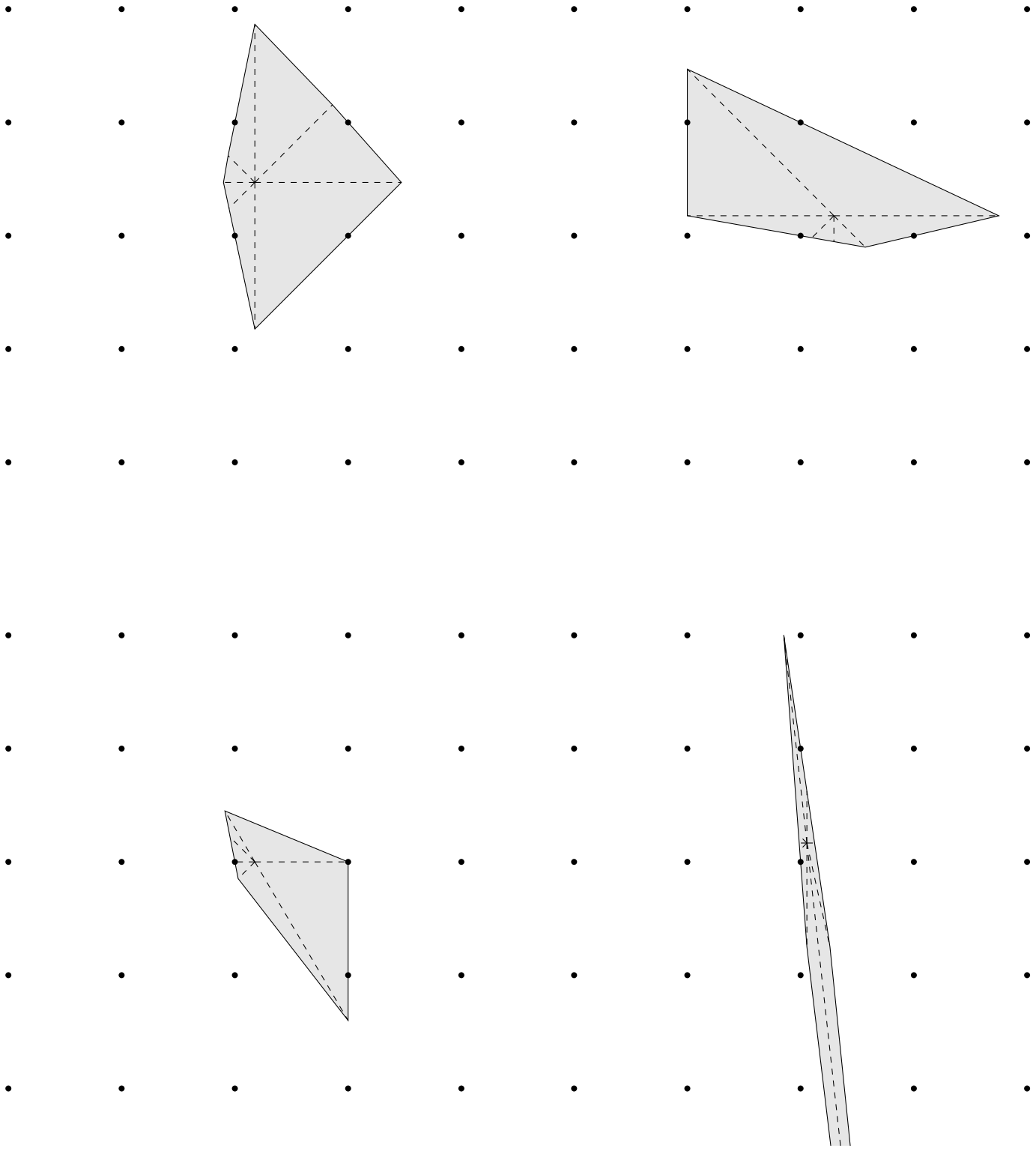


Figure 4.5: Infinity cuts generated from instance `gen` (continued)

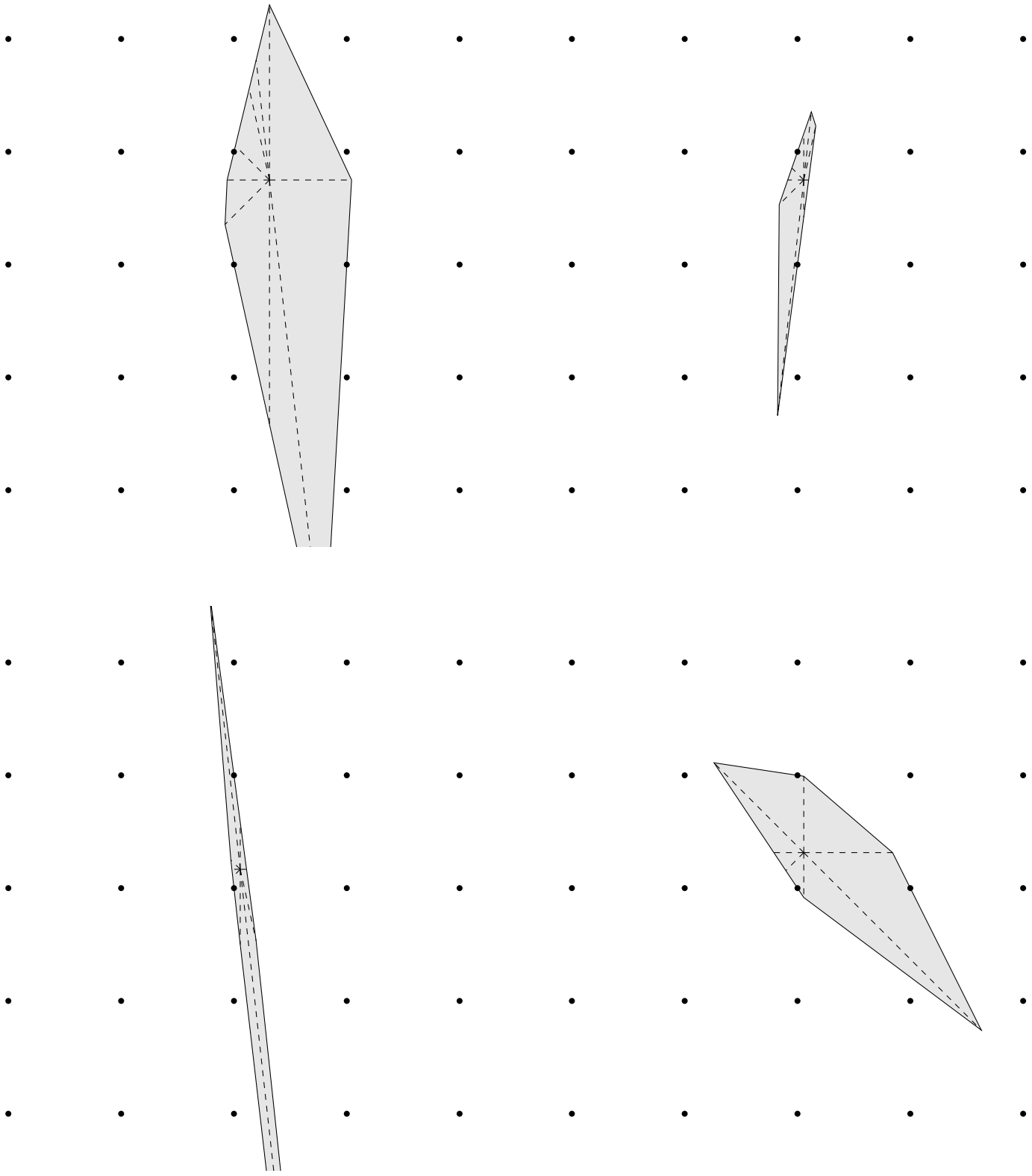


Figure 4.5: Infinity cuts generated from instance `gen` (continued)

Chapter 5

Intersection Cuts for Single-Row Relaxations

In the previous chapters, we used the framework of intersection cuts and trivial lifting to produce valid inequalities for relaxations of the simplex tableau containing two or more rows. These techniques, however, can also be used to produce strong inequalities for single-row relaxations. In this chapter, we revisit the single-row mixed-integer relaxation of the simplex tableau, and we use the trivial lifting approach to produce inequalities that are not dominated by Gomory Mixed-Integer (GMI) cuts.

Consider the mixed-integer single-row relaxation of the simplex tableau, given by

$$\begin{aligned} x &= \phi + \sum_{j=1}^{m} \omega^j s_j + \sum_{j=1}^{p} \rho^j z_j \\ x &\in \mathbb{Z} \\ s &\in \mathbb{R}^m_+ \\ z &\in \mathbb{Z}^p_+ \end{aligned} \tag{M1}$$

where $\phi \in \mathbb{Q} \setminus \mathbb{Z}$ and $\rho^1, \ldots, \rho^m, \omega^1, \ldots, \omega^p \in \mathbb{Q} \setminus \{0\}$. We use Greek letters here, instead of the usual notation, to stress the fact these variables are scalars. Although, in previous chapters, we used the trivial lifting approach on relaxations with two or more rows, this approach can also be applied here. Indeed, when applied to (M1), this approach yields Gomory Mixed-Integer cuts, a very important class of single-row cuts that is widely used in practice, as we demonstrate next. As we recall, the first step obtain a strong inequality of (M1) by the trivial lifting approach, is to construct a maximal lattice-free $B \subseteq \mathbb{R}$ containing ϕ in its interior. In this case, there is only one choice for B, namely

$$B = \{x \in \mathbb{R} : \lfloor \phi \rfloor \leq x \leq \lceil \phi \rceil\}.$$

The gauge function ψ of $B - \phi$, and that the trivial lifting π of ψ, are given by

$$\psi(\omega) = \max\left\{\frac{-\omega}{\widehat{\phi}}, \frac{\omega}{1-\widehat{\phi}}\right\}, \qquad \pi(\rho) = \min\left\{\frac{\rho - \lfloor\rho\rfloor}{1-\widehat{\phi}}, \frac{1 + \lfloor\rho\rfloor - \rho}{\widehat{\phi}}\right\},$$

respectively, where $\widehat{\phi}$ is the fractional part of ϕ. Plugging these functions into

$$\sum_{j=1}^{m} \psi(\omega^j) s_j + \sum_{j=1}^{p} \pi(\rho^j) z_j \geq 1$$

we obtain a valid inequality for (M1). It can be verified that this inequality is precisely the GMI cut for this relaxation. This fact was already observed by Dey and Wolsey [40], and it was the main motivation for the introduction of the lifting approach. Since it yields such an important class of inequalities for single-row relaxations, the authors reasoned that this approach had the potential to generate an equally important class of cuts for relaxations with two or more rows.

Despite their usefulness, we recall that GMI cuts are not the only inequalities that induce facets of the convex hull of (M1). Since the lifting approach starts by first discarding the integral non-basic variables z, the only inequalities obtained in this way are those where the continuous non-basic variables receive the strongest possible coefficients. Many other classes of single-row cuts described in the literature, including, for example, two-step MIR cuts [33] and knapsack cuts [44], are not dominated by GMI cuts. What these inequalities have in common is that they exploit the integrality of the non-basic variables from the beginning.

In this chapter, we revisit the single-row mixed-integer relaxation (M1), with the goal of generating additional valid inequalities for (M1); in particular, we are interested in cuts that are not dominated by GMI cuts. Although single-row relaxations have been extensively studied in the literature, the novelty in our approach is that we use the framework of multi-row intersection cuts and trivial lifting to derive our cuts. By doing so, we obtain a geometric interpretation of our inequalities, and of other inequalities in the literature, such as two-step MIR cuts. Moreover, as is central in this thesis, the tools we develop are particularly well-suited for a practical implementation, and we present computational results using our cuts on real-world instances.

The inequalities we consider are obtained by preserving the integrality of a single non-basic variable. More specifically, we first rewrite (M1) as a mixed-integer two-row

relaxation, given by

$$\begin{aligned} &\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \phi \\ 0 \end{pmatrix} + \sum_{j=1}^{m} \begin{pmatrix} \omega^j \\ 0 \end{pmatrix} s_j + \begin{pmatrix} \rho_1 \\ 1 \end{pmatrix} z_1 + \sum_{j=2}^{p} \begin{pmatrix} \rho^j \\ 0 \end{pmatrix} z_j \\ &x \in \mathbb{Z}^2 \\ &s \in \mathbb{R}^m_+, z_1 \in \mathbb{R}_+ \\ &z_2, \dots, z_p \in \mathbb{Z}_+ \end{aligned} \tag{M2}$$

Note that (M2) was obtained from (M1) by adding a new variable $x_2 \in \mathbb{Z}$ and a new row $x_2 = z_1$. In this model, the integrality of the non-basic variable z_1 can be discarded, since it is already implied by the integrality of the basic variable x_2. The idea of transferring complex constraints on the non-basic variables to the basic variables by adding rows was first suggested by Conforti, Cornuéjols and Zambelli [28]. In order to obtain a strong valid inequality for (M2), we apply the trivial lifting approach. First, we fix the variables $z_2, \dots, z_p$ to zero, to obtain the continuous two-row relaxation

$$\begin{aligned} &\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \phi \\ 0 \end{pmatrix} + \sum_{j=1}^{m} \begin{pmatrix} \omega^j \\ 0 \end{pmatrix} s_j + \begin{pmatrix} \rho_1 \\ 1 \end{pmatrix} z_1 \\ &x \in \mathbb{Z}^2 \\ &s \in \mathbb{R}^m_+, z_1 \in \mathbb{R}_+ \end{aligned} \tag{C2}$$

Then, we find a maximal lattice-free set in $\mathbb{R}^2$ containing $\left(\begin{smallmatrix} \phi \\ 0 \end{smallmatrix}\right)$ in its interior, which will give us the cut coefficients for the variables $s_1, \dots, s_m, z_1$. Finally we perform the trivial lifting, in order to find the cut coefficients for the variables $z_2, \dots, z_p$.

As in the previous chapters, in order to generate these inequalities in practice, two questions must be answered. First, how to choose the lattice-free sets that will generate a valid inequality for (C2); and second, once this valid inequality is generated, how to perform the trivial lifting. For the second question, the results from Chapter 3 can be used, since this is a continuous two-row relaxation. In order to answer the first question, we study, in Section 5.1, the facial structure of the convex hull of (C2). Exploiting the fact that this two-row relaxation has a very special structure, we are able to find a precise description of the lattice-free sets that induce facets of its convex hull, and in Section 5.2, we present a geometric algorithm to enumerate them. This algorithm also leads to a natural upper bound on the split rank of (C2), which we prove in Section 5.3. Finally, in Section 5.4, we run computational experiments to compare the strength of the cuts developed here against GMI cuts alone. Our results indicate that, for some

instances, we close significantly more gap than GMI with our single-row cuts.

5.1 Basic results

In this section, we study the structure of the convex hull of (C2). Our goal is to obtain the complete list of lattice-free sets that induce a facet-defining inequality for this set.

First, observe that (C2) can be simplified by aggregating the continuous non-basic variables s_j according to the sign of ω_j. Indeed, the structure of (C2) is essentially the same as of the set

$$P_I = \left\{ (x, s) \in S \times \mathbb{R}^3_+ : \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \phi \\ 0 \end{pmatrix} + \begin{pmatrix} \rho \\ 1 \end{pmatrix} s_1 + \begin{pmatrix} 1 \\ 0 \end{pmatrix} s_2 + \begin{pmatrix} -1 \\ 0 \end{pmatrix} s_3 \right\},$$

where $\phi \in \mathbb{Q} \setminus \mathbb{Z}, \rho \in \mathbb{Q} \setminus \{0\}$ and where we let $S := (\mathbb{Z} \times \mathbb{Z}_+)$. Note that we use $S = (\mathbb{Z} \times \mathbb{Z}_+)$ to emphasize that x_2 is non-negative, although $S = \mathbb{Z}^2$ would yield the same set since $x_2 = s_1$ and $s_1 \geq 0$. It can also be shown that P_I is closely related to the set of solutions of a mixed-integer knapsack problem having two integral variables and one continuous variable, which has been studied before. Hirschberg and Wong [54] developed a polynomial-time algorithm to optimize over pure integer knapsack problems with two variables. Agra and Constantino [1, 2] provided a complete characterization of its convex hull, and a polynomial-time method exploiting the approach in [54] to enumerate its facet-defining inequalities. Similar results are also due to Atamtürk and Rajan [8]. The particularity of our approach is that we use the framework of multi-row intersection cuts [12] to derive inequalities for $\text{conv}(P_I)$.

Let $f = \left(\begin{smallmatrix} \phi \\ 0 \end{smallmatrix}\right)$, $r^1 := \left(\begin{smallmatrix} \rho \\ 1 \end{smallmatrix}\right)$, $r^2 := \left(\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right)$, $r^3 := \left(\begin{smallmatrix} -1 \\ 0 \end{smallmatrix}\right)$ and $R := [r^1|r^2|r^3]$, i.e.,

$$P_I = \left\{ (x, s) \in S \times \mathbb{R}^3_+ : x = f + Rs \right\},$$

Our definition of $\text{conv}(P_I)$ is a special case of the set with the same name in [5], and the following properties carry over from [5]:

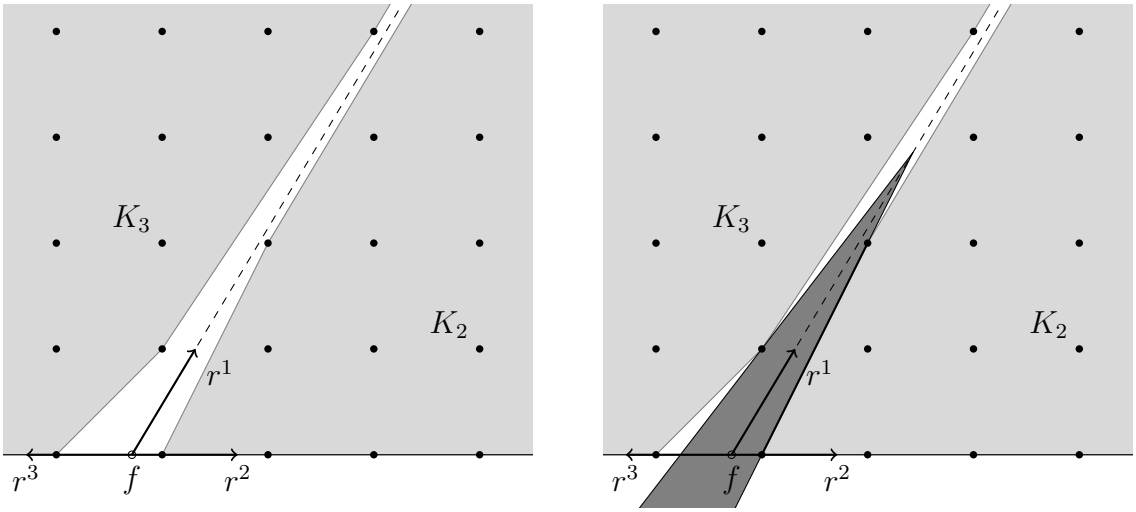
Proposition 41. *[5]*

(i) The dimension of $\text{conv}(P_I)$ *is three.*

(ii) The extreme rays of $\text{conv}(P_I)$ *are* $(\rho, 1, 1, 0, 0)$, $(1, 0, 0, 1, 0)$ *and* $(-1, 0, 0, 0, 1)$.

Closely related to the structure of $\text{conv}(P_I)$ are the two knapsack sets

$$K_j = \text{conv}\left(\mathbb{Z}^2 \cap \left(f + \text{cone}(r^1, r^j)\right)\right) \qquad \text{for } j \in \{2, 3\},$$



(a) Knapsack sets K_2 and K_3.

(b) Example of facet-defining wedge.

Figure 5.1: Knapsack sets and facet-defining S-free sets.

illustrated in Figure 5.1a. For the vertices of $\text{conv}(P_I)$, we can refine the characterization from [5]:

Proposition 42. *A point $(\bar{x}, \bar{s}) \in P_I$ is a vertex of $\text{conv}(P_I)$ if and only if $\bar{s} = \bar{s}_1 e_1 + \bar{s}_j e_j$ for some $j \in \{2, 3\}$ and $\bar{x}$ is a vertex of $\text{conv}(K_j)$.*

Proof. ($\Rightarrow$) Assume that $(\bar{x}, \bar{s})$ is a vertex of $\text{conv}(P_I)$. Then, $\bar{x}$ is integer and $\bar{s}$ is a vertex of $P_I \cap \{(x, s) : x = \bar{x}\}$, hence a basic feasible solution to the system $\{s \in \mathbb{R}^3_+ : Rs = \bar{x} - f\}$. Thus, $\bar{s}$ has at most two nonzero components. Furthermore, since the submatrix $[r^2|r^3]$ is not invertible, either s_2 or s_3 is nonbasic, hence zero. Therefore, $\bar{s} = \bar{s}_1 e_1 + \bar{s}_j e_j$ for some $j \in \{2, 3\}$. Since $\bar{x}$ is integer, this implies that $\bar{x} \in K_j$. We next show that $\bar{x}$ is a vertex of $\text{conv}(K_j)$. In the following, we assume $j = 2$, since the other case is similar. Suppose, by contradiction, that $\bar{x}$ is not a vertex of $\text{conv}(K_2)$. Then, there must exist $x^1, \dots, x^k \in K_2 \cap \mathbb{Z}^2$ distinct from $\bar{x}$ and $\lambda \in \mathbb{R}^k_+$ such that $\bar{x} = \sum_{i=1}^k \lambda_i x^i$ and $\sum_{i=1}^k \lambda_i = 1$. Let $M = [r^1|r^2]$. Note that since $\text{lin}(r^1) \neq \text{lin}(r^2)$, M is invertible. For each $i \in \{1, \dots, k\}$, let $s^i \in \mathbb{R}^3_+$ be such that $s^i = s^i_1 e_1 + s^i_2 e_2$ and

$$\begin{pmatrix} s^i_1 \\ s^i_2 \end{pmatrix} = M^{-1}(x^i - f).$$

For every $i \in \{1, \dots, k\}$, $s^i_1, s^i_2 \geq 0$ because $x^i \in K_2$, so $(x^i, s^i) \in P_I$. Furthermore, by linearity, $\bar{s} = \sum_{i=1}^k \lambda_i s^i$, thus $(\bar{x}, \bar{s}) = \sum_{i=1}^k \lambda_i (x^i, s^i)$. This contradicts the assumption that $(\bar{x}, \bar{s})$ is a vertex of $\text{conv}(P_I)$.

($\Leftarrow$) Let $(\bar{x}, \bar{s}) \in P_I$ be such that $\bar{s} = \bar{s}_1 e_1 + \bar{s}_j e_j$ for some $j \in \{2, 3\}$ and $\bar{x}$ is a vertex of $\text{conv}(K_j)$. In the following, we assume $j = 2$, since the other case is similar. We

prove that $(\bar{x}, \bar{s})$ is a vertex of $\text{conv}(P_I)$. Suppose, by contradiction, that this is not the case. Then, there must exist k points $(x^1, s^1), \ldots, (x^k, s^k) \in P_I$ distinct from $(\bar{x}, \bar{s})$ and $\lambda \in \mathbb{R}^k_+$ such that $(\bar{x}, \bar{s}) = \sum_{i=1}^k \lambda_i (x^i, s^i)$ and $\sum_{i=1}^k \lambda_i = 1$. Since $\bar{s}_3 = 0$ and $\lambda \geq 0$, we have $s^i_3 = 0$ for all i. Therefore $x^i \in K_2$ for all i and these points are all distinct from $\bar{x}$. We can construct $\bar{x}$ as a convex combination of k points $x^1, \ldots, x^k \in K_2$ distinct from $\bar{x}$. This contradicts the assumption that $\bar{x}$ is a vertex of $\text{conv}(K_2)$. ■

We now look at the facet-defining inequalities for $\text{conv}(P_I)$.

Proposition 43. *[5] The facet-defining inequalities of* $\text{conv}(P_I)$ *take the form*

(i) $s_j \geq 0$ *for* $j \in \{1, 2, 3\}$,

(ii) $\alpha^T s \geq 1$ *for some* $\alpha \geq 0$.

Note that inequalities of the form (i) in Proposition 43, i.e. $s_j \geq 0$ for some $j \in \{1, 2, 3\}$, are called *trivial*, while those of the form (ii) are called *nontrivial*. For the nontrivial inequalities, we have the following further characterization.

Proposition 44. *Every nontrivial facet-defining inequality* $\alpha^T s \geq 1$ *of* $\text{conv}(P_I)$ *satisfies* $\alpha_2 > 0$ *and* $\alpha_3 > 0$. *If* $\alpha_1 = 0$, *then there are no integer points on the ray* $f + \text{cone}(r^1)$, *and there is only one facet-defining inequality of that form.*

Proof. Let $z^2 := (\lceil \phi \rceil, 0, 0, \lceil \phi \rceil - \phi, 0)$ and $z^3 := (\lfloor \phi \rfloor, 0, 0, 0, \phi - \lfloor \phi \rfloor)$. Since z^2 and z^3 belong to P_I, we must have $\alpha_2 > 0$ and $\alpha_3 > 0$, respectively. Suppose $f + \lambda r^1 = \bar{x} \in \mathbb{Z}^2$ for some $\lambda \in \mathbb{R}_+$. Since $\phi \notin \mathbb{Z}$ we have $\lambda > 0$. Then $(\bar{x}_1, \bar{x}_2, \lambda, 0, 0) \in P_I$, and therefore $\alpha_1 > 0$. It follows that if $\alpha_1 = 0$, then $f + \lambda r^1 = \bar{x} \in \mathbb{Z}^2$ does not exist. Finally, we show uniqueness for a facet-defining inequality with $\alpha_1 = 0$. Suppose that $\alpha_2 s_2 + \alpha_3 s_3 \geq 1$ and $\alpha'_2 s_2 + \alpha'_3 s_3 \geq 1$ are facet-defining for $\text{conv}(P_I)$. Consider the vertices of $\text{conv}(P_I)$ that are tight on $\alpha_2 s_2 + \alpha_3 s_3 \geq 1$. By Proposition 42, they all have $s_h = 0$ for some $h \in \{2, 3\}$. However, the value of h is not the same for all of them, otherwise we could set $\alpha_h = 0$ and the resulting inequality would cut off z^h. Let $(\bar{x}, \bar{s})$ be one such vertex and let $\{j\} := \{1, 2\} \setminus \{h\}$. Since $\alpha'_2 s_2 + \alpha'_3 s_3 \geq 1$ is valid, $\alpha'_j \geq \alpha_j$. By applying the process to all vertices, then repeating for those that are tight on $\alpha'_2 s_2 + \alpha'_3 s_3 \geq 1$, we obtain $\alpha'_2 = \alpha_2$ and $\alpha'_3 = \alpha_3$. ■

Our motivation for studying a model of the form of P_I is that such model is an ideal setting for computing and using intersection cuts [12]. Specifically, every nontrivial valid inequality for P_I is an intersection cut from some S-free set in $\mathbb{R}^2$ [38]. A convex set $B \subseteq \mathbb{R}^m$ is *S-free* if its interior contains f but no point of S. The set is *maximal* if it is not

properly contained into any other S-free set. Maximal sets are the only ones that interest us, since any non-dominated inequality can be obtained from such sets. Note that Basu, Conforti, Cornuéjols and Zambelli [21] proved that every maximal S-free set is polyhedral, and given a polyhedral S-free set $B := \left\{x \in \mathbb{R}^m \ : \ g_i^T(x - f) \leq 1, \ i = 1, \ldots, k\right\}$, the intersection cut coefficient for s_j is given by $\psi_B(r^j) = \max_{i=1,\ldots,k} g_i^T r^j$ [38]. In the context of $\text{conv}(P_I)$, $x \in S = \mathbb{Z} \times \mathbb{Z}_+$ and $s \in \mathbb{R}^3_+$. Proposition 45 shows that in this case, we may restrict our attention to S-free sets B with two faces, i.e. $k = 2$. An analogous result was obtained in [28] for an infinite relaxation of P_I.

Proposition 45. *If $\alpha^T x \geq 1$ is a nontrivial valid inequality for P_I, then there exists an S-free set*

$$B = \left\{x \in \mathbb{R}^2 \ : \ g_1^T(x - f) \leq 1, \ g_2^T(x - f) \leq 1\right\}$$

such that $\alpha^T x \geq 1$ is the intersection cut computed from B.

Proposition 45 has a very simple justification: Only the intersections (if any) of the facets of B with the line $\text{lin}(r^j)$ affect the intersection cut coefficient α_j. Therefore, for a given cut $\alpha \in \mathbb{R}^3_+$, and one can easily construct a wedge or a split in $\mathbb{R}^2$ that provides the three desired intersections. It implies that all facet-defining inequalities for $\text{conv}(P_I)$ can be obtained from maximal S-free splits unbounded along the line $f + \text{lin}(r^1)$ and maximal S-free wedges with vertex on that same line. As this reasoning relies on a geometric intuition for intersection cuts, we also provide a formal proof.

Proof of Proposition 45. The proof is constructive. Let $\alpha^T x \geq 1$ be a nontrivial valid inequality for P_I. By Proposition 43, $\alpha \geq 0$, and by Proposition 44, $\alpha_2, \alpha_3 > 0$. We let $g_1 := (\alpha_2, \ \alpha_1 - \rho\alpha_2)$ and $g_2 := (-\alpha_3, \ \alpha_1 + \rho\alpha_3)$. It is straightforward to verify that B then yields the appropriate intersection cut coefficients. Suppose that B is not S-free. Then, there exists $\bar{x} \in S$ such that $g_1^T(x - f) < 1$ and $g_2^T(x - f) < 1$. We construct $\bar{s}$ such that $(\bar{x}, \bar{s}) \in P_I$. By substituting $x - f = Rs$ in the two above inequalities, we obtain $\alpha_1\bar{s}_1 + \bar{s}_2\alpha_2 - \bar{s}_3\alpha_2 < 1$ and $\alpha_1\bar{s}_1 - \bar{s}_2\alpha_3 + \bar{s}_3\alpha_3 < 1$, respectively. We can assume without loss of generality that either $\bar{s}_2 = 0$ or $\bar{s}_3 = 0$. In each case, one of the latter inequalities yields $\alpha^T\bar{s} < 1$, which contradicts the validity of $\alpha^T x \geq 1$ for P_I. ■

An interesting feature of the set B constructed above is that a vertex $(\bar{x}, \bar{s})$ of P_I is tight on $\alpha^T s \geq 1$ if and only if $\bar{x}$ is on the boundary of B. Indeed, the latter implies either $g_1^T(x - f) = 1$ (if $\bar{s}_3 = 0$), or $g_2^T(x - f) = 1$ (if $\bar{s}_2 = 0$). Again, substituting $x - f = Rs$ yields $\alpha^T\bar{s} = 1$ in both cases.

We now prove that we can restrict our attention even further, to a specific finite family of splits and wedges. This will let us develop an algorithm to enumerate all

these relevant S-free sets in Section 5.2. Proposition 44 states that if $\alpha^T s \geq 1$ is facet-defining for $\text{conv}(P_I)$, then $\alpha_2, \alpha_3 > 0$. If $\alpha_1 = 0$, then there is exactly one facet-defining inequality of that form. The proof of Proposition 45 gives us the split set $B = \{x \in \mathbb{R}^2 : \frac{1}{\alpha_2} \leq \left(\begin{smallmatrix} -1 \\ \rho \end{smallmatrix}\right)(x - f) \leq \frac{1}{\alpha_3}\}$[1]. Otherwise, $\alpha > 0$ and B is a wedge with its apex on the line $f + \text{lin}(r^1)$. Then, Theorem 46 gives a useful characterization of the corresponding facet-defining inequalities.

Theorem 46. *(i) A valid inequality $\alpha^T s \geq 1$ where $\alpha > 0$ is facet-defining for $\text{conv}(P_I)$ if and only if it is tight at three distinct vertices of $\text{conv}(P_I)$. (ii) Furthermore, at least one of those three vertices corresponds to a vertex of $\text{conv}(K_2)$, and at least one corresponds to a vertex of $\text{conv}(K_3)$.*

Proof. Let $P_s := \text{proj}_s \text{conv}(P_I)$ be the projection of $\text{conv}(P_I)$ on the space of the s variables. (i) $\Leftarrow$: Since $\dim(P_s) = 3$, a valid inequality that is tight at three affinely independent points is facet-defining. (i) $\Rightarrow$: Since $\dim(P_s) = 3$, a facet of P_s may contain fewer than three vertices of P_s only if its affine hull contains an extreme ray of P_s. Assume that $\alpha^T s \geq 1$ is a corresponding facet-defining inequality that is tight at $\bar{s} \in P_s$, i.e. $\alpha^T \bar{s} = 1$. Then, $\alpha^T(\bar{s} + e_j) = 1$ for some $j \in \{1, 2, 3\}$, implying that $\alpha_j = 0$. This contradicts $\alpha > 0$. (ii): Assume that three tight vertices $(x^1, s^1), (x^2, s^2), (x^3, s^3)$ of $\text{conv}(P_I)$ correspond to three vertices x^1, x^2, x^3 of $\text{conv}(K_j)$, for a single fixed $j \in \{2, 3\}$. Let $\{h\} = \{2, 3\} \setminus \{j\}$. Then, $s^1_h = s^2_h = s^3_h = 0$. The facet-defining inequality of $\text{conv}(P_I)$ that is tight at these three vertices is $s_h \geq 0$ (Proposition 43), contradicting $\alpha > 0$. ■

Theorem 46 means that in order to obtain facet-defining intersection cuts for P_I, one should focus on S-free sets that have at least three S points on their boundary: at least one of each of K_2 and K_3. This means that each of those S-free sets is tight at two points of either K_2 or K_3. In other words, one of its facets coincides with a facet of either $\text{conv}(K_2)$ or $\text{conv}(K_3)$. See Figure 5.1b. An analogous result is well-known in the case of an infinite relaxation of P_I [38, 28].

5.2 Enumerating the vertices of the knapsacks

In this section we describe a simple algorithm for enumerating the vertices of the two knapsack sets K_2 and K_3 described in Section 5.1, allowing us to enumerate all the splits

[1] If f and ρ are rational numbers, we can compute geometrically a maximal lattice-free set of that form. Specifically, letting $d \in \mathbb{Z}$ such that $fd \in \mathbb{Z}$ and $\rho d \in \mathbb{Z}$, $g = \gcd(d, \rho d)$ and $v = \frac{fd}{g} - \left\lfloor \frac{fd}{g} \right\rfloor$, we get the cut $\frac{g}{d(1-v)} s_2 + \frac{g}{dv} s_3 \geq 1$, provided that $\frac{fd}{g} \notin \mathbb{Z}$.

and wedges that induce facets of $\text{conv}(P_I)$.

Since we have a complete description of the extreme points and rays of $\text{conv}(P_I)$, its facet-defining inequalities could be obtained by enumerating the vertices of its polar, as shown by Andersen, Louveaux, Weismantel and Wolsey [5, 4] in dimension two, and Basu, Hildebrand and Köeppe [23] in general dimensions. Although this approach has been performed [61] it has two drawbacks: Even separation in two dimensions relies on optimizing over a cut-generating linear program (CGLP) with the simplex method, which adds a source of numerical inaccuracies. Then, finding *all* facet-defining inequalities would require enumerating the vertices of this CGLP, a difficult computational task. Here, instead, we exploit the characterization provided by Theorem 46 to enumerate the facet-defining inequalities of $\text{conv}(P_I)$.

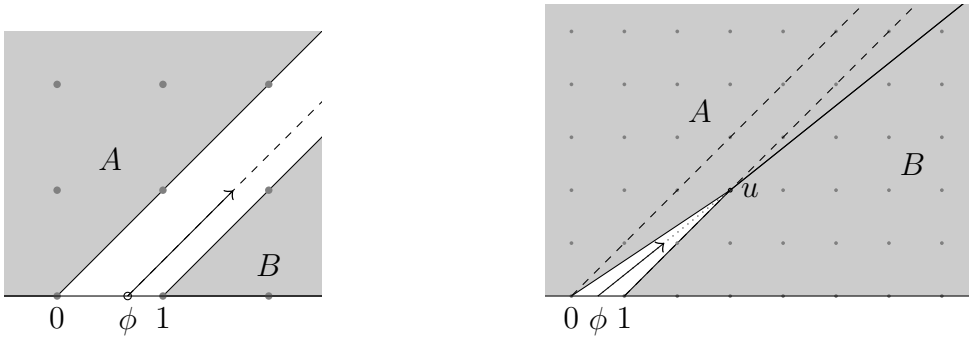
Enumerating the vertices of the knapsack sets K_2 and K_3 is a particular case of the integer hull problem. Harvey [53] devised an algorithm for enumerating the facets of the integer hull of an arbitrary two-dimensional polyhedron. The complexity of the algorithm is $O(n \log A_{\max})$ where n is the number of input inequalities and $A_{\max}$ is the magnitude of the largest input coefficient. This algorithm is optimal in the sense that no better asymptotic bound is possible for the problem. In the more specific case of a two-dimensional knapsack set, Agra and Constantino [1, 2] and Atamtürk and Rajan [8] independently gave polynomial-time algorithms. Both are based on the two-dimensional knapsack optimization algorithm of Hirschberg and Wong [54].

Despite the abundant earlier work on the topic, we develop a different method for computing the vertices of the integer hull of a knapsack, with the following motivation. First, our method has a simple geometric interpretation that allows us to prove an upper bound on the split rank of $\text{conv}(P_I)$ (Section 5.3). Secondly, it is easy to implement and yields a very fast code, which we use in our computations (Section 5.4).

Consider the two sets

$$A = \text{conv}\left(\mathbb{Z}^2 \cap \left(\binom{\phi}{0} + \text{cone}\{\binom{\rho}{1}, \binom{-1}{0}\}\right)\right),$$
$$B = \text{conv}\left(\mathbb{Z}^2 \cap \left(\binom{\phi}{0} + \text{cone}\{\binom{\rho}{1}, \binom{1}{0}\}\right)\right).$$

Observe that A is simply $\text{conv}(K_3)$ and B is $\text{conv}(K_2)$. Our goal is to obtain the set of vertices of A and of B. For simplicity, we assume $0 < \phi < 1$. If that is not the case, A and B can be translated along the x_1 axis to enforce the assumption; the resulting vertices can then be translated back to obtain those of the original sets. An alternative



(a) $\phi = \widehat{\phi + \rho}$

(b) $\phi \neq \widehat{\phi + \rho}$ and $u \in \mathbb{Z}^2$ ($\mathcal{S}$ shown in dashed lines)

Figure 5.2: Illustration of Propositions 47 and 48.

definition of A and B, then, is the following:

$$A = \operatorname{conv}\left\{x \in \mathbb{Z}^2 : x_1 - \rho x_2 \leq \phi, x_2 \geq 0\right\},$$
$$B = \operatorname{conv}\left\{x \in \mathbb{Z}^2 : x_1 - \rho x_2 \geq \phi, x_2 \geq 0\right\}.$$

Note that $(0,0)$ and $(1,0)$ are always vertices of A and B, respectively. The next proposition shows that, in some cases, these are the only vertices of these two sets (Figure 5.2a). We recall that $\widehat{\phi + \rho}$ is the fractional part of $\phi + \rho$.

Proposition 47. *If* $\phi = \widehat{\phi + \rho}$, *then* $\operatorname{vert}(A) = \{\binom{0}{0}\}$ *and* $\operatorname{vert}(B) = \{\binom{1}{0}\}$.

Proof. First, note that the condition on ϕ implies that $\rho \in \mathbb{Z}$. We can, therefore, round down the right-hand side of one of the inequalities that define A, to obtain

$$A = \operatorname{conv}\left\{x \in \mathbb{Z}^2 : x_1 - \rho x_2 \leq 0, x_2 \geq 0\right\}.$$

Clearly, (0,0) is the only vertex of the linear relaxation of this set. Since the vertex is integral, then the linear relaxation coincides with its integer hull. We conclude that $(0,0)$ is the only vertex of A. To prove that $(1,0)$ is the only vertex of B, we proceed similarly. ■

Now, we focus on the case where $\phi \neq \widehat{\phi + \rho}$, and the previous proposition does not apply. It turns out that the vertices of A and B are related to the lattice-free split $\mathcal{S}$, given by

$$\mathcal{S} = \left\{x \in \mathbb{R}^2 : 0 \leq x_1 - \lfloor \phi + \rho \rfloor x_2 \leq 1\right\}.$$

Let $u \in \mathbb{R}^2$ be the point where the ray $\binom{\phi}{0} + \operatorname{cone}\binom{\rho}{1}$ meets the split. The next

proposition shows that, when u is an integral point, the vertices of A and B can be also easily determined (see Figure 5.2b).

Proposition 48. *If* $u \in \mathbb{Z}^2$ *then* $\text{vert}(A) = \{\binom{0}{0}, u\}$ *and* $\text{vert}(B) = \{\binom{1}{0}, u\}$.

Proof. First, we prove that $u_2x_1 - u_1x_2 \leq 0$ is a valid inequality for A. We assume that the ray hits the boundary of the split on the "B-side", i.e. on the line $x_1 - \lfloor \phi + \rho \rfloor x_2 = 1$. The other case is analogous. Let $x \in A \cap \mathbb{Z}^2$. Since x is not in the interior of the split, it must satisfy either $x_1 - \lfloor \rho + \phi \rfloor x_2 \leq 0$ or $x_1 - \lfloor \rho + \phi \rfloor x_2 \geq 1$. We prove that, in either case, $u_2x_1 - u_1x_2 \leq 0$.

First, suppose $x_1 - \lfloor \phi + \rho \rfloor x_2 \leq 0$. Since $u_2 \geq 0$, we can multiply both sides of this inequality by u_2 to obtain $u_2x_1 - \lfloor \phi + \rho \rfloor u_2x_2 \leq 0$. Also, since u is on the B-side boundary of the split, then $u_1 - \lfloor \phi + \rho \rfloor u_2 = 1$. Therefore, $-(u_1 - \lfloor \phi + \rho \rfloor u_2)x_2 \leq 0$. Summing the two previous inequalities, we obtain $u_2x_1 - u_1x_2 \leq 0$, as desired.

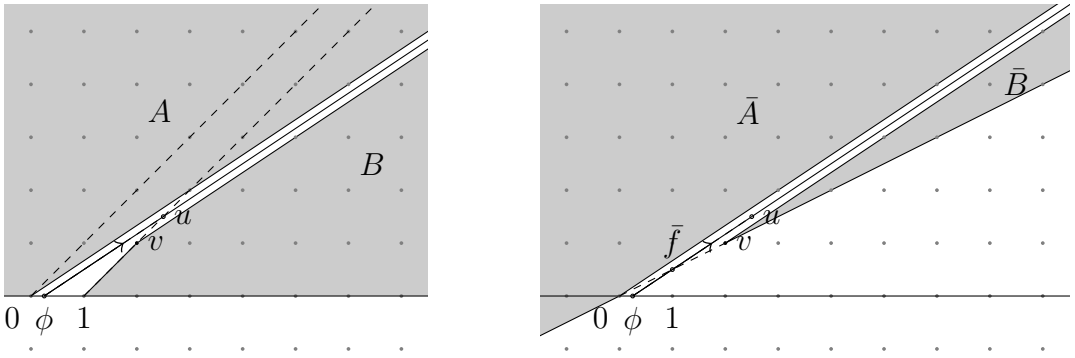
Now suppose $x_1 - \lfloor \phi + \rho \rfloor x_2 \geq 1$. Since u satisfies $u_1 - \rho u_2 = \phi$ and $u_1 - \lfloor \phi + \rho \rfloor u_2 = 1$, then we must have $u_1 = \frac{\rho - \phi\lfloor \phi + \rho \rfloor}{\rho - \lfloor \phi + \rho \rfloor}, u_2 = \frac{1-\phi}{\rho - \lfloor \phi + \rho \rfloor}$. Let $\lambda_1 = \frac{\phi}{\rho - \lfloor \phi + \rho \rfloor}$ and $\lambda_2 = \frac{1}{\rho - \lfloor \phi + \rho \rfloor}$. Since u is on the B-side boundary of the split, we have $\phi < \widehat{\phi + \rho}$. That is, $\phi < \phi + \rho - \lfloor \phi + \rho \rfloor$, which implies $\rho - \lfloor \phi + \rho \rfloor > 0$. Since, by assumption, $0 < \phi < 1$, we conclude that $\lambda_1, \lambda_2 \geq 0$. Using the previous characterization of u, it is also straightforward to verify that, if we multiply the valid inequality $-x_1 + \lfloor \phi + \rho \rfloor x_2 \leq -1$ by λ_1, multiply the valid inequality $x_1 - \rho x_2 \leq \phi$ by λ_2, and then sum the resulting inequalities, we obtain $u_2x_1 - u_1x_2 \leq 0$, as desired.

Since $u_2x_1 - u_1x_2 \leq 0$ is valid, we may write

$$A = \text{conv}\left\{ x \in \mathbb{Z}^2 : \begin{array}{l} u_2x_1 - u_1x_2 \leq 0 \\ x_1 - \rho x_2 \leq \phi \\ x_2 \geq 0 \end{array} \right\}.$$

It is not hard to see that $(0, 0)$ and u are the only vertices of the linear relaxation of this set. Since the linear relaxation has integer vertices, it coincides with its integer hull. We conclude that $(0, 0)$ and u are the only vertices of A. The proof for $\text{vert}(B)$ is similar. ■

Now we consider two more interesting cases, when $u \notin \mathbb{Z}^2$. In the first case, illustrated in Figure 5.3, the ray hits the boundary of the split on the "B-side", i.e. on the line $x_1 - \lfloor \phi + \rho \rfloor x_2 = 1$. The next proposition describes two sets $\bar{A}$ and $\bar{B}$ such that $\text{vert}(A) = \text{vert}(\bar{A})$ and $\text{vert}(B) = \{\binom{1}{0}\} \cup \text{vert}(\bar{B})$. That is, in order to enumerate the vertices of A and B, it is sufficient to enumerate the vertices of $\bar{A}$ and $\bar{B}$, then add the point $\binom{1}{0}$. The vertices of $\bar{A}$ and $\bar{B}$ can be enumerated recursively, as we will see later.



(a) A, B, u, v and $\mathcal{S}$ (dashed lines) (b) $\bar{A}$ and $\bar{B}$

Figure 5.3: Illustration of proposition 49.

Proposition 49. *Suppose $u \notin \mathbb{Z}^2$ and $\phi < \widehat{\phi + \rho}$. Let v be the lattice point closest to u in the segment between u and $\binom{1}{0}$. Let $\bar{f}$ be the intersection between the segment connecting $\binom{0}{0}$ to v, and the segment connecting $\binom{\phi}{0}$ and u. Define*

$$\bar{A} = \operatorname{conv}\left(\mathbb{Z}^2 \cap \left(\bar{f} + \operatorname{cone}\left\{\binom{\rho}{1}, \binom{0}{0} - \bar{f}\right\}\right)\right)$$
$$\bar{B} = \operatorname{conv}\left(\mathbb{Z}^2 \cap \left(\bar{f} + \operatorname{cone}\left\{\binom{\rho}{1}, v - \bar{f}\right\}\right)\right)$$

Then $\operatorname{vert}(A) = \operatorname{vert}(\bar{A})$ *and* $\operatorname{vert}(B) = \{\binom{1}{0}\} \cup \operatorname{vert}(\bar{B})$.

Proof. (i) First, observe that $\bar{A}$ can be written as

$$\bar{A} = \operatorname{conv}\left\{x \in \mathbb{Z}^2 : \begin{array}{l} x_1 - \rho x_2 \leq \phi \\ v_2 x_1 - v_1 x_2 \leq 0 \end{array}\right\}.$$

Also, because the triangle $\binom{0}{0}, \binom{\phi}{0}, \bar{f}$ is contained in the split $\mathcal{S}$, and thus contains no integral points, the inequality $v_2 x_1 - v_1 x_2 \leq 0$ can be added to the description of the set A, without affecting its definition:

$$A = \operatorname{conv}\left\{x \in \mathbb{Z}^2 : \begin{array}{l} x_1 - \rho x_2 \leq \phi \\ v_2 x_1 - v_1 x_2 \leq 0 \\ x_2 \geq 0 \end{array}\right\}.$$

Thus, it becomes clear that $A \subseteq \bar{A}$.

In the following, we prove that $\operatorname{vert}(A) = \operatorname{vert}(\bar{A})$. Let $x \in \mathbb{R}^2$. First, we prove that, if $x \notin \operatorname{vert}(A)$, then $x \notin \operatorname{vert}(\bar{A})$. We may assume $x \in \bar{A}$, since otherwise x is clearly not a vertex of $\bar{A}$, and there is nothing to prove. We have two cases. In the first case, suppose $x \in A$. Since x is not a vertex of A, there exist $y^1, y^2 \in A \setminus \{x\}$

such that $x = \frac{1}{2}y^1 + \frac{1}{2}y^2$. Since $A \subseteq \bar{A}$, then $y^1, y^2 \in \bar{A}$, and we conclude that x is also not a vertex of $\bar{A}$. In the second case, suppose $x \notin A$. This implies $x_2 < 0$. Let $y^1 = x - \varepsilon v, y^2 = x + \varepsilon v$, where $\varepsilon = -\frac{x_2}{v_2} > 0$. Note that $y^1 \in \bar{A}$, since

$$y^1 = \underbrace{x}_{\in \bar{A}} + \underbrace{\varepsilon}_{\geq 0} \underbrace{(-v)}_{\in \text{rec}(\bar{A})}.$$

We also have $y^2 \in \bar{A}$, since

$$y^2 = \underbrace{\begin{pmatrix} 0 \\ 0 \end{pmatrix}}_{\in \bar{A}} + \underbrace{\left(-x_1 + \frac{v_1 x_2}{v_2}\right)}_{\geq 0} \underbrace{\begin{pmatrix} -1 \\ 0 \end{pmatrix}}_{\in \text{rec}(\bar{A})}.$$

Since $y^1, y^2 \in \bar{A}$ and $x = \frac{1}{2}y^1 + \frac{1}{2}y^2$, we conclude, also in this case, that $x \notin \text{vert}(\bar{A})$.

Now we prove that, if $x \notin \text{vert}(\bar{A})$, then $x \notin \text{vert}(A)$. Similarly, we may assume $x \in A$, otherwise x is clearly not a vertex of A, and there is nothing to prove. Since $A \subseteq \bar{A}$, this also implies that $x \in \bar{A}$. We have two cases. In the first case, suppose $x_2 = 0$. Since x satisfies $v_2 x_1 - v_1 x_2 \leq 0$, then $x_1 \leq 0$. Since x is not a vertex of $\bar{A}$, then $x \neq \left(\begin{smallmatrix} 0 \\ 0 \end{smallmatrix}\right)$, and $x_1 < 0$. Therefore,

$$x = \underbrace{\begin{pmatrix} 0 \\ 0 \end{pmatrix}}_{\in A} + \underbrace{(-x_1)}_{>0} \underbrace{\begin{pmatrix} -1 \\ 0 \end{pmatrix}}_{\in \text{rec}(A)}.$$

We conclude that x is not a vertex of A. In the second case, suppose $x_2 > 0$. Since x is not a vertex of $\bar{A}$, there exists $d \in \mathbb{R}^2 \setminus \{0\}$ such that $x + d, x - d \in \bar{A}$. Let $y^1 = x + \varepsilon d, y^2 = x - \varepsilon d$, for some $\varepsilon > 0$. If ε is small enough, we have $y^1_2, y^2_2 \geq 0$, which implies $y^1, y^2 \in A$. Since $x = \frac{1}{2}y^1 + \frac{1}{2}y^2$, we conclude that x is not a vertex of A.

(ii) First, observe that B and $\bar{B}$ can be written as

$$B = \text{conv}\left\{ x \in \mathbb{Z}^2 : \begin{array}{l} x_1 - \rho x_2 \geq \phi \\ x_2 \geq 0 \end{array} \right\},$$

$$\bar{B} = \text{conv}\left\{ x \in \mathbb{Z}^2 : \begin{array}{l} x_1 - \rho x_2 \geq \phi \\ v_2 x_1 - v_1 x_2 \leq 0 \\ x_2 \geq 0 \end{array} \right\}.$$

Thus, it is clear that $\bar{B} \subseteq B$. In the following, we prove that $\text{vert}(B) = \{\left(\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right)\} \cup \text{vert}(\bar{B})$. Let $x \in \mathbb{R}^2$. First, we prove that if $x \notin \text{vert}(B)$ then $x \neq \left(\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right)$ and $x \notin \text{vert}(\bar{B})$. It is

easy to see that $\binom{1}{0} \in \text{vert}(B)$, thus $x \neq \binom{1}{0}$. Furthermore, if $x \notin \bar{B}$, then x is clearly not a vertex of $\bar{B}$ and we done, so we assume $x \in \bar{B}$. We have two cases. In the first case, suppose $v_2 x_1 - v_1 x_2 = 0$. We can prove that there exists $\varepsilon > 0$ such that

$$x = \underbrace{v}_{\in \bar{B}} + \underbrace{\varepsilon}_{>0} \underbrace{v}_{\in \text{rec}(\bar{B})}.$$

Therefore, x is not a vertex of $\bar{B}$. In the second case, suppose $v_2 x_1 - v_1 x_2 < 0$. Since x is not a vertex of B, there exists $d \in \mathbb{R}^2 \setminus \{0\}$ such that $x + d, x - d \in B$. For a small enough $\varepsilon > 0$, we can prove that $x + \varepsilon d, x - \varepsilon d \in \bar{B}$. In either case, we conclude that x is not a vertex of $\bar{B}$.

Now we prove that, if $x \neq \binom{1}{0}$ and $x \notin \text{vert}(\bar{B})$, then $x \notin \text{vert}(B)$. Clearly, if $x \notin B$ then $x \notin \text{vert}(B)$ and we are done, so we assume $x \in B$. Once again, we have two cases. In the first case, suppose $x \in \bar{B}$. Since $x \notin \text{vert}(\bar{B})$, there exist $y^1, y^2 \in \bar{B} \setminus \{x\}$ such that $x = \frac{1}{2} y^1 + \frac{1}{2} y^2$. Since $\bar{B} \subseteq B$, then $y^1, y^2 \in B$. Therefore, x is not a vertex of B. In the second case, suppose $x \notin \bar{B}$. Since $x \neq \binom{1}{0}$, it is clear that, if $x_2 = 0$, then x is not a vertex of B and we are done. Therefore, we assume $x_2 > 0$. Let

$$y^1 = x + \epsilon \left[v - \tbinom{1}{0} \right], y^2 = x - \epsilon \left[v - \tbinom{1}{0} \right],$$

where $\epsilon > 0$. It is not hard to prove that, for a small enough ϵ, we have $y^1, y^2 \in B$. Since $x = \frac{1}{2} y^1 + \frac{1}{2} y^2$, we conclude that, in any case, $x \notin \text{vert}(B)$. ■

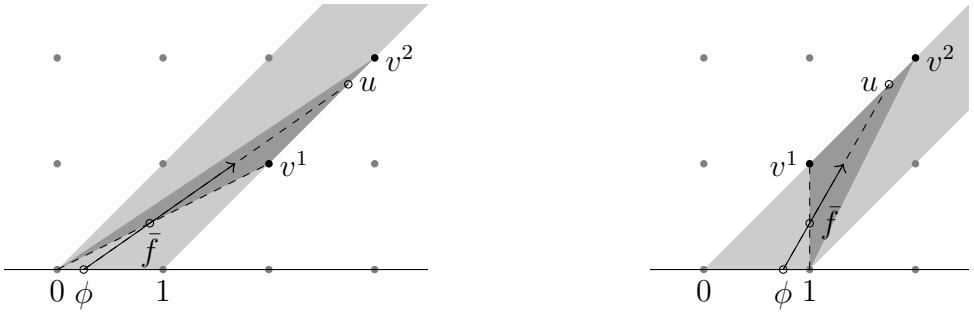
In the case where the ray hits the boundary of the split on the "A-side", i.e. on the line $0 = x_1 - \lfloor \phi + \rho \rfloor x_2$, we have a similar result. The next proposition describes two sets $\bar{A}$ and $\bar{B}$ such that $\text{vert}(A) = \{\binom{0}{0}\} \cup \text{vert}(\bar{A})$ and $\text{vert}(B) = \text{vert}(\bar{B})$. We skip its proof since it is analogous to the proof of Proposition 49.

Proposition 50. *Suppose $u \notin \mathbb{Z}^2$ and $\phi > \widehat{\phi + \rho}$. Let v be the lattice point closest to u in the segment between u and $\binom{0}{0}$. Let $\bar{f}$ be the intersection between the segment connecting $\binom{1}{0}$ to v, and the segment connecting $\binom{\phi}{0}$ and u. Define*

$$\bar{A} = \text{conv}\left(\mathbb{Z}^2 \cap \left(\bar{f} + \text{cone}\left\{\tbinom{\rho}{1}, v - \bar{f}\right\}\right)\right)$$
$$\bar{B} = \text{conv}\left(\mathbb{Z}^2 \cap \left(\bar{f} + \text{cone}\left\{\tbinom{\rho}{1}, \tbinom{0}{0} - \bar{f}\right\}\right)\right)$$

Then $\text{vert}(A) = \{\binom{0}{0}\} \cup \text{vert}(\bar{A})$ *and* $\text{vert}(B) = \text{vert}(\bar{B})$.

Now the only question remaining is how to compute the vertices of $\bar{A}$ and $\bar{B}$. The next proposition shows that this can be done recursively. By applying an appropriate affine



(a) $\phi < \widehat{\phi + \rho}$ (b) $\phi > \widehat{\phi + \rho}$

Figure 5.4: How $\bar{f}$, u, v^1 and v^2 are found in Proposition 51.

integral unimodular transformation to the coordinate system and scaling of the rays, the sets $\bar{A}$ and $\bar{B}$ can be written in the same form as the original sets A and B. Therefore, the vertices of $\bar{A}$ and $\bar{B}$ can be obtained by recursively applying Propositions 47–50.

Proposition 51. *Suppose $u \notin \mathbb{Z}^2$. If $\phi < \widehat{\phi + \rho}$, let $\bar{A}$ and $\bar{B}$ be defined as in Proposition 49. If $\phi > \widehat{\phi + \rho}$, let $\bar{A}$ and $\bar{B}$ be defined as in Proposition 50. In either case, there exist $\bar{\phi}, \bar{\rho} \in \mathbb{R}$ and an affine integral unimodular transformation $\tau : \mathbb{R}^2 \to \mathbb{R}^2$ such that*

$$\tau(\bar{A}) = \operatorname{conv}\left(\mathbb{Z}^2 \cap \left(\binom{\bar{\phi}}{0} + \operatorname{cone}\{\binom{\bar{\rho}}{1}, \binom{-1}{0}\}\right)\right),$$
$$\tau(\bar{B}) = \operatorname{conv}\left(\mathbb{Z}^2 \cap \left(\binom{\bar{\phi}}{0} + \operatorname{cone}\{\binom{\bar{\rho}}{1}, \binom{1}{0}\}\right)\right).$$

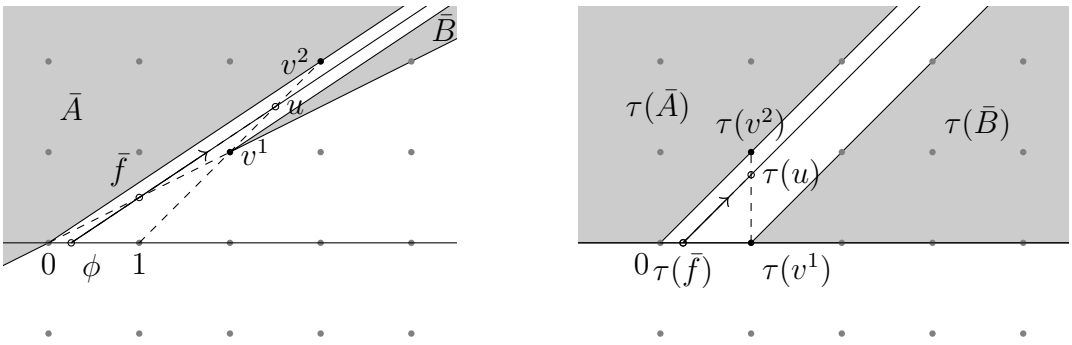
Proof. Suppose $\phi < \widehat{\phi + \rho}$.

Let $\bar{f}$ and $v \in \mathbb{Z}^2$ as defined in Proposition 49. Let $v^1 = v$ and let v^2 be the lattice point closest to u in the half-line $u + \lambda(u - \binom{1}{0}), \lambda \geq 0$. That is, v^1 and v^2 are the closest lattice points to u in the line passing through u and $\binom{1}{0}$ (see figure 5.4a).

Let $\tau : \mathbb{R}^2 \to \mathbb{R}^2$ be an affine function such that $\tau\binom{0}{0} = \binom{0}{0}, \tau(v^1) = \binom{1}{0}, \tau(v^2) = \binom{1}{1}$. Such a transformation exists, since v^1 and v^2 are linearly independent. Furthermore, it is integral and unimodular, since the triangle defined by $\binom{0}{0}, v^1$ and v^2 has integral vertices and its area equals $\frac{1}{2}$. Therefore,

$$\tau(\bar{A}) = \operatorname{conv}\left(\mathbb{Z}^2 \cap \left(\tau(\bar{f}) + \operatorname{cone}\left\{\tau\binom{\rho}{1}, \tau\left(\binom{0}{0} - \bar{f}\right)\right\}\right)\right)$$
$$\tau(\bar{B}) = \operatorname{conv}\left(\mathbb{Z}^2 \cap \left(\tau(\bar{f}) + \operatorname{cone}\left\{\tau\binom{\rho}{1}, \tau(v^1 - \bar{f})\right\}\right)\right)$$

Since $\bar{f} \in \operatorname{conv}\{\binom{0}{0}, v^1\}$, then $\tau(\bar{f}) \in \operatorname{conv}\{\binom{0}{0}, \binom{1}{0}\}$, which implies that there exists $\bar{\phi} \in \mathbb{R}$ such that $\tau(\bar{f}) = \binom{\bar{\phi}}{0}$. Furthermore, it is not hard to see that there exist



(a) Figure 5.3b zoomed in. (b) Transformed by τ

Figure 5.5: Transformation τ of Proposition 51.

$\lambda_1, \lambda_2, \lambda_3 \in \mathbb{R}_+$ such that

$$\lambda_1 \tau \begin{pmatrix} \rho \\ 1 \end{pmatrix} = \begin{pmatrix} \bar{\rho} \\ 1 \end{pmatrix}, \lambda_2 \tau \left(\begin{pmatrix} 0 \\ 0 \end{pmatrix} - \bar{f} \right) = \begin{pmatrix} -1 \\ 0 \end{pmatrix}, \lambda_3 \tau \left(v^1 - \bar{f} \right) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

This concludes the proof for this case (see Figure 5.5). When $\phi > \widehat{\phi + \rho}$, the proof is similar, constructing v^1 and v^2 in an analogous way (see figure 5.4b), but we let τ be an affine function satisfying $\tau \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \tau(v^1) = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \tau(v^2) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$, instead. ■

Using Propositions 47–50, we now have a complete recursive algorithm for computing the vertices of A and B. The first step is to verify whether $\phi = \widehat{\phi + \rho}$. If so, the vertices are given by Proposition 47. If not, we construct the split $\mathcal{S}$ and verify whether the intersection of its boundary with the ray $\begin{pmatrix} \phi \\ 0 \end{pmatrix} + \text{cone} \begin{pmatrix} \rho \\ 1 \end{pmatrix}$ is an integral point. If so, the vertices of A and B are given by Proposition 48. If not, then either Propositions 49 or 50 apply, in which case the vertices of A and B are the same as the vertices of $\bar{A}$ and $\bar{B}$, in addition to either $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ or $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$. In order to compute the vertices of $\bar{A}$ and $\bar{B}$, we proceed recursively. First, we apply the transformation described in Proposition 51, so that $\bar{A}$ and $\bar{B}$ are written in the same form as the original sets A and B, then we repeatedly apply Propositions 47–50. A non-recursive version of this algorithm is described in Algorithm 52.

Now that we have an algorithm for enumerating the vertices of A and B, we finish this section by describing how can we use the previous propositions to get a complete list of maximal S-free sets that induce facets of P_I.

Definition 53 describes the sequence of S-free sets that we construct, one per iteration of the vertex enumeration algorithm. Note that the definition is recursive. Given ϕ and ρ, Propositions 47–50 show how to compute one S-free set W_u. Then, Proposition 51 provides an affine transformation τ and a new model, determined by $\bar{\phi}$ and $\bar{\rho}$, which will

Algorithm 52 Algorithm for enumerating the vertices of the knapsack sets

function EnumerateVertices(ϕ, ρ)
  $U \leftarrow I$, $t \leftarrow \mathbf{0}$
  $X^A \leftarrow \{(0,0)\}$, $X^B \leftarrow \{(1,0)\}$
  loop
    if $\phi == \widehat{\phi + \rho}$ **then**
      return X^A, X^B
    else if $\phi < \widehat{\phi + \rho}$ **then**
      $u \leftarrow \left(\frac{\rho - \phi\lfloor\phi+\rho\rfloor}{\rho - \lfloor\phi+\rho\rfloor}, \frac{1-\phi}{\rho-\lfloor\phi+\rho\rfloor}\right)$
      if $u \in \mathbb{Z}^2$ **then**
        return $X^A \cup \{Uu + t\}, X^B \cup \{Uu + t\}$
      $v^1 \leftarrow (1 + \lfloor\phi+\rho\rfloor \lfloor u_2 \rfloor, \lfloor u_2 \rfloor)$
      $v^2 \leftarrow (1 + \lfloor\phi+\rho\rfloor \lceil u_1 \rceil, \lceil u_1 \rceil)$
      $X^B \leftarrow X^B \cup \{Uv^1 + t\}$
      $W \leftarrow \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_1^1 & v_1^2 \\ v_2^1 & v_2^2 \end{bmatrix}^{-1}$
      $y \leftarrow \begin{pmatrix} 0 \\ 0 \end{pmatrix}$
      $f' \leftarrow \left(\frac{v_2^1\phi}{v_1^1 - v_2^1\rho}, \frac{v_1^1\phi}{v_1^1 - v_2^1\rho}\right)$
    else if $\phi > \widehat{\phi + \rho}$ **then**
      $u \leftarrow \left(\frac{\phi\lfloor\phi+\rho\rfloor}{\lfloor\phi+\rho\rfloor - \rho}, \frac{\phi}{\lfloor\phi+\rho\rfloor-\rho}\right)$
      if $u \in \mathbb{Z}^2$ **then**
        return $X^A \cup \{Uu + t\}, X^B \cup \{Uu + t\}$
      $v^1 \leftarrow (\lfloor\phi+\rho\rfloor \lfloor u_2 \rfloor, \lfloor u_2 \rfloor)$
      $v^2 \leftarrow (\lfloor\phi+\rho\rfloor \lceil u_1 \rceil, \lceil u_1 \rceil)$
      $X^A \leftarrow X^A \cup \{Uv^1 + t\}$
      $W \leftarrow \begin{bmatrix} -1 & -1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_1^1 - 1 & v_1^2 - 1 \\ v_2^1 & v_2^2 \end{bmatrix}^{-1}$
      $y \leftarrow \begin{pmatrix} 1 \\ 0 \end{pmatrix} - W \begin{pmatrix} 1 \\ 0 \end{pmatrix}$
      $f' \leftarrow \left(\frac{\phi(v_1^1-1) - \rho v_2^1}{\rho v_2^1 - (v_1^1 - 1)}, \frac{(1-\phi)v_2^1}{\rho v_2^1 - (v_1^1-1)}\right)$
    $f \leftarrow Wf' + y$, $\phi \leftarrow f_1$
    $r \leftarrow W \begin{pmatrix} \rho \\ 1 \end{pmatrix} + y$, $\rho \leftarrow \frac{r_1}{r_2}$
    $t \leftarrow t - UW^{-1}y$
    $U \leftarrow UW^{-1}$

yield further S-free sets. The sequence $\mathcal{W}(\phi, \rho)$ is constructed by concatenating W_u and the subsequent S-free sets $\mathcal{W}(\bar{\phi}, \bar{\rho})$ given by the new model, suitably transformed back into the original space.

Definition 53. *Let $\mathcal{W}(\phi, \rho) = \langle W_1, \dots, W_k \rangle$ be a sequence of sets defined as follows:*

(i) If the conditions of Proposition 47 are satisfied, then $\mathcal{W}(\phi, \rho) = \langle \mathcal{S} \rangle$, where $\mathcal{S}$ is the split defined previously.

(ii) If the conditions of Proposition 48 are satisfied, then $W(\phi, \rho) = \langle W_u \rangle$, where

$$W_u = u + \operatorname{cone}\left\{\left(\begin{smallmatrix}0\\0\end{smallmatrix}\right) - u, \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right) - u\right\}.$$

(iii) Suppose that the conditions of either Proposition 49 or Proposition 50 are satisfied. Let $\bar{\phi}, \bar{\rho}, \tau$ be defined as in Proposition 51, and let

$$\mathcal{W}(\bar{\phi}, \bar{\rho}) = \left\langle \bar{W}_1, \dots, \bar{W}_l \right\rangle.$$

Then we define

$$\mathcal{W}(\phi, \rho) = \left\langle W_u, \tau^{-1}(\bar{W}_1), \dots, \tau^{-1}(\bar{W}_l) \right\rangle,$$

where W_u is defined as in (ii).

For every $j \in \{1, \dots, k\}$, it is easy to see that W_j is tight at three integral points; either two vertices of A and one vertex of B, or two vertices of B and one vertex of A. Observe moreover that, for any combination of three vertices not generated in this fashion, one could not construct a corresponding S-free wedge: First, note that the two vertices belonging to the same side must be consecutive, otherwise the wedge cannot be S-free. Then, given a pair of tight vertices on one side, the S-free wedge that is tight at those vertices and a third on the other side is unique. For every pair of consecutive vertices of either A or B there is a wedge W_j that is tight for these vertices. W_j also has a vertex that is tight on the other side. If we replace this third vertex by any other, the other vertex will either be on the boundary or outside of the initial wedge. In the first case, the new wedge would be identical to the initial one, and in the second case, it would not be S-free.

The next proposition shows that W_j is also S-free. Then Theorem 46 implies that the intersection cut from W_j yields a facet-defining inequality for $\operatorname{conv}(P_I)$. By Proposition 44, we now have a complete H-description of $\operatorname{conv}(P_I)$.

Proposition 54. *Every set in $\mathcal{W}(\phi, \rho)$ is maximal and S-free.*

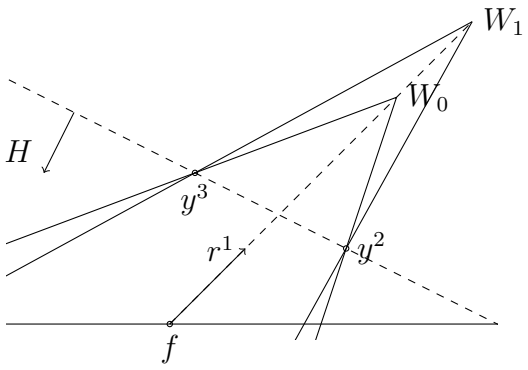


Figure 5.6: Wedges W_0 and W_1 in the configuration of Lemma 55.

Proof. We prove the claim by structural induction. If $\mathcal{W}(\phi, \rho) = \langle S \rangle$ or $\mathcal{W}(\phi, \rho) = \langle W_u \rangle$, then the proposition is clearly true. Now suppose $\mathcal{W}(\phi, \rho) = \left\langle W_u, \tau^{-1}(\bar{W}_1), \ldots, \tau^{-1}(\bar{W}_l) \right\rangle$, and suppose, by induction, that $\bar{W}_1, \ldots, \bar{W}_l$ are maximal S-free sets containing $\binom{\bar{\phi}}{0}$ in their interior. Clearly, W_u is maximal S-free and contains $\binom{\phi}{0}$. Let $j \in \{1, \ldots, k\}$. We prove that the same holds for $\bar{W}_j$. Since $\bar{W}_j$ is S-free, then $\tau^{-1}(W_j)$ does not contain any integral points above the line that connects $(0, 0)$ and v^1 (in the first case of Proposition 51) or $(1, 0)$ and v^1 (in the second case). Furthermore, the region of $\tau^{-1}(\bar{W}_j)$ that lies below the line is entirely contained in W_u. Therefore, $\bar{W}_j$ is S-free. Since $\bar{W}_j$ is maximal, it is not hard to see that $\tau^{-1}(W_j)$ is also maximal. ■

5.3 Upper bound on the split rank

In this section, we prove that the split rank of $\text{conv}(P_I)$ is at most the sum of the number of vertices of A and the number of vertices of B. In the following, let

$$P_{LP} = \left\{(x, s) \in \mathbb{R}^2 \times \mathbb{R}^3_+ : \binom{x_1}{x_2} = \binom{\phi}{0} + \binom{\rho}{1} s_1 + \binom{1}{0} s_2 + \binom{-1}{0} s_3\right\}$$

be the linear relaxation of P_I. In order to prove our result, we first need Lemma 55. It shows that given two wedges in a specific configuration and their induced intersection cuts, there is a half-plane where any point cut off by one is cut off by the other.

Lemma 55. *Let W_0 and W_1 be two distinct wedges with apex on $f + \text{cone}(r^1)$ and containing f in their relative interiors. Let $(\alpha^0)^T s \geq 1$ and $(\alpha^1)^T s \geq 1$ be the intersection cuts obtained from W_0 and W_1, respectively. Suppose that $\alpha^1_1 < \alpha^0_1$, and that the boundaries of W_0, W_1 intersect at two distinct points $y^2, y^3 \in \mathbb{R}^2$. Let $H \subseteq \mathbb{R}^2$ be the closed half-space that contains y^2, y^3 in its boundary and does not contain the apices of W_0, W_1 (Figure 5.6). If $f \in \text{interior}(H)$, then, for any $(\bar{x}, \bar{s}) \in P_{LP}$ such that $\bar{x} \in \text{interior}(H)$*

and ${\alpha^1}^T \bar{s} < 1$*, we also have* ${\alpha^0}^T \bar{s} < 1$*.*

Proof. First, we construct $\alpha^h \in \mathbb{R}^3$ such that, for any $(\bar{x}, \bar{s}) \in P_{LP}$, we have $\bar{x} \in$ interior(H) if and only if $(\alpha^h)^T \bar{s} < 1$. The statement $\bar{x} \in$ interior(H) is equivalent to $c^T \bar{x} < d$ for some $c \in \mathbb{R}^2, d \in \mathbb{R}$. Recalling that $\bar{x} = f + R\bar{s}$, this is equivalent to $c^T(f + R\bar{s}) < d$, i.e., $(c^T R)\bar{s} < d - c^T f$. Since f belongs to the interior of H, we know that $c^T f < d$, so the right-hand side of the previous equation is positive. Dividing by that right-hand side, we obtain the desired form $(\alpha^h)^T \bar{s} < 1$ where $\alpha^h = \frac{c^T R}{d - c^T f}$.

Next, we prove that α^0 is a convex combination of α^1 and α^h. Consider the three lines $\alpha_1^0 s_1 + \alpha_2^0 s_2 = 1$, $\alpha_1^1 s_1 + \alpha_2^1 s_2 = 1$ and $\alpha_1^h s_1 + \alpha_2^h s_2 = 1$. Note that each of these correspond to one face of each of W_0 and W_1 and H, so they intersect in a single point y^2. Therefore, $(\alpha_1^0, \alpha_2^0) = \lambda^2(\alpha_1^h, \alpha_2^h) + (1 - \lambda^2)(\alpha_1^1, \alpha_2^1)$, for some $\lambda^2 \in \mathbb{R}$. Similarly, for the other intersection y^3, we obtain $(\alpha_1^0, \alpha_3^0) = \lambda^3(\alpha_1^h, \alpha_3^h) + (1 - \lambda^3)(\alpha_1^1, \alpha_3^1)$. for some $\lambda^3 \in \mathbb{R}$. Together, these relationships show $\lambda^2 = \lambda^3$. Let $\lambda := \lambda^2 = \lambda^3$, we get $\alpha^0 = \lambda\alpha^h + (1 - \lambda)\alpha^1$. Since $\alpha_1^1 < \alpha_1^0$ and H does not contain the apex of W_0 or W_1, we have that $\alpha_1^1 < \alpha_1^0 < \alpha_1^h$, so α^0 is not only a linear combination of α^h and α^1, but also a convex combination (i.e. $0 \leq \lambda \leq 1$). Therefore, ${\alpha^h}^T s < 1$ and ${\alpha^1}^T s < 1$ together imply ${\alpha^0}^T s < 1$. ■

Let $W_1, \ldots, W_k$ be as defined in Subsection 5.2. The next theorem shows that the intersection cut from W_j has a split rank at most j.

Theorem 56. *For every $j \in \{1, \ldots, k\}$, the intersection cut from W_j has split rank at most j.*

Proof. We prove the claim by induction. The first wedge W_1 has the same intersection points as the split $\mathcal{S}$, so the corresponding cut has split rank 1. Let $j \in \{2, \ldots, k\}$ and assume now that W_{j-1} yields a cut of split rank $j - 1$ or less. In the following, we prove that the intersection cut generated from W_j is implied by the intersection cut generated from W_{j-1} together with a split cut, and, therefore, W_j yields a cut with split rank at most j. We assume that W_j is not a split, otherwise there is nothing to prove. Note that W_{j-1} and W_j are in the same configuration as wedges W_0 and W_1 of Lemma 55. Let $(x, s) \in P_{LP}$ be a point that does not satisfy the intersection cut from wedge W_j. This implies that $x \in$ interior(W_j). If $x \in$ interior(H), we apply Lemma 55 to show that (x, s) also does not satisfy the cut from wedge W_{j-1}. If $x \notin$ interior(H), then x belongs to the interior of the split that was considered when generating W_j, hence (x, s) does not satisfy the intersection cut obtained from this split. We conclude that the intersection cut obtained from W_j is implied by the cut from W_{j-1} together with a split cut. Since

the cut obtained from W_j has split rank at most $j-1$, then the cut obtained from W_j has split rank at most j. ■

Corollary 57. *Let k_2 and k_3 be the number of vertices of* $\text{conv}(K_2)$ *and* $\text{conv}(K_3)$*, respectively. The split rank of* $\text{conv}(P_I)$ *is at most $k_2 + k_3 - 1$.*

We finish this section by noting that 2-step MIR inequalities [33] can be derived as inequalities for $\text{conv}(P_I)$, and in fact they can be seen as inequalities obtained once the algorithm switches from using Proposition 49 to using Proposition 50 (or vice-versa) for the first time. This suggests that perhaps a lower bound on the split rank may be obtained by considering such cases, since 2-step MIR inequalities have split rank 2. We were, however, unable to derive any such lower bound.

5.4 Computational Experiments

In order to evaluate the strength of wedge cuts, we implemented a cut generator and tested it on the benchmark set of the MIPLIB 2010. We measured the gap closed by the inclusion of wedge cuts and compared it to the gap closed by considering GMI cuts alone.

The cut generator performed the following steps. First, the linear relaxation of the presolved problem was solved, and a certain basic solution with value z_{LP} was obtained. The optimal tableau was stored. Although we solved the relaxation again at a later time, we always used this first optimal tableau to generate all the cuts, hence obtaining only rank-1 cuts. Next, for each row of the tableau corresponding to an integral basic variable, an GMI cut was generated and added to the problem. The strengthened relaxation was then solved again, and another basic solution x^{GMI}, with value z_{GMI} was obtained. Then, every possible wedge cut was generated and added to the problem, provided that it cut off the previous solution x^{GMI}. More precisely, for each row of the tableau corresponding to an integral basic variable, and for each integral non-basic variable x_i that has non-zero coefficient in that row, we identified the coefficient corresponding to x_i with ρ, and generated all the facet-defining wedge and split cuts, as described in Section 5.1. The cut coefficients for the remaining integral non-basic variables was calculated according to the algorithm from Chapter 3. Finally, the relaxation was solved again, and a basic solution with value z_W was obtained. In the following, we also denote by z_{OPT} the value of the optimal solution for the original mixed-integer problem.

The cut generator was implemented in C++ and compiled with the GNU C++ Compiler 4.8.4. The complete source code has been made available online [67]. For the LP

solver, we used the library IBM ILOG CPLEX 12.6.2. Considerable care was taken to avoid the generation of invalid cuts. CPLEX was configured for numerical emphasis, and once the LP was solved, each double-precision floating point entry of the resulting tableau was converted to an exact rational number. To avoid the propagation of floating point errors, the enumeration of the facets of the knapsack sets was performed using exact arithmetic, with the help of the GNU Multiple Precision Arithmetic Library 6.1.0 [49]. The cut coefficients were then converted back to double-precision floating point numbers and given to CPLEX. We discarded all cuts with high coefficient dynamism (ratio between the magnitudes of largest and the smallest coefficients of 10^6 or larger), then considered only the remaining inequalities that cut off the original fractional solution x^{LP} by a significant amount (10^{-6} or more).

Our testbed was the benchmark set of the MIPLIB 2010, which is composed by 87 instances of real-world mixed integer programs. For each instance, the following performance indicators were computed:

- ORIG-GAP, the original gap between the first linear relaxation and the original mixed-integer program:
$$\frac{z_{OPT} - z_{LP}}{|z_{OPT}|}$$
- GMI-PERF, the amount of the original gap that was closed by the inclusion of the GMI inequalities:
$$\frac{z_{GMI} - z_{LP}}{z_{OPT} - z_{LP}}$$
- W-PERF, the amount of the original gap that what was closed by the inclusion of all the wedge inequalities:
$$\frac{z_{W} - z_{LP}}{z_{OPT} - z_{LP}}$$
- W-REL, the contribution of the wedge cuts to the gap closure; that is, the amount of the original gap that was closed by wedge inequalities which are not equivalent to GMI inequalities:
$$\frac{z_{W} - z_{GMI}}{z_{W} - z_{LP}}$$
- TIME, the total CPU time, in seconds, required process the instance.

Out of the 87 instances, three were infeasible (`ash608gpia-3col`, `enlight14`, `ns1766074`) and four (`acc-tight5`, `bnatt350`, `m100n500k4r1`, `neos-849702`) had z_{LP} equal to z_{OPT}. These instances were not considered. Ten instances exceeded our 60 hour CPU-time

Instance	ORIG-GAP (%)	MIR-PERF (%)	W-PERF (%)	W-REL (%)	TIME (s)
gmu-35-40.pre	0.01	0.07	9.94	99.26	122
eil33-2.pre	13.14	4.28	15.25	71.97	8972
neos-1337307.pre	0.4	3.76	6.45	41.66	32572
opm2-z7-s2.pre	25.29	0.62	0.98	37.17	284884
mik-250-1-100-1.pre	19.65	53.52	73.38	27.07	14
neos-686190.pre	23.7	4.61	5.54	16.82	33697
mine-90-10.pre	11.15	12.4	14.51	14.6	970
cov1075.pre	14.29	3.6	4.19	13.9	95
mine-166-5.pre	45.09	6.57	7.58	13.35	1892
n3div36.pre	12.59	16.38	18.85	13.09	268969
air04.pre	1.07	8.14	9.12	10.81	144318
rococoC10-001000.pre	34.42	21.16	22.41	5.58	226
rmine6.pre	1.12	14.57	15.34	5	2751
reblock67.pre	11.61	21.38	22.46	4.81	446
ran16x16.pre	18.48	17.25	18.07	4.5	1
iis-bupa-cov.pre	26.4	1.22	1.26	3.59	9892
sp98ir.pre	1.37	4.63	4.77	2.88	10242
iis-pima-cov.pre	19.33	2.1	2.14	1.94	34347
iis-100-0-cov.pre	42.53	1.76	1.79	1.89	48
eilB101.pre	11.64	2.64	2.69	1.82	13247
mzzv11.pre	4.86	26.99	27.11	0.43	49301
roll3000.pre	13.9	21.83	21.91	0.37	356
dfn-gwin-UUM.pre	29.12	41.82	41.9	0.18	0
csched010.pre	18.52	3.89	3.9	0.15	647
msc98-ip.pre	1.56	17.78	17.81	0.14	26752
neos-916792.pre	17.53	4.06	4.06	0.14	112
mcsched.pre	8.56	0.04	0.04	0.08	4634
beasleyC3.pre	68.44	15.58	15.59	0.05	0

Table 5.1: Strength of wedge cuts versus GMI cuts alone.

Instance	CUTS-GMI	CUTS-W	GMI-T	WEDGE-T
cov1075	582	174970	0.16	0.20
eil33-2	30	566411	7.61	8.35
gmu-35-40	27	58555	0.83	1.16
mik-250-1-100-1	100	30221	0.17	0.28
mine-166-5	1436	1336080	0.29	0.57
mine-90-10	1875	1022638	0.18	0.38
n3div36	48	3838798	32.06	41.67
neos-1337307	2263	8302981	1.13	1.52
neos-686190	254	3162782	5.56	5.54
opm2-z7-s2	7859	38797773	3.26	3.70

Table 5.2: Speed of wedge cuts versus GMI cuts.

limit. Out of the remaining 70 instances, 42 instances presented $z_{GMI} = z_W$. Table 5.1 presents the performance indicators for the remaining 28 instances.

It is well known that, when considering cuts from a single row of the simplex tableau, GMI cuts are very hard to outperform. Indeed, Fukasawa and Goycoolea [44] implemented an exact separator for *knapsack cuts*, a more general set of cuts that includes our wedge cuts, and tested it on the MIPLIB 3.0 and the MIPLIB 2003. Out of the 48 instances processed, on top of GMI, knapsack cuts increased the gap closure by more than 1 percentage point for only 8 instances, and more than 5 percentage points for only one instance. It should be noted, however, that 44 instances could not be processed due to time constraints in that study.

In our experiment, we obtained noticeably better results. Out of the 70 instances processed, wedge cuts contributed to more than 1% of the gap closure for 20 instances, and more than 5% for 13 instances. In fact, for 5 instances, the contribution from wedge cuts was greater than 25%. For two instances, `gmu-35-40` and `eil33-2`, the percentage was exceptionally high, at 99.26% and 71.97%, respectively. For the instance `gmu-35-40`, GMI cuts alone were only able to close 0.07% of the integrality gap, a negligible amount. The inclusion of wedge cuts improved that closure to 9.94%, which is noticeable. For the instance `mik-250-1-100-1`, although GMI cuts were able to reduce 53.52% of the gap, the inclusion of wedge cuts pushed that reduction to 73.38%, a significant improvement. Therefore, while our results indicate that, for most problems, wedge cuts do not seem to improve the integrality gap significantly when compared to GMI cuts alone, they might be useful for some particular classes of problems.

A side goal of our computational experiment to evaluate the efficiency of the enu-

meration algorithm presented in Section 5.2 In order to do that, we run the experiments again for the 10 instances for which wedge cuts presented the best performance, and we collected the additional statistics:

- CUTS-GMI and CUTS-W, the number of GMI cuts and wedge cuts, respectively, generated but not necessarily added to the relaxation,
- GMI-T and WEDGE-T, the average time needed to generate a single GMI cut and a single wedge cut, respectively, in milliseconds,

The results are presented on Table 5.2. On average, the time spent to generate one wedge cut was not much higher than the time spent to generate a single GMI cut. Note, however, that the number of wedge cuts generated, on all instances, was much larger than the number of GMI cuts, since we generate cuts for every tableau row, and for every integral non-basic variable. As a consequence, the total running time for many instances is prohibitively high, as seen in Table 5.1.

Next, instead of generating all possible wedge cuts for all possible combinations of rows and integral non-basic variables, we implement some heuristics to bring down the total number of cuts generated, and therefore the total running time of the algorithm. Our goal was to select a small subset of wedge cuts that, when added to the linear relaxation of the problem, yields approximately the same benefits, in terms of gap closure, as adding this entire family of cuts. We implemented three simple heuristics and tested their impact, both individually and combined, on the 10 instances for which wedge cuts presented the best performance.

Given a row and an integral non-basic variable, the first heuristic (Depth 5) limits the number wedge cuts generated to five. More precisely, we stop Algorithm 52 after five iterations, even if the stopping condition has not actually been reached. The motivation is that, as the vertices of the knapsack sets get farther away from f, the wedges become progressively thinner and, therefore, we would expect them to have progressively smaller practical impact. As shown in Table 5.3, this indeed turns out to be the case. After activating this heuristic, the gap closed is reduced by only 0.1 percentage points on average, a negligible amount, while the total running time is reduced by 7.4%. We conclude that this heuristic is moderately effective for most instances, although we do note that, for a few instances, the reduction on running time may not be worth the reduction in the gap closure (see instance `cov1075`, for example).

The second heuristic (100 Rows) limits to 100 the number of tableau rows selected to generate wedge cuts. More precisely, after solving the initial linear relaxation, we discard all, but the 100 tableau rows that have the most fractional right-hand side. This

is motivated by the fact that, if f is very close to an integral point, we would expect the number of generated wedges to be very small, and the generated cuts to be more numerically unstable. As shown in Table 5.3, this heuristic is very effective. While the gap closed is reduced by only 0.2 percentage points on average, the total running time is reduced by as much as 55%.

Given a tableau row, the third heuristic (100 Rays) limits to 100 the number of integral non-basic variables selected to generated wedge cuts, based on their reduced cost. The motivation for this heuristic is that, when a non-basic integral variable is selected to play the role of s_1, it tends to receive better cut coefficients. It is natural, therefore, to try to assign better coefficients to the variables that have the most impact on the objective value. Unfortunately, as Table 5.3 shows, while this heuristic was very successful at bringing down the total running time (a 98% reduction, on average), it also had considerable impact on the gap closure (a reduction of 2.1 percentage points). We conclude that this heuristic is too aggressive for most instances. Unfortunately, we were unable to derive a better heuristic for selecting the integral non-basic variables.

Finally, we evaluated the impact of the three previous heuristics combined. The results are shown in Table 5.3, under the header Combined. As we see, while the three heuristics, together, were very effective at reducing the total running time of the cut generating procedure, they unfortunately also had considerable negative impact on the gap closure. Despite this, we note that wedge cuts still presented considerable improvement for a small subset of instances (see `gmu-35-40` and `mik-250-1-100-1`), under very reasonable running times.

		All Combinations		Depth 5		100 Rows		100 Rays		Combined	
Instance	MIR-PERF	W-PERF	W-TIME	W-PERF	W-TIME	W-PERF	W-TIME	W-PERF	W-TIME	W-PERF	W-TIME
cov1075.pre	3.6	4.19	95	3.83	96	3.81	11	3.79	46	3.79	10
eil33-2.pre	4.28	15.27	8972	15.27	8652	15.27	9061	5.7	136	5.69	212
gmu-35-40.pre	0.07	9.94	122	9.92	104	9.94	116	8.14	16	8.14	27
mik-250-1-100-1.pre	53.52	73.48	14	73.48	13	73.48	14	70.88	10	70.91	14
mine-166-5.pre	6.57	7.58	1892	7.58	1805	7.27	71	6.89	119	6.81	14
mine-90-10.pre	12.4	14.51	970	14.51	895	13.07	46	13.03	74	12.48	11
n3div36.pre	16.38	18.85	268969	18.85	238709	18.85	257430	16.38	845	16.38	1218
neos-1337307.pre	3.76	6.45	32572	6.45	31355	6.44	1152	5.15	460	5.15	29
neos-686190.pre	4.61	5.51	33697	5.51	32521	5.39	10524	5.3	491	5.21	286
opm2-z7-s2.pre	0.62	0.98	284884	0.98	271012	0.98	2717	0.7	7152	0.7	134
average	10.6	15.7	63218.7	15.6	58516.2	15.5	28114.2	13.6	934.9	13.5	195.5

Table 5.3: Impact of cut selection heuristics.

Chapter 6

Conclusion

Since their introduction in the late 1950s by Gomory, cutting planes have become one of the main ingredients of modern MIP solvers. A direction that has received considerable attention in recent years is the generation of cutting planes from multi-row relaxations of the simplex tableau. In this thesis, we studied computational aspects necessary for the use of such multi-row cuts in practical settings. Next, we summarize our main contributions and discuss possible directions for future research.

Trivial Lifting in Two Dimensions. In Chapter 3, we developed a more practical method for performing the trivial lifting on two-row relaxations. For maximal lattice-free sets, we proved that our algorithm always terminates in constant time, and for the cases where the closed formula described in [19] is applicable, we showed that our algorithm required precisely the same number of evaluations to the function ψ as the closed formula. We also obtained an upper bound on the number of evaluations for non-maximal lattice-free sets, which depends on the lattice-width of the set and on its second covering minimum. Finally, we ran computational experiments, in order to evaluate the practical efficiency of our algorithm, and observed that the algorithm performs at least two orders of magnitude faster than the alternatives proposed in the literature.

An important direction for future research is obtaining similar algorithms for relaxations with three or more rows. The methods currently available for these relaxations can quickly become a bottleneck for any cut generation scheme. Even if a practical exact algorithm cannot be found for arbitrary dimensions, it would still be interesting to have good heuristics.

Intersection Cuts from the Infinity Norm. In Chapter 4, we introduced a new subset of multi-row intersection cuts based on the infinity norm. This subset is very small, which makes it attractive computationally, and its cuts are minimal, meaning

that they are not easily dominated. Unlike the other classes of multi-row intersection cuts in the literature, our cuts work for relaxations with arbitrary numbers of rows and they take into account the columns of the simplex tableau. We developed a geometric algorithm to generate them and ran computational experiments to evaluate their impact when used as cutting planes for solving real-world instances. We concluded that this subset of valid inequalities yields, in terms of gap closure, around 50% of the benefits of using all the valid inequalities for the continuous multi-row relaxation, but at a small fraction of the computational cost, and with a significantly smaller number of cuts.

We highlight the following directions for future work:

(1) One limitation of our cut generator is that it does not produce maximal lattice-free sets. While this has no effect on the coefficients of the continuous non-basic variables, it does negatively impact the coefficients for the integral non-basic variables. It would be interesting to have a practical algorithm for converting non-maximal lattice-free sets into maximal ones.

(2) In Section 4.3, we described several heuristics for selecting a subset of rows that would be used to generate infinity cuts. While these heuristics worked well for relaxations with two and three rows, they proved too slow for relaxations with four or more rows. More sophisticated heuristics are needed.

Intersection Cuts for Single-Row Relaxations. In Chapter 5, we revisited single-row cuts, with the objective of generating more valid inequalities; specifically, we were interested in generating cuts that were not dominated by Gomory Mixed-Integer cuts. Although single-row relaxations have been widely studied before, the novelty in our approach was that we used the framework of multi-row intersection cuts. We focused on a single-row relaxation of the simplex tableau where the integrality of a single non-basic variable is preserved. We developed an algorithm to enumerate all the facet-defining inequalities for this model, which leads to an upper bound on its split rank. We implemented all the methods proposed, and performed computational experiments using real-world instances. Our cut generation scheme proved to be very fast in practice. As far as the effectiveness of the cuts is concerned, expectations were limited, since we generate a subset of knapsack cuts, which have been shown by Fukasawa and Goycoolea [44] to be only slightly stronger in practice than the GMI cuts they generalize. For some instances, however, our cuts presented a clear improvement in terms of gap closed, over GMI cuts alone.

We highlight the following directions for future work:

(1) As discussed in Section 5.4, although our cut generator was very fast, in the sense that the time it took to generate a single wedge cut was not much higher than the time it took to generate a single GMI cut, we note that the number of wedge cuts generated was exceedingly high. A better way of deciding which wedge cuts to generate is needed.

(2) It would be interesting to generalize the results presented in Chapter 5 to two-row relaxations with a single integral non-basic variables, and to measure the practical performance of such cuts.

Bibliography

[1] Agostinho Agra and Miguel F. Constantino. Description of 2-integer continuous knapsack polyhedra. *Discrete Optimization*, 3(2):95 – 110, 2006. (Cited on pages 71 and 76.)

[2] Agostinho Agra and Miguel F. Constantino. Lifting two-integer knapsack inequalities. *Mathematical Programming*, 109(1):115–154, 2007. (Cited on pages 71 and 76.)

[3] Kent Andersen, Quentin Louveaux, and Robert Weismantel. Mixed-integer sets from two rows of two adjacent simplex bases. *Mathematical Programming*, 124:455–480, 2010. (Cited on page 3.)

[4] Kent Andersen, Quentin Louveaux, Robert Weismantel, and Laurence Wolsey. Cutting planes from two rows of a simplex tableau (extended version), 2006. Working Paper available on `http://orbi.ulg.ac.be/handle/2268/82794`. (Cited on page 76.)

[5] Kent Andersen, Quentin Louveaux, Robert Weismantel, and Laurence Wolsey. Inequalities from two rows of a simplex tableau. In Matteo Fischetti and David Williamson, editors, *Integer Programming and Combinatorial Optimization*, volume 4513 of *Lecture Notes in Computer Science*, pages 1–15. Springer Berlin / Heidelberg, 2007. (Cited on pages 3, 4, 9, 71, 72, 73, and 76.)

[6] Kent Andersen, Christian Wagner, and Robert Weismantel. On an analysis of the strength of mixed-integer cutting planes from multiple simplex tableau rows. *SIAM Journal on Optimization*, 20(2):967–982, 2009. (Cited on page 3.)

[7] David Applegate, Robert Bixby, William Cook, and Vasek Chvátal. On the solution of travelling salesman problems. *Documenta Mathematica*, Extra Volume ICM 1998(III):645–656, 1998. (Cited on page 2.)

[8] Alper Atamtürk and Deepak Rajan. Valid inequalities for mixed-integer knapsack from two-integer variable restrictions. Research Report BCOL.04.02, IEOR, University of California–Berkeley, December 2004. (Cited on pages 71 and 76.)

[9] Gennadiy Averkov and Amitabh Basu. Lifting properties of maximal lattice-free polyhedra. *Mathematical Programming*, 154(1-2):81–111, 2015. (Cited on pages 15, 18, 33, and 35.)

[10] Gennadiy Averkov, Amitabh Basu, and Joseph Paat. Approximation guarantees for intersection cut closures. *Submitted*, 2017. (Cited on page 3.)

[11] Gennadiy Averkov, Christian Wagner, and Robert Weismantel. Maximal lattice-free polyhedra: Finiteness and an explicit description in dimension three. *Mathematics of Operations Research*, 36(4):721–742, 2011. (Cited on pages 3 and 11.)

[12] Egon Balas. Intersection cuts – a new type of cutting planes for integer programming. *Operations Research*, 1(19):19–39, 1971. (Cited on pages 3, 12, 71, and 73.)

[13] Egon Balas. Integer programming. In *Encyclopedia of Optimization*, pages 1617–1624. Springer, 2008. (Cited on page 1.)

[14] Egon Balas, S. Ceria, G. Cornuejols, and N. Natraj. Gomory cuts revisited. *Operations Research Letters*, (19):1–9, 1996. (Cited on page 2.)

[15] Egon Balas and Robert G. Jeroslow. Strengthening cuts for mixed integer programs. *European Journal of Operational Research*, 4(4):224–234, 1980. (Cited on page 15.)

[16] Egon Balas and Anureet Saxena. Optimizing over the split closure. *Mathematical Programming*, 113(2):219–240, 2008. (Cited on page 40.)

[17] Egon Balas and Eitan Zemel. Facets of the knapsack polytope from minimal covers. *SIAM Journal on Applied Mathematics*, 34(1):119–148, 1978. (Cited on page 14.)

[18] Francisco Barahona, Martin Grötschel, Michael Jünger, and Gerhard Reinelt. An application of combinatorial optimization to statistical physics and circuit layout design. *Operations Research*, 36(3):493–513, 1988. (Cited on page 2.)

[19] Amitabh Basu, Pierre Bonami, Gérard Cornuéjols, and François Margot. Experiments with two-row cuts from degenerate tableaux. *INFORMS Journal on Computing*, 23:578–590, 2011. (Cited on pages 3, 4, 18, 31, 41, 42, and 95.)

[20] Amitabh Basu, Manoel Campêlo, Michele Conforti, Gérard Cornuéjols, and Giacomo Zambelli. Unique lifting of integer variables in minimal inequalities. *Mathematical Programming*, 141(1-2):561–576, 2013. (Cited on page 15.)

[21] Amitabh Basu, Michele Conforti, Gerard Cornuéjols, and Zambelli Giacomo. Minimal inequalities for an infinite relaxation of integer programs. *SIAM Journal on Discrete Mathematics*, 24(1):158–168, 2010. (Cited on page 74.)

[22] Amitabh Basu, Gerard Cornuejols, and Matthias Koppe. Unique minimal liftings for simplicial polytopes. *Mathematics of Operations Research*, 37(2):346–355, 2012. (Cited on page 15.)

[23] Amitabh Basu, Robert Hildebrand, and Matthias Köeppe. Algorithmic and complexity results for cutting planes derived from maximal lattice-free convex sets. *manuscript*, 2011. (Cited on page 76.)

[24] David E. Bell. A theorem concerning the integer lattice. *Studies in Applied Mathematics*, 56(2):187–188, 1977. (Cited on page 11.)

[25] Robert E. Bixby. A brief history of linear and mixed-integer programming computation. *Documenta Mathematica*, pages 107–121, 2012. (Cited on page 1.)

[26] Robert E. Bixby, Mary Fenelon, Zonghao Gu, Ed Rothberg, and Roland Wunderling. *MIP: Theory and Practice — Closing the Gap*, pages 19–49. Springer US, Boston, MA, 2000. (Cited on page 2.)

[27] Valentin Borozan and Gérard Cornuéjols. Minimal valid inequalities for integer constraints. *Mathematics of Operations Research*, 34(3):538–546, 2009. (Cited on pages 3, 10, and 11.)

[28] Michele Conforti, Gérard Cornuéjols, and Giacomo Zambelli. A geometric perspective on lifting. *Oper. Res.*, 59:569–577, May 2011. (Cited on pages 15, 70, 74, and 75.)

[29] Michele Conforti, Gérard Cornuéjols, and Giacomo Zambelli. *Integer programming*, volume 271. Springer, 2014. (Cited on page 7.)

[30] William Cook, Ravindran Kannan, and Alexander Schrijver. Chvátal closures for mixed integer programming problems. *Mathematical Programming*, 47(1-3):155–174, 1990. (Cited on page 4.)

[31] Gérard Cornuéjols, Yanjun Li, and Dieter Vandenbussche. K-cuts: A variation of Gomory mixed integer cuts from the LP tableau. *INFORMS Journal on Computing*, 15(4):385–396, 2003. (Cited on page 2.)

[32] Gérard Cornuéjols and François Margot. On the facets of mixed integer programs with two integer variables and two constraints. *Mathematical Programming*, 120(2):429–456, 2009. (Cited on pages 3 and 9.)

[33] Sanjeeb Dash, Marcos Goycoolea, and Oktay Günlük. Two-step mir inequalities for mixed integer programs. *INFORMS Journal on Computing*, 22(2):236–249, 2010. (Cited on pages 2, 69, and 88.)

[34] Santanu S. Dey, Andrea Lodi, Andrea Tramontani, and Laurence A. Wolsey. Experiments with two row tableau cuts. In Friedrich Eisenbrand and F. Bruce Shepherd, editors, *Integer Programming and Combinatorial Optimization, 14th International Conference, IPCO 2010, Lausanne, Switzerland, June 9-11, 2010. Proceedings*, volume 6080 of *Lecture Notes in Computer Science*, pages 424–437. Springer, 2010. (Cited on pages 4, 18, 40, and 42.)

[35] Santanu S. Dey, Andrea Lodi, Andrea Tramontani, and Laurence A. Wolsey. On the practical strength of two-row tableau cuts. *INFORMS Journal on Computing*, 26(2):222–237, 2014. (Cited on pages 4, 5, 10, 18, 40, and 42.)

[36] Santanu S. Dey and Quentin Louveaux. Split rank of triangle and quadrilateral inequalities. *Mathematics of Operations Research*, 36(3):432–461, 2011. (Cited on page 22.)

[37] Santanu S. Dey and Laurence A. Wolsey. *Lifting Integer Variables in Minimal Inequalities Corresponding to Lattice-Free Triangles*, pages 463–475. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. (Cited on page 10.)

[38] Santanu S. Dey and Laurence A. Wolsey. Constrained infinite group relaxations of MIPs. CORE Discussion Papers 2009033, Universite catholique de Louvain, Center for Operations Research and Econometrics (CORE), May 2009. (Cited on pages 73, 74, and 75.)

[39] Santanu S. Dey and Laurence A. Wolsey. Composite lifting of group inequalities and an application to two-row mixing inequalities. *Discrete Optimization*, 7(4):256 – 268, 2010. (Cited on pages 3 and 15.)

[40] Santanu S. Dey and Laurence A. Wolsey. Two row mixed-integer cuts via lifting. *Mathematical Programming*, 124:143–174, 2010. (Cited on pages 3, 5, 15, 17, and 69.)

[41] Jean-Paul Doignon. Convexity in cristallographical lattices. *Journal of Geometry*, 3(1):71–85, 1973. (Cited on page 11.)

[42] Daniel G. Espinoza. Computing with multi-row Gomory cuts. *Operations Research Letters*, 38(2):115 – 120, 2010. (Cited on pages 4, 17, 40, 41, and 42.)

[43] Matteo Fischetti and Cristiano Saturni. *Mixed-Integer Cuts from Cyclic Groups*, pages 1–11. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005. (Cited on page 2.)

[44] Ricardo Fukasawa and Marcos Goycoolea. On the exact separation of mixed integer knapsack cuts. *Mathematical Programming*, 128(1-2):19–41, 2011. (Cited on pages 2, 69, 91, and 96.)

[45] Ralph E. Gomory. Outline of an algorithm for integer solutions to linear programs. *Bulletin of the American Mathematical Society*, 64(5):275–278, 1958. (Cited on page 2.)

[46] Ralph E. Gomory. Some polyhedra related to combinatorial problems. *Linear Algebra and its Applications*, 2(4):451–558, 1969. (Cited on pages 2, 9, and 14.)

[47] Ralph E. Gomory and Ellis L. Johnson. Some continuous functions related to corner polyhedra, part I. *Mathematical Programming*, 3:23–85, 1972. (Cited on pages 2 and 15.)

[48] Ralph E. Gomory and Ellis L. Johnson. Some continuous functions related to corner polyhedra, part II. *Mathematical Programming*, 3(1):359–389, 1972. (Cited on page 2.)

[49] Torbjrn Granlund. *GNU MP 6.0 Multiple Precision Arithmetic Library*. Samurai Media Limited, 2015. (Cited on page 89.)

[50] Martin Grötschel, Michael Jünger, and Gerhard Reinelt. A cutting plane algorithm for the linear ordering problem. *Operations Research*, 32(6):1195–1220, 1984. (Cited on page 2.)

[51] Martin Grötschel, Alexander Martin, and Robert Weismantel. Packing steiner trees: A cutting plane algorithm and computational results. *Mathematical Programming*, 72(2):125–145, 1996. (Cited on page 2.)

[52] Zonghao Gu, George L. Nemhauser, and Martin W. P. Savelsbergh. Sequence independent lifting in mixed integer programming. *Journal of Combinatorial Optimization*, 4:109–129, 2000. (Cited on page 15.)

[53] Warwick Harvey. Computing two-dimensional integer hulls. *SIAM Journal on Computing*, 28(6):2285–2299, 1999. (Cited on page 76.)

[54] Daniel S. Hirschberg and Chung K. Wong. A polynomial-time algorithm for the knapsack problem with two variables. *Journal of the ACM*, 23(1):147–154, January 1976. (Cited on pages 71 and 76.)

[55] C.A.J. Hurkens. Blowing up convex sets in the plane. *Linear Algebra and its Applications*, 134:121 – 128, 1990. (Cited on page 23.)

[56] Ellis L. Johnson. On the group problem for mixed integer programming. *Mathematical Programming Study*, 2:137–179, 1974. (Cited on pages 2, 14, and 15.)

[57] Ravi Kannan and László Lovász. *Covering minima and lattice point free convex bodies*, pages 193–213. Springer Berlin Heidelberg, Berlin, Heidelberg, 1986. (Cited on page 28.)

[58] Thorsten Koch, Tobias Achterberg, Erling Andersen, Oliver Bastert, Timo Berthold, Robert E. Bixby, Emilie Danna, Gerald Gamrath, Ambros M. Gleixner, Stefan Heinz, Andrea Lodi, Hans Mittelmann, Ted Ralphs, Domenico Salvagnin, Daniel E. Steffy, and Kati Wolter. MIPLIB 2010. *Mathematical Programming Computation*, 3(2):103–163, 2011. (Cited on page 36.)

[59] Ailsa H. Land and Alison G. Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497–520, 1960. (Cited on page 1.)

[60] Quentin Louveaux and Laurent Poirrier. An algorithm for the separation of two-row cuts. *Mathematical Programming*, 143(1-2):111–146, 2014. (Cited on pages 36, 41, and 56.)

[61] Quentin Louveaux, Laurent Poirrier, and Domenico Salvagnin. The strength of multi-row models. *Mathematical Programming Computation*, 7(2):113–148, 2015. (Cited on pages 3, 4, 41, 56, 57, and 76.)

[62] László Lovász. Geometry of numbers and integer programming. *Mathematical Programming: Recent Developments and Applications*, pages 177–210, 1989. (Cited on page 10.)

[63] François Margot. Testing cut generators for mixed-integer linear programming. *Mathematical Programming Computation*, 1(1):69–95, 2009. (Cited on page 41.)

[64] Felipe Serrano Musalem. Some experiments on separation with multi-row cuts. Master's thesis, Universidad de Chile, 2012. (Cited on pages 4, 18, 40, and 42.)

[65] Timm Oertel, Christian Wagner, and Robert Weismantel. Convex integer minimization in fixed dimension. Available online at http://arxiv.org/pdf/1203.4175v1.pdf, 2012. (Cited on page 18.)

[66] Manfred W. Padberg. On the facial structure of set packing polyhedra. *Mathematical programming*, 5(1):199–215, 1973. (Cited on page 14.)

[67] Laurent Poirrier and Álinson S. Xavier. Benchmark for multi-row intersection cuts. https://github.com/iSoron/multirow, Accessed May 15, 2017. (Cited on pages 34, 56, and 88.)

[68] Herbert E. Scarf. An observation on the structure of production sets with indivisibilities. *Proceedings of the National Academy of Sciences*, 74(9):3637–3641, 1977. (Cited on page 11.)

[69] Laurence A. Wolsey. Facets and strong valid inequalities for integer programs. *Operations Research*, 24(2):367–372, 1976. (Cited on page 14.)

[70] Emre Yamangil. *Valid Inequalities for Mixed-Integer Linear Programming Problems*. PhD thesis, Rutgers, The State University of New Jersey, 2015. (Cited on pages 40 and 42.)

[71] Giacomo Zambelli. On degenerate multi-row gomory cuts. *Operations Research Letters*, 37(1):21–22, 2009. (Cited on page 3.)

[72] Eitan Zemel. Lifting the facets of zero–one polytopes. *Mathematical Programming*, 15(1):268–277, 1978. (Cited on page 14.)