On the Decomposition of Test Sets:

Building Blocks, Connection Sets, and Algorithms

Von der Fakultät der Naturwissenschaften der
Gerhard-Mercator-Universität Duisburg
zur Erlangung des akademischen Grades eines

Dr. rer. nat.

genehmigte Dissertation
von

Raymond Hemmecke
aus
Kölleda

Referent: Prof. Dr. Rüdiger Schultz
Koreferent: Prof. Dr. Rekha R. Thomas

Tag der mündlichen Prüfung: 24. September 2001

Contents

Acknowledgements

First I want to thank my advisor, Rüdiger Schultz, for providing excellent working and studying conditions both at the University of Leipzig and at the University of Duisburg. His advice and support strongly influenced my academic development, in teaching, in giving talks and lectures, and in writing articles.

Moreover, I want to thank Rekha Thomas (University of Washington, Seattle), for pointing my attention to Diane Maclagan's theorem ([32, 33]) which finally proved finiteness of the set $\mathcal{H}_\infty$ also for two-stage stochastic integer programs, as presented in Chapter 3.

In this respect, I am clearly very grateful to Diane Maclagan (Stanford University) for proving her theorem on finite sequences of monomial ideals.

Moreover, I would like to thank Jesus de Loera (University of California at Davis), Herbert Scarf (Yale University, New Haven), Bernd Sturmfels (University of California at Berkley), and Rekha Thomas for their support and many interesting discussions.

I am grateful to Jack Graver (Syracuse University) for giving access to the unpublished manuscript [17].

I want to thank my brother, Ralf Hemmecke (RISC Linz) for his help in optimizing the final C-code of the program MLP, that has been developed in this scope of this thesis.

Moreover, I want to thank Holger Traczinski (University of Duisburg) for fruitful discussions on the fast computation of the Minkowski sum of two lists of vectors, which led to the improvement presented in Subsection 7.3.2

Last but not least, I gratefully acknowledge the support by the Schwerpunktprogramm "Echtzeit-Optimierung großer Systeme" of the Deutsche Forschungsgemeinschaft.

Introduction

Many important classes of optimization problems arising in practical applications can be modeled as (usually) large mixed-integer linear programming problems. Therefore, much research both on theory and on algorithmic aspects has been done. Powerful algorithms like branch-and-bound or cutting-plane methods have been developed, improved, and implemented into fast optimization software in order to solve bigger and bigger problems. Algorithmic improvements, however, seem to become smaller and harder to achieve. Therefore, it is worth studying other algorithmic ideas.

During the last few years, there has been a renewed interest in a different solution approach. These so-called primal methods are based on a simple augmentation procedure that repeatedly improves a given feasible solution as long as this solution is not optimal. A prominent example of such an algorithm is the simplex algorithm that moves along edges of the feasible region from vertex to vertex in order to find an optimal solution to a given linear program. In the general augmentation algorithm, however, we are allowed to make augmentation steps also through the interior of the set of feasible solutions.

Although primal methods have not yet been proven applicable to solve optimization problems of interesting (practical) sizes, some promising ideas and computational experience have been reported very recently [20, 21, 29].

The major ingredient of the augmentation algorithm is the computation of improving directions to given non-optimal feasible solutions. This problem can be solved by the help of test sets.

Test sets are (finite) sets of vectors (or directions) with the property that there is always an (improving) direction in this set that can be used to augment a given non-optimal feasible solution of our problem. In this work we will describe a particularly nice test set, the Graver test set or Graver basis, that was introduced by Graver [19] in 1975. We will present novel algorithms to compute these sets in the linear and the integer linear situations, together with a finite augmentation procedure for the mixed-integer linear case, where test sets need not be finite.

These algorithms, however, are not the main bottleneck of the presented test set approach. Test sets turned out to be quite large already for small problems and need not even be finite in the mixed-integer situation. Therefore, in order to overcome this problem, we move our attention from algorithmic improvements to size reduction of test sets. Thus, the major and novel concept of this thesis is the decomposition of test set vectors into smaller building blocks. We present algorithms to compute these much fewer building blocks together with the details on how to find improving vectors for the augmentation algorithm that solves our optimization problem at hand. All algorithms work entirely on the building block level, that is, the full test set is not computed. We apply this decomposition idea specifically to two- and multi-stage stochastic programs, where the special structure of the problem matrices admits a particularly effective decomposition. We include some first computational experience where our decomposition approach is superior to the existing algorithms (computer codes). Thereafter, we extend the decomposition idea to test sets for problems with arbitrary matrices.

After this general introduction let us become more specific. In this work we deal with the solution of mixed-integer linear programs via a simple augmentation algorithm. For given $d = d_z + d_q$, $A \in \mathbb{Z}^{l \times d}$, $c \in \mathbb{R}^d$, and $b \in \mathbb{R}^l$, let

$$(\mathrm{P})_{c,b}: \qquad \min\{c^\intercal z : Az = b,\ z \in \mathbb{X}_+\}$$

be the family of mixed-integer linear optimization problems as $c \in \mathbb{R}^d$ and $b \in \mathbb{R}^l$ vary. Herein, $\mathbb{X} = \mathbb{Z}^{d_z} \times \mathbb{R}^{d_q}$, and $\mathbb{X}_+$ denotes the corresponding non-negative orthant. As usual, $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$ denote the integers, rationals, and reals, respectively. Instead of $(\mathrm{P})_{c,b}$ we write $(\mathrm{LP})_{c,b}$ if $\mathbb{X} = \mathbb{R}^d$ and $(\mathrm{IP})_{c,b}$ if $\mathbb{X} = \mathbb{Z}^d$. By abuse of notation we refer to subsets of the problem family $(\mathrm{P})_{c,b}$ as $(\mathrm{P})_{c,b}$ as well, but state which data is kept fixed and which is allowed to vary. Thus, a single instance, that is where b and c are given, is also denoted by $(\mathrm{P})_{c,b}$.

Primal methods are based on repeated augmentation of feasible solutions to $(\mathrm{P})_{c,b}$ to optimality. Strongly related to this augmentation scheme is the notion of a test set.

Definition 0.0.1 *(Test Set)*

A set $\mathcal{T} \subseteq \mathbb{R}^d$ is called a test set for $(P)_{c,b}$ if

1. *$c^\intercal t > 0$ for all $t \in \mathcal{T}$, and*
2. *for every $b \in \mathbb{R}^l$ and for every non-optimal feasible point z_0 of $(P)_{c,b}$ there exist a vector $t \in \mathcal{T}$ and a scalar $\alpha > 0$ such that $z_0 - \alpha t$ is feasible.*

A vector $t \in \mathcal{T}$ satisfying these two conditions is called an improving vector or an improving direction.

A set is called a universal test set for $(P)_{c,b}$ if it contains a test set for $(P)_{c,b}$ for every cost vector $c \in \mathbb{R}^d$.

In contrast to the common definition of test sets we do not impose finiteness on $\mathcal{T}$. This allows a treatment of test sets for mixed-integer programs which need not be finite in general (see for example Cook et al. [14]).

Once a finite (universal) test set $\mathcal{T}$ for and a feasible solution z_0 to $(\mathrm{P})_{c,b}$ are available, the following augmentation algorithm can be employed in order to solve the optimization problem $(\mathrm{P})_{c,b}$.

Algorithm 0.0.2 *(Augmentation Algorithm)*

Input: $\mathbb{X}$, *a matrix* A, *a finite test set* $\mathcal{T}$ *for* $(P)_{c,b}$, *a cost vector* c, *a feasible solution* z_0 *to* $(P)_{c,b}$

Output: an optimum $z_{\min}$ *of* $(P)_{c,b}$

while there is $t \in \mathcal{T}$ *with* $c^\intercal t > 0$ *and a scalar* $\alpha > 0$ *such that* $z_0 - \alpha t$ *is feasible do*

$z_0 := z_0 - \alpha t$, *where* α *is chosen maximal such that* $z_0 - \alpha t$ *is still feasible*

return z_0

In 1975, Graver [19] introduced finite universal test sets for linear programs (LP) and for integer linear programs (IP). However, he provided no algorithm to compute them. He also presented a similar test set for linear mixed-integer programs (MIP) and already pointed out that these sets need not be finite in general [17]. Graver test sets can be characterized by the following simple property.

Definition 0.0.3 *(Positive Sum Property)*

A set G has the positive sum property with respect to $S \subseteq \mathbb{R}^d$ if $G \subseteq S$ and if any non-zero $v \in S$ can be written as a finite linear combination $v = \sum \alpha_i g_i$ with

- *$g_i \in G$, $\alpha_i > 0$, $\alpha_i g_i \in S$, and*
- *for all i, g_i and v belong to the same orthant, that is, $g_i^{(k)} v^{(k)} \geq 0$ for every component $k = 1, \ldots, d$.*

Graver test sets are inclusion minimal sets having the positive sum property with respect to the set $\ker_{\mathbb{X}}(A) := \{v \in \mathbb{X} : Av = 0\}$, the (mixed-integer) kernel of A.

Already Graver [19] showed that the positive sum property implies the universal test set property. For completeness, we include a proof for this important fact in Section 1.1.

In 1981, Scarf [43, 44, 45] presented a finite test set for $(\mathrm{IP})_{c,b}$ for fixed c, the neighbors of the origin. Herein, however, the matrix A had to fulfill strong non-degenericity assumptions. At that time, Scarf, too, could not provide an algorithm for the computation of his set.

Only in 1991, Conti and Traverso [13] succeeded in computing a test set for $(\mathrm{IP})_{c,b}$ for fixed c by transforming the problem into an algebraic question about polynomial ideals and by using Buchberger's Gröbner basis algorithm [6] in order to solve it. The algorithm by Conti and Traverso, however, had a large computational overhead, because it used many additional variables. Since then, however, a lot of work has been done to improve its performance. By fundamental algebraic relations the problem was reduced to the saturation of a certain polynomial ideal with respect to all occurring variables and to a final Gröbner basis computation. Both algorithms involve only the original n variables. Later it was found that already saturation with respect to a subset of variables is sufficient. For more details see for example Sturmfels [48].

Buchberger's algorithm being a field of active research in computer algebra, test set computation has also benefited from efficient implementations and improvements achieved in that area (see Bigatti et al. [5], or Hosten and Sturmfels [25]).

The first geometrical interpretation of the Conti-Traverso algorithm was given by Thomas [52]. This has initiated further work on geometric algorithms for the computation of test sets, see Cornuéjols et al. [15] and Urbaniak et al. [55].

Structural similarities of Graver test sets in LP and IP were elaborated by Sturmfels and Thomas [49]. In [57], Weismantel gives a survey on all currently known test sets in IP together with their computation.

Some recent work has been done on MIP test sets. Following a geometric approach, Köppe et al. [23, 28] obtained a finiteness result similar to the one we obtain by an algebraic argument in Chapter 2. This allows a finite algorithmic treatment of mixed-integer linear programs via the Augmentation Algorithm 0.0.2.

Regardless of these algorithmic improvements, it turned out that test sets tend to be huge already for comparatively small problem matrices, having 100 columns, say. To overcome this problem, truncated test sets were considered. A truncated test set has to give improving vectors to non-optimal feasible solutions only for a subset of all possible problems $(\mathrm{IP})_{c,b}$ [53, 55], or even only for a fixed problem [30]. In Section 2.2 we will introduce truncated Graver test sets where fixed upper bounds on variables

are used to truncate the test set to a sufficient subset. Although the presented ideas lead to drastic improvements in the speed of the algorithms, the resulting test sets remain huge for small problems.

This enormous size of test sets led us to the question, whether one could use the structure of the problem matrix in order to encode the test set more efficiently. For this, we turned our attention to two-stage stochastic optimization problems. The two-stage mixed-integer linear stochastic program is the optimization problem

$$\min\{h^\intercal x + Q(x) : Ax = a, x \in X\} \tag{1}$$

where

$$Q(x) := \int_{\mathbb{R}^s} \Phi(\xi - Tx)\mu(d\xi) \tag{2}$$

and

$$\Phi(\tau) := \min\{q^\intercal y : Wy = \tau, y \in Y\}. \tag{3}$$

Here, $X \subseteq \mathbb{R}^m_+$ and $Y \subseteq \mathbb{R}^n_+$ denote the non-negative orthants, possibly involving integer requirements to variables. The measure μ is a Borel probability measure on $\mathbb{R}^s$, and all the remaining data have conformal dimensions.

The model (1)-(3) arises in optimization under uncertainty. Given an optimization problem with random data where parts of the decisions (the first-stage variables x) have to be taken before and parts (the second-stage variables y) are taken after the realizations of the random data are known, the purpose of (1)-(3) is to minimize the sum of the direct costs $h^\intercal x$ and the expected costs of optimal decisions in the second stage. The model has a multi-stage extension where a multi-stage process of alternating decision and observation replaces the two-stage process assumed above. For further details on the modeling background we refer to [3, 26, 38].

Under mild assumptions, all ingredients in the above model are well defined [46]. Algorithmically, (1)-(3) provides some challenges, since the integral behind $Q(x)$ is multidimensional and its integrand is given only implicitly. Due to the fact that the model behaves stable when perturbing the underlying probability measure [3, 26, 38, 46], computations, almost exclusively, work with discrete measures μ, tacitly assuming that, if necessary, continuous measures have been approximated by discrete ones beforehand. Therefore, our algorithmic considerations will rest on the assumption that μ is a discrete probability measure with finitely many realizations (or scenarios) $\xi^1, \ldots, \xi^N$, and probabilities $\pi^1, \ldots, \pi^N$. Then (1)-(3) is equivalent to the mixed-integer linear program

$$\begin{array}{rrcl} \min\{h^\intercal x + \sum_{\nu=1}^N \pi^\nu q^\intercal y^\nu : & Ax & = & a, \quad x \in X, \\ & Tx + Wy^\nu & = & \xi^\nu, \ y^\nu \in Y, \quad \nu = 1, \ldots, N\}. \end{array} \tag{4}$$

The number N of scenarios being big in general, (4) is large-scale and not amenable to general purpose mixed-integer linear programming solvers. This has motivated research into decomposition algorithms. The latter have a long tradition in stochastic linear programming without integer requirements [3, 24, 26, 38, 41]. Then models enjoy convexity properties, and algorithmic developments can be based on subgradient techniques from convex minimization and convex duality, for instance. With integer requirements in (4) these convexity properties are lost, leading to substantially different decomposition approaches (see [27] for a recent survey). So far, two main lines of research have been pursued in decomposition of stochastic mixed-integer programs: (primal) decomposition by rows and (dual) decomposition by columns of the constraint matrix of (4) whose block-angular structure is depicted in Figure 1.

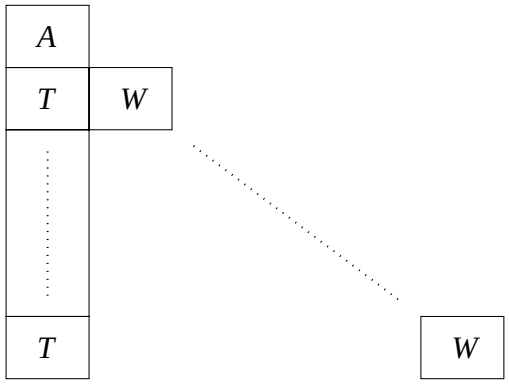

Figure 1: Constraints matrix structure of (4)

In primal decomposition, the variable x is iterated in an outer master problem, and advantage is taken of the fact that, for fixed x, the minimization with respect to $(y^1, \ldots, y^N)$ can be carried out separately for each y^ν. The major difficulty with this approach in the (mixed-) integer situation is the complicated structure of the master problem whose objective function is non-convex and discontinuous, in fact lower semicontinuous. Research has been devoted to studying cutting planes for the master problem that are derived via subadditive duality [8, 11, 12], to solving the master problem by enumeration and bounding while handling the second-stage problems by methods exploiting problem similarities [47], and to extending the latter into a branch-and-bound method in the space of first-stage variables [1].

Dual decomposition methods start from an explicit (linear) representation of non-anticipativity constraints. These constraints are inherent to (1)-(3) and (4). They state that first-stage decisions must not depend on future observations. In (1)-(3) and (4) this is modeled implicitly by the independence of the variable x on the random vector ξ in (1)-(3) and on the scenarios $\xi^\nu, \nu = 1, \ldots, N$, in (4), respectively.

Decomposition then is achieved by Lagrangian relaxation of the non-anticipativity constraints. In [31] the framework of progressive hedging [39] is adopted where Augmented Lagrangians lead to decomposition into quadratic mixed-integer programs which are tackled by tabu search. In [9, 10] the standard Lagrangian is employed. This leads to linear mixed-integer subproblems that can be solved by general-purpose or custom-made mixed-integer linear programming software. On top of the Lagrangian relaxation, a branch-and-bound scheme in the space of first-stage variables is placed, providing quality bounds in the course of the iteration.

Another mode of decomposition that has been applied successfully mainly to industrial problems in power optimization relies on problem dependent decoupling into submodels which are still two- or even multi-stage stochastic integer programs but amenable to specialized algorithms based on dynamic programming or network flow, see for instance [34, 35, 51].

A common feature of all the algorithms discussed above is decomposition of the problem itself. In this thesis we adopt an entirely different viewpoint. Instead of decomposition of the problem (4), we study decomposition of its Graver test set. Decomposition of the Graver test set of (4) will lead us to an augmentation algorithm that, to the best of our knowledge, so far has no counterpart in the existing stochastic programming literature. In particular, we have observed that, once an algorithmic bottleneck determined by the sizes of A, T, and W is passed, the algorithmic effort grows only mildly with the number N of scenarios. In contrast, the algorithms discussed above have been observed to be quite sensitive to that number.

We extend this decomposition idea to multi-stage stochastic programs and to problems with arbitrary problem matrix. To this end, we introduce the concept of connection vectors that couple the building blocks of (Graver) test set vectors. This yields finite sets of connection vectors, so-called connection sets, even in the mixed-integer situation, where test sets need not be finite in general.

This thesis is structured as follows.

Chapters 1 and 2 lay the notational and algorithmic basis to our test set decomposition approaches to two- and multi-stage stochastic programs and to arbitrary (mixed-integer and integer) linear programs as presented in Chapters 3, 4, and 5.

In Chapter 1 we present the positive sum property and show that this property ensures the universal test set property. This fact has been proved already by Graver [19]. However, it is also this positive sum property that directs us to completion procedures [7], a well-known and powerful algorithmic concept from computational algebra: step by step an initial set of vectors is completed with respect to the positive sum property

by adding new vectors to the set as long as necessary. This leads to novel algorithms to compute LP and IP Graver test sets (see Chapter 2) that work entirely in the kernel of the problem matrix. It turned out, however, that the IP algorithm had been already presented in the disguised form of a d-dimensional Euclidean algorithm in a research report by Pottier [37]. We present a slightly more general IP algorithm that computes minimal points (with respect to a certain property) not only in $\ker_{\mathbb{Z}^d}(A)$ but in general integer lattices $\Lambda \subseteq \mathbb{Z}^d$. In this way we can extend the concept of IP Graver bases also to optimization problems

$$(\mathrm{MOD})_{c,b,\bar{b}}: \qquad \min\{c^\intercal z : Az = b, \bar{A}z \equiv \bar{b} \pmod{p}, z \in \mathbb{Z}^d_+\},$$

where $\bar{A}z \equiv \bar{b} \pmod{p}$ abbreviates the relations $\bar{a}_i z \equiv \bar{b}_i \pmod{p_i}$ for $p_i \in \mathbb{Z}_+$, $i = 1, \ldots, \bar{l}$. These problems have a close connection for example to the "group problem in integer programming" [42] and to the integer Fourier-Motzkin elimination [58, 59, 60], respectively. Another algorithm for the computation of LP Graver test sets can be found for example in [48].

In Chapter 2 we introduce Graver test sets for LP, IP, and MIP as inclusion minimal sets that have the positive sum property with respect to $\ker_{\mathbb{X}}(A)$. Then we treat each case separately in more detail. In the LP and in the IP cases we relate the Graver test set to the circuits of the matrix A and to Hilbert bases of certain pointed rational cones, respectively.

Right after the section on IP Graver bases we include a section on truncated Graver test sets for the family of problems $\min\{c^\intercal z : Az = b, 0 \leq z \leq u, z \in \mathbb{Z}^d\}$ as only $c \in \mathbb{R}^d$ and $b \in \mathbb{R}^l$ vary. Here, fixed finite upper bounds on z are used to truncate the Graver test set for $\min\{c^\intercal z : Az = b, z \in \mathbb{Z}^d_+\}$ to a sufficient subset and to speed up the direct computation of this truncated set. The presented algorithm, however, is more general and computes all non-zero minimal integer vectors in the set $\{z : Az = 0, l \leq z \leq u, z \in \mathbb{Z}^d\}$, $l \leq 0 \leq u$, where a vector v is called minimal if there is no vector w in this set such that v and w belong to the same orthant of $\mathbb{R}^d$ and such that the components of w are not greater in absolute value than the corresponding components of v. Again, the same algorithmic pattern as for general Graver test set computations, a completion procedure, is used. For special choices of l and u this includes algorithmic solutions to the following problems:

- For $l = 0$ and $u = 1$ (component-wise) the algorithm computes (partial) test sets for combinatorial optimization problems.
- For $l = -u$ the algorithm computes the desired truncated Graver test set.

- For $l = -\infty$ and $u = +\infty$ the algorithm is similar to Algorithm 2.1.7 in [54] and computes the IP Graver test set corresponding to A. However, it is much slower than Algorithm 1.3.1 (Section 1.3), since the computation is done in a higher dimensional space.

- For $l = 0$ and $u = +\infty$ the algorithm computes the unique minimal Hilbert basis for the pointed rational cone $\{z : Az = 0, z \in \mathbb{R}^d_+\}$.

The notion of a truncated Graver test set already appeared in a paper by Thomas and Weismantel [53]. They define a partial order on the right-hand side (b, u) in order to truncate test sets for problems of type $\min\{c^\intercal z : Az = b, 0 \leq z \leq u, z \in \mathbb{Z}^d\}$. Our algorithm to compute a truncated Graver test set could be deduced from their approach by defining their partial order not on all components of the right-hand side (b, u) but only on the components of u, the upper bounds on z. Our approach uses a simpler algebraic machinery, whereas the one by Thomas and Weismantel is more general.

Next, we introduce MIP Graver test sets. We present an example similar to the one by Cook et al. [14], which implies that MIP test sets need not be finite in general. In [14] the existence of finite MIP test sets is shown if considerations are restricted to a smaller family of optimization problems. In contrast to this approach, we introduce a finite set of integer vectors that is no test set by Definition 0.0.1. However, we will present algorithms to compute these finitely many integer vectors and to compute an improving direction for a given non-optimal solution. Thus, the Augmentation Algorithm 0.0.2 can be employed in the MIP case, too.

In the following two sections we deal with the two main questions concerning the Augmentation Algorithm 0.0.2: does the augmentation algorithm always terminate and how do we find an initial feasible solution for its input? First, in Section 2.5, we discuss termination of the Augmentation Algorithm 0.0.2. We demonstrate that particularly in the LP case some caution is appropriate to avoid zig-zagging (even to non-optimal solutions) and we present a strategy which ensures termination. Then, we deal with feasible solutions. A test set has to give improving directions for any right-hand side b and hence cannot depend on the specific choice of b. The only and, if integrality constraints on variables are present, usually hard step specific to b is to find some feasible solution of the given problem. Although there are algorithmic alternatives, we show in Section 2.6 that universal test sets can be used to solve this feasibility problem as well. Again, some care has to be taken in the LP case to ensure termination of this procedure.

In Chapter 3 we present our decomposition approach for two-stage stochastic programming. We introduce building blocks that can be employed to construct test set vectors for two-stage stochastic linear and integer linear programs. Existence of a finite set $\mathcal{H}_\infty$ is proved, that contains all the building blocks of test set vectors of (4) for arbitrary objective function and arbitrary right-hand side vectors, and, most importantly, for arbitrary number N of scenarios. Then we develop a finite algorithm that computes $\mathcal{H}_\infty$ which again is based on a completion procedure. This algorithm entirely works at the building block level. No explicit information on test set vectors is needed. The set $\mathcal{H}_\infty$ is employed by a finite augmentation procedure for solving (4) that, again, is entirely working at the building block level. The chapter is completed by a discussion of stochastic programs with simple recourse and with a section on initial computational experience in the integer situation.

In Chapter 4 we show how the above decomposition approach for two-stage stochastic programs together with all necessary notions can be generalized to the multi-stage situation. To this end, however, we will present the details for the 3-stage case only. Again, we define a set $\mathcal{H}_\infty$ of building blocks that does not depend on the number of scenarios. Moreover, we present an algorithm, again a completion procedure, which, upon termination, returns the set $\mathcal{H}_\infty$. We prove termination of the algorithm, and thus finiteness of $\mathcal{H}_\infty$, for the LP case. The same question, however, is still open in the IP situation. Again, the set $\mathcal{H}_\infty$ is employed by a finite augmentation procedure for solving the multi-stage problem. Note that also this procedure works entirely at the building block level.

The treatment of the 3-stage case should make clear, how to define the decomposition, the notions, and the algorithms for the higher multi-stage situations. Also the proofs follow analogous counting or construction arguments.

In Chapter 5 we apply our decomposition method to (mixed-integer and integer) linear optimization problems with arbitrary problem matrix A. We split the matrix A into $(A_1|A_2)$ and each test set vector v accordingly into (v_1, v_2). We call v_1 and v_2 the building blocks of v which are connected via the vector $A_1v_1 = -A_2v_2$. The collection of all these connection vectors A_1v_1, the connection set, may be much smaller than the test set it corresponds to. Moreover, we show that the knowledge of the connection set is already enough to solve the original optimization problem. More importantly, even if the full test set is known, the problem under consideration may be solved much faster by first extracting the connection set and then solving the problem with the help of these connection vectors only.

We look again at Graver bases for LP, IP, and MIP, and at our decomposition approach for two-stage stochastic programs (Section 5.2) but this time from the point

of view of connection sets. We show how an improving vector can efficiently be reconstructed from the connection set in all these cases which makes the application of the Augmentation Algorithm 0.0.2 possible.

Finally, we present an algorithm to compute connection sets also for two-stage stochastic mixed-integer linear programs where first-stage and second-stage continuous variables are not coupled by a row of $(T|W)$. Surprisingly, the connection set corresponding to $\mathcal{H}_\infty$ in this case turns out to be finite.

In Chapter 6 we collect some open problems which arose in the previous chapters. These problems either deal with structural questions (finiteness of $\mathcal{H}_\infty$ for multi-stage stochastic integer programs) or with computational improvements.

In Chapter 7 we present implementational details of the computer code MLP [22] which has been developed during the last three years within the scope of this thesis. MLP computes (truncated) IP Graver test sets, Hilbert bases of certain pointed rational cones, and the set $\mathcal{H}_\infty$ for two-stage stochastic integer programs. In this chapter we also present a novel algorithmic improvement to IP Graver bases computations, if the kernel of the problem matrix has a particularly nice generating set of the form $\{(e_1, -Ae_1), ..., (e_d, -Ae_d)\}$, where $e_1, \ldots, e_d$ denote the unit vectors in $\mathbb{R}^d$. The same improvement can be employed for the computation of Hilbert bases of $\ker_{\mathbb{R}^d} \cap \mathbb{R}^d_+$.

Finally, we have a collection of the notation used in this thesis and a conclusions chapter.

Chapter 1

The Positive Sum Property

In this chapter we introduce the positive sum property and show that it implies the universal test set property. Afterwards we present two criteria which allow a finite test whether a given symmetric set G has the positive sum property with respect to an integer lattice $\Lambda \subseteq \mathbb{Z}^d$, for example $\ker_{\mathbb{Z}^d}(A)$, or with respect to $\ker_{\mathbb{R}^d}(A)$. Herein, a set G is called symmetric if $v \in G$ implies $-v \in G$.

These two criteria immediately lead us to the concept of a completion procedure: a given (symmetric) generating set G of an integer lattice Λ over $\mathbb{Z}$ or of $\ker_{\mathbb{R}^d}(A)$ over $\mathbb{R}$ is completed with respect to the positive sum property by adding new elements to G as long as necessary. Once the completion procedure terminates, the two criteria below already imply its correctness, that is, the set of vectors returned has indeed the positive sum property with respect to Λ or $\ker_{\mathbb{R}^d}(A)$.

1.1 Positive Sum Property implies Universal Test Set Property

To simplify the subsequent proofs it is convenient to introduce the following relation.

Definition 1.1.1 *We define the relation $\sqsubseteq$ on $\mathbb{R}^d$ by $u \sqsubseteq v$ if $u^{(j)}v^{(j)} \geq 0$ and $|u^{(j)}| \leq |v^{(j)}|$ for all components $j = 1, \ldots, d$, that is, u belongs to the same orthant as v and its components are not greater in absolute value than the corresponding components of v.*

Clearly, if $u \sqsubseteq v$ then also $v - u \sqsubseteq v$. The following lemma was already proved by Graver [19]. We include a proof because of its importance.

Lemma 1.1.2 *(Positive Sum Property implies Universal Test Set Property)*

If G has the positive sum property with respect to $\ker_{\mathbb{X}}(A)$ then G is a universal test set for $(P)_{c,b}$.

Proof. Given $b \in \mathbb{R}^l$, $c \in \mathbb{R}^d$, and a non-optimal feasible point z_0 of $(\mathrm{P})_{c,b}$. We have to prove that there exist a vector $v \in G$ and a scalar $\alpha > 0$ such that $z_0 - \alpha v$ is a better feasible solution than z_0.

By z_1 denote another feasible solution with smaller cost function value than z_0. Thus, by the assumptions on G, we can write the vector $z_0 - z_1$ as a finite linear combination $z_0 - z_1 = \sum \alpha_i g_i$, where $g_i \in G$, $\alpha_i > 0$, $\alpha_i g_i \in \mathbb{X}$, and $\alpha_i g_i \sqsubseteq z_0 - z_1$ for all i. Since $c^\intercal(z_0 - z_1) > 0$ we have $c^\intercal(\alpha_j g_j) > 0$ for at least one g_j. We claim that $z_0 - \alpha_j g_j$ is feasible and has a better cost function value than z_0.

Clearly, $A(z_0 - \alpha_j g_j) = Az_0 - \alpha_j A g_j = b - 0 = b$ and $c^\intercal z_0 > c^\intercal(z_0 - \alpha_j g_j)$, since $g_j \in \ker_{\mathbb{X}}(A)$ and $c^\intercal(\alpha_j g_j) > 0$. So it remains to prove that $z_0 - \alpha_j g_j \geq 0$. We prove this for each component k separately. By the definition of the relation $\sqsubseteq$ we conclude from $\alpha_j g_j \sqsubseteq z_0 - z_1$ that

$$\alpha_j g_j^{(k)} (z_0 - z_1)^{(k)} \geq 0 \tag{1.1}$$

and

$$\left|\alpha_j g_j^{(k)}\right| \leq \left|(z_0 - z_1)^{(k)}\right| \tag{1.2}$$

If $\alpha_j g_j^{(k)} \leq 0$ then $(z_0 - \alpha_j g_j)^{(k)} = z_0^{(k)} - \alpha_j g_j^{(k)} \geq z_0^{(k)} \geq 0$. If, on the contrary, $\alpha_j g_j^{(k)} > 0$ then, by (1.1) and (1.2), also $(z_0 - z_1)^{(k)} \geq \alpha_j g_j^{(k)} > 0$ which implies $z_0^{(k)} - \alpha_j g_j^{(k)} \geq z_1^{(k)} \geq 0$ as desired. □

1.2 Criteria to check Positive Sum Property

1.2.1 Integer case

The following result is not only valid for the (saturated) lattice $\ker_{\mathbb{Z}^d}(A)$ but holds for arbitrary integer lattices Λ, as well. This more general version allows us to generalize Graver test sets and their computation to optimization problems $(\mathrm{MOD})_{c,b,\bar{b}}$ (introduced in the introductory chapter, page 16), see Section 2.1.

Lemma 1.2.1 *(Criterion for Positive Sum Property with respect to integer lattice)*

Let Λ be an integer sublattice of $\mathbb{Z}^d$. A symmetric set $G \subseteq \Lambda$ has the positive sum property with respect to Λ if and only if the following two conditions hold:

- *G finitely generates Λ over $\mathbb{Z}$, and*
- *for every pair $v, w \in G$, the vector $v + w$ can be written as a finite linear combination $v + w = \sum \alpha_i g_i$, where for all i we have $g_i \in G$, $\alpha_i \in \mathbb{Z}_{>0}$, and $g_i \sqsubseteq v + w$.*

Proof. We have to show that any non-zero $z \in \Lambda$ can be written as a finite positive linear integer combination of elements from G where each vector in this combination belongs to the same orthant as z. Since G generates Λ over $\mathbb{Z}$ and since G is symmetric, the vector z can be written as a linear combination $z = \sum \alpha_i v_i$ for finitely many $\alpha_i \in \mathbb{Z}_{>0}$ and $v_i \in G$. Note that $\sum \alpha_i \|v_i\|_1 \geq \|z\|_1$ with equality if and only if $v_i \sqsubseteq z$ for all i.

From the set of all such linear integer combinations $\sum \alpha_i v_i$ choose one such that $\sum \alpha_i \|v_i\|_1$ is minimal and assume that $\sum \alpha_i \|v_i\|_1 > \|z\|_1$. Otherwise $v_i \sqsubseteq z$ for all i and we are done. Therefore, there have to exist vectors v_{i_1}, v_{i_2} in this representation which have some component $k = k_0$ of different signs.

By the assumptions on G, the vector $v_{i_1} + v_{i_2}$ can be written as $v_{i_1} + v_{i_2} = \sum \beta_j v'_j$ for finitely many $\beta_j \in \mathbb{Z}_{>0}$, $v'_j \in G$, and $\beta_j v'_j \sqsubseteq v_{i_1} + v_{i_2}$ for all j. The latter implies that we have for each component $k = 1, \ldots, d$,

$$\sum_j \beta_j |v'^{(k)}_j| = |\sum_j \beta_j v'^{(k)}_j| = |(v_{i_1} + v_{i_2})^{(k)}| \leq |v^{(k)}_{i_1}| + |v^{(k)}_{i_2}|,$$

where the last inequality is strict for $k = k_0$ by construction. Summing up over $k = 1, \ldots, d$, yields $\sum \beta_j \|v'_j\|_1 = \|v_{i_1} + v_{i_2}\|_1 < \|v_{i_1}\|_1 + \|v_{i_2}\|_1$. But now z can be represented as

$$\begin{aligned} z &= \alpha_{i_1} v_{i_1} + \alpha_{i_2} v_{i_2} + \sum_{i \neq i_1, i_2} \alpha_i v_i \\ &= \sum \beta_j v'_j + (\alpha_{i_1} - 1) v_{i_1} + (\alpha_{i_2} - 1) v_{i_2} + \sum_{i \neq i_1, i_2} \alpha_i v_i \end{aligned}$$

and it holds

$$\sum \beta_j \|v'_j\|_1 + (\alpha_{i_1} - 1)\|v_{i_1}\|_1 + (\alpha_{i_2} - 1)\|v_{i_2}\|_1 + \sum_{i \neq i_1, i_2} \alpha_i \|v_i\|_1 < \sum \alpha_i \|v_i\|_1$$

in contradiction to the minimality required on $\sum \alpha_i \|v_i\|_1$. Altogether we obtain that $\|z\|_1 = \sum \alpha_i \|v_i\|_1$, that is, G has the positive sum property with respect to Λ. $\square$

1.2.2 Continuous case

In order to prove the criterion of Lemma 1.2.7 below we need to introduce the set of circuits associated with a matrix A. This set will turn out to be a minimal set having the positive sum property with respect to $\ker_{\mathbb{R}^d}(A)$.

Definition 1.2.2 *For $v \in \mathbb{R}^d$ the set $\operatorname{supp}(v) := \{j : v^{(j)} \neq 0\}$ is called the support of v.*

Definition 1.2.3 *(Circuits of a Matrix)*

A circuit of a matrix $A \in \mathbb{Z}^{l\times d}$ is a vector $q \in \ker_{\mathbb{R}^d}(A) \cap \mathbb{Z}^d$ of inclusion minimal support in $\ker_{\mathbb{R}^d}(A) \setminus \{0\}$ whose components have greatest common divisor 1.

Note that if A has only integer (or rational) entries then there is always a rational (and thus also an integer) vector for every minimal support in $\ker_{\mathbb{R}^d}(A)$. Proofs for the following three statements can be found in [19].

Lemma 1.2.4 *If v is a circuit of A and $w \in \ker_{\mathbb{R}^d}(A)$ satisfies $\operatorname{supp}(w) \subseteq \operatorname{supp}(v)$ then $w = \alpha v$ for some $\alpha \in \mathbb{R}$. In particular, we have $w = \pm v$ if v and w are circuits with the same support.*

Corollary 1.2.5 *Every matrix $A \in \mathbb{Z}^{l\times d}$ has only finitely many circuits.*

Lemma 1.2.6 *The set of all circuits of A has the positive sum property with respect to $\ker_{\mathbb{R}^d}(A)$.*

Now we are in the position to state and to prove the main result of this subsection.

Lemma 1.2.7 *(Criterion for Positive Sum Property with respect to $\ker_{\mathbb{R}^d}(A)$)*

A symmetric set $G \subseteq \ker_{\mathbb{R}^d}(A)$ has the positive sum property with respect to $\ker_{\mathbb{R}^d}(A)$ if and only if the following two conditions hold:

- *G finitely generates $\ker_{\mathbb{R}^d}(A)$ over $\mathbb{R}$, and*
- *for every pair $v, w \in G$, and all $\alpha \in \mathbb{R}$, the vector $v + \alpha w$ can be written as a finite linear combination $v + \alpha w = \sum \alpha_i g_i$, where for all $i = 1, \ldots, d$, we have $g_i \in G$ and $\operatorname{supp}(g_i) \subseteq \operatorname{supp}(v + \alpha w)$. Herein, only those values for $\alpha \in \mathbb{R}$ need to be considered, for which $v + \alpha w$ contains a zero entry at some component k at which neither v nor w have a zero entry, that is, $(v + \alpha w)^{(k)} = 0$ but $v^{(k)} w^{(k)} \neq 0$.*

Note that the condition $\operatorname{supp}(g_i) \subseteq \operatorname{supp}(v + \alpha w)$ *is in particular satisfied if* $\alpha_i > 0$ *and* $\alpha_i g_i \sqsubseteq v + \alpha w$.

Proof. It suffices to prove that G contains some non-zero scalar multiple of every circuit q of A. Lemma 1.2.7 then follows by application of Lemma 1.2.6.

Let q be an arbitrary circuit of A. Without loss of generality we may assume that the zero components of q are the first m components. Otherwise we may rearrange the variables to ensure this property. Since the vectors in G generate $\ker_{\mathbb{R}^d}(A)$ over $\mathbb{R}$ we have $q = \sum \alpha_i g_i$ for finitely many non-zero $\alpha_i \in \mathbb{R}$ and $g_i \in G$.

We will now rewrite this linear combination into a linear combination where all appearing vectors from G have a zero first component. Repeating this process with the second, third, $\ldots, m^{\text{th}}$ component, we arrive at a linear representation of q by elements from G which have zeros in their first m components. Choose any $g_j \in G$ that occurs in this last linear combination. Since $\operatorname{supp}(g_j) \subseteq \operatorname{supp}(q)$ and since q is a circuit, we conclude by Lemma 1.2.4 that $g_j \in G$ is a non-zero scalar multiple of q and our claim follows. Thus, it remains to show that this rewriting step is always possible.

If all g_i in $q = \sum \alpha_i g_i$ have already a zero first component we can go on with the second, third, $\ldots, m^{\text{th}}$ component. If not, $0 = q^{(1)} = \sum \alpha_i g_i^{(1)}$ implies that there exist at least two vectors g_j and g_k with non-zero first components. Take g_j and g_k and rewrite $\alpha_j g_j + \alpha_k g_k$ as follows.

$$\begin{aligned} \alpha_j g_j + \alpha_k g_k &= \alpha_j g_j + \alpha_k \left(g_k - \frac{g_k^{(1)}}{g_j^{(1)}} g_j \right) + \alpha_k \frac{g_k^{(1)}}{g_j^{(1)}} g_j \\ &= \left(\alpha_j + \alpha_k \frac{g_k^{(1)}}{g_j^{(1)}} \right) g_j + \alpha_k \left(g_k - \frac{g_k^{(1)}}{g_j^{(1)}} g_j \right). \end{aligned}$$

Since the first component of $(g_k - g_k^{(1)} g_j / g_j^{(1)})$ vanishes, the assumptions on G yield a representation of $(g_k - g_k^{(1)} g_j / g_j^{(1)})$ as a linear combination of elements in G whose support is contained in the support of $(g_k - g_k^{(1)} g_j / g_j^{(1)})$. Thus, each vector in this linear combination has a zero first component. Substituting this representation back into the above linear combination for q we arrive at a new representation $q = \sum \alpha_i' g_i'$, where the number of vectors in this linear combination with non-zero first component is at least one less than the corresponding number for $q = \sum \alpha_i g_i$. Hence this process terminates with a finite linear representation of q using only elements from G with a zero first component. Now we can go on with the second, third, $\ldots, m^{\text{th}}$ component.

Clearly, zero entries on previous components (that are already considered) will not be destroyed by our rewriting steps. This finally concludes the proof. □

1.3 Completion Algorithm

Lemmas 1.2.1 and 1.2.7 reduce the decision on whether a finite symmetric set F has the positive sum property with respect to an integer lattice Λ or with respect to $\ker_{\mathbb{R}^d}$ to a finite number of representability tests which may be treated algorithmically. In order to complete a set of vectors with respect to the positive sum property, these lemmas suggest the following procedure which adds further elements to the set as long as it does not have the desired positive sum property. Such procedures are known as completion procedures [7].

Algorithm 1.3.1 *(Completion Procedure)*

Input: a finite symmetric set $F \subseteq \ker_{\mathbb{R}^d}(A)$ *generating* Λ *over* $\mathbb{Z}$ *or* $\ker_{\mathbb{R}^d}(A)$ *over* $\mathbb{R}$

Output: a set $G \supseteq F$ *that has the positive sum property with respect to* Λ *or* $\ker_{\mathbb{R}^d}(A)$

$G := F$

$C := \bigcup_{f,g \in G} \text{S-vectors}(f,g)$

while $C \neq \emptyset$ *do*

  $s :=$ *an element in* C

  $C := C \setminus \{s\}$

  $f := \text{normalForm}(s, G)$

  if $f \neq 0$ *then*

    $C := C \cup \bigcup_{g \in G} \text{S-vectors}(f,g)$

    $G := G \cup \{f\}$

return G.

In this algorithm, the set S-vectors(f,g) corresponds to the "critical" vectors that are described in Lemmas 1.2.1 and 1.2.7, respectively. That is, S-vectors$(f,g) = \{f+g\}$ for the integer case and S-vectors$(f,g) = \bigcup_\alpha \{f + \alpha g\}$ for the continuous case. As specified in Lemma 1.2.7, only those values for $\alpha \in \mathbb{R}$ need to be considered, for which $f + \alpha g$ contains a zero entry at some component k at which neither f nor g have a

zero entry, that is, $(f+\alpha g)^{(k)} = 0$ but $f^{(k)}g^{(k)} \neq 0$. Note that if $s \in \text{S-vectors}(v,w)$ then $\text{supp}(s) \subseteq (\text{supp}(v) \cup \text{supp}(w)) \setminus \{e\}$ for some $e \in \text{supp}(v) \cap \text{supp}(w)$. This construction is also used in matroid theory to characterize circuits (see for example Oxley [36]).

Behind the function normalForm(s,G) there is the following algorithm which returns 0 if a representation $s = \sum \alpha_i g_i$ with finitely many $\alpha_i \in \mathbb{Z}_{>0}$ (or $\alpha_i \in \mathbb{R}_{>0}$), $g_i \in G$, and $\alpha_i g_i \sqsubseteq s$ is found, or it returns a vector t such that a representation of this kind is possible for $G \cup \{t\}$.

The normalForm algorithm aims at finding a representation $s = \sum \alpha_i g_i$ with finitely many $\alpha_i \in \mathbb{Z}_{>0}$ (or $\alpha_i \in \mathbb{R}_{>0}$), $g_i \in G$, and $\alpha_i g_i \sqsubseteq s$ by reducing s by elements of G in such a way that, if at some point of this reduction the zero vector is reached, a desired representation $s = \sum \alpha_i g_i$ has been found. If the reduction process terminates with a vector $t \neq 0$ then a desired representation $s = \sum \alpha_i g_i$ with vectors from $G \cup \{t\}$ has been constructed. The vector t is called a normal form of s with respect to the set G.

Algorithm 1.3.2 *(Normal form algorithm)*

Input: a vector s, a set G of vectors

Output: a normal form of s with respect to G

while there is some $g \in G$ such that s is reducible by g do

    $s :=$ *reduce s by g*

return s

The reduction involved in the normalForm algorithm has to be defined separately for the integer and the continuous cases. For the special situation $\Lambda = \ker_{\mathbb{Z}^d}(A)$, our definition for the integer case leads to an algorithm already presented by Pottier [37] in the disguised form of a d-dimensional Euclidean algorithm.

1.3.1 Integer Case

Here we say that $s \in \mathbb{Z}^d$ can be reduced by $g \in \mathbb{Z}^d$ to $s-g$ if $g \sqsubseteq s$. Thus, in case of reducibility, we have $s = g + (s-g)$ with $g \sqsubseteq s$ and $s-g \sqsubseteq s$. Since $\|s-g\|_1 < \|s\|_1$, we conclude that normalForm(s,G) always terminates.

To prove termination of the completion algorithm in the integer case we need the Gordan-Dickson Lemma (see for example Section 4.2 in [16]). The following is an equivalent geometric formulation.

Lemma 1.3.3 *(Gordan-Dickson Lemma)*

Let $\{p_1, p_2, \ldots\}$ *be a sequence of points in* $\mathbb{Z}_+^d$ *such that* $p_i \not\leq p_j$ *whenever* $i < j$*. Then this sequence is finite.*

Lemma 1.3.4 *With the above definitions of* S-vectors *and of* normalForm *for the integer case the Completion Procedure 1.3.1 terminates and satisfies its specifications.*

Proof. Termination of the above algorithm follows immediately by application of the Gordan-Dickson Lemma to the sequence $\{(v^+, v^-) : v \in G \setminus F\}$, where for $v \in \mathbb{Z}^d$ we component-wise define $v^+ := \max(0, v)$ and $v^- := (-v)^+ = \max(0, -v)$. To see this, note that $f = \text{normalForm}(s, G)$ implies that there is no $g \in G$ with $g \sqsubseteq f$, or in other words, there is no $g \in G$ with $(g^+, g^-) \leq (f^+, f^-)$. Thus, the algorithm produces a sequence of vectors $\{(v^+, v^-) : v \in G \setminus F\} = \{f_1, f_2, \ldots\}$, where $(f_i^+, f_i^-) \not\leq (f_j^+, f_j^-)$ for $i < j$. Such a sequence is always finite by the Gordan-Dickson Lemma. Correctness of the algorithm follows immediately from Lemma 1.2.1, since upon termination $\text{normalForm}(v + w, G) = 0$ for all $v, w \in G$, giving a representation $v + w = \sum \alpha_i g_i$ with $\alpha_i \in \mathbb{Z}_{>0}$, $g_i \in G$, and $g_i \sqsubseteq v + w$. □

1.3.2 Continuous Case

Here we say that $s \in \mathbb{R}^d$ can be reduced by $g \in \mathbb{R}^d$ if $\text{supp}(g) \subseteq \text{supp}(s)$. In case of reducibility s is reduced to $s - \alpha g$ where $\alpha \in \mathbb{R}$ is chosen in such a way that $\text{supp}(s - \alpha g) \subsetneq \text{supp}(s)$, which implies that $\text{normalForm}(s, G)$ always terminates. Moreover, $s = \alpha g + (s - \alpha g)$ where $\text{supp}(\alpha g) \subseteq \text{supp}(s)$ and $\text{supp}(s - \alpha g) \subseteq \text{supp}(s)$.

Remark 1.3.5 *The main ingredient in the proof of Lemma 1.2.7 was that* $f + \alpha g$ *can be written as* $f + \alpha g = \sum \alpha_i g_i$ *with* $g_i \in G$ *and* $\text{supp}(g_i) \subseteq \text{supp}(f + \alpha g)$ *for all* i*. Note that this is less strict than the reduction defined above, since we do not assume that* $\text{supp}(f + \alpha g - \alpha_i g_i) \subsetneq \text{supp}(f + \alpha g)$ *for some* i*. Reducing* $f + \alpha g$ *by some* g_i *to* $f + \alpha g - \alpha_i g_i$*, however, would require this latter condition.*

Thus, in later proofs, we will only construct representations $f + \alpha g = \sum \alpha_i g_i$ *with* $g_i \in G$ *and* $\text{supp}(g_i) \subseteq \text{supp}(f + \alpha g)$ *for all* i*.*

Lemma 1.3.6 *With the above definitions of* S-vectors *and of* normalForm *for the continuous case the Completion Procedure 1.3.1 terminates and satisfies its specifications.*

Proof. Each vector in $G \setminus F$ must have a different support. Hence the algorithm terminates. Correctness of the above algorithm follows from Lemma 1.2.7 since upon termination normalForm$(v + \alpha w, G) = 0$ for all $v, w \in G$, and for all α such that $v + \alpha w$ contains a zero entry at some component at which both v and w are non-zero. This normalForm reduction to zero provides a representation $v + \alpha w = \sum \alpha_i g_i$ with $\alpha_i \in \mathbb{R}$, $g_i \in G$, and $\text{supp}(\alpha_i g_i) \subseteq \text{supp}(v + \alpha w)$ as required. □

1.4 Conclusions

In this chapter we have presented the positive sum property, a property inherent to LP, IP, and MIP Graver bases, which already ensures their universal test set property. In Lemmas 1.2.1 and 1.2.7 we identified two criteria to check the positive sum property of a set with respect to an integer lattice or with respect to the (continuous) kernel of a given matrix. These criteria led us to the algorithmic pattern of a completion procedure in order to complete a given generating set of the lattice or of the kernel with respect to the positive sum property. As we will see in the subsequent chapters, this algorithmic pattern is closely related to Graver basis computations and reappears using more complicated structures than vectors. It will suffice to redefine the input set, the function normalForm, and the set of S-vectors that are employed by the completion procedure.

In the following we will introduce Graver test sets for LP, IP, and MIP.

Chapter 2

Graver Test Sets

In the sequel we introduce, separately for each situation, Graver test sets in LP, IP, and MIP as minimal sets of vectors having the positive sum property with respect to $\ker_{\mathbb{X}}(A)$, where $\mathbb{X} = \mathbb{R}^d$, $\mathbb{X} = \mathbb{Z}^d$, and $\mathbb{X} = \mathbb{Z}^{d_z} \times \mathbb{R}^{d_q}$, $d = d_z + d_q$, respectively.

LP and IP Graver test sets are always finite as was already shown by Graver in 1975 [19]. We show how the two completion procedures presented in Section 1.3 for the integer and the continuous cases can be used to compute both sets.

Moreover, we will present truncated IP Graver test sets and present a completion algorithm to compute them.

MIP Graver test sets need not be finite in general. To demonstrate this, we present a small example similar to the one by Cook et al. [14]. Thereafter, we give a solution to this finiteness problem by presenting a finite set of integer vectors which is inherent to the MIP Graver test set and from which an improving vector to a non-optimal solution can be reconstructed by solving a finite number of pure LP's. Moreover, we will again employ a completion procedure to compute these finitely many integer vectors.

Having introduced Graver test sets, we will deal with termination of the Augmentation Algorithm 0.0.2. It turns out that some caution is appropriate in the LP case to avoid zig-zagging even to non-optimal points. Finally, we will concentrate on finding initial feasible solutions to our problem. Such a solution is needed as input to the Augmentation Algorithm 0.0.2. Although there are algorithmic alternatives, we show that universal test sets can be used to solve this feasibility problem as well. Again, some care has to be taken in the LP case to ensure termination of this procedure.

2.1 IP Graver Test Sets

The notion of a Graver test set in IP is strongly related to the notion of a Hilbert basis, which we will recall now. For further details we refer to Schrijver [42] and Weismantel [57].

Definition 2.1.1 *(Hilbert Basis)*

Let C be a rational cone. A finite set $H = \{h_1, \ldots, h_t\} \subseteq C \cap \mathbb{Z}^d$ is called a Hilbert basis of C if every $z \in C \cap \mathbb{Z}^d$ has a representation of the form

$$z = \sum_{i=1}^{t} \lambda_i h_i$$

with non-negative integral multipliers $\lambda_1, \ldots, \lambda_t$.

Note that every pointed rational cone possesses a unique Hilbert basis that is minimal with respect to inclusion.

Definition 2.1.2 *Let $\mathbb{O}_j$ be the j^{th} orthant of $\mathbb{R}^d$ and H_j the unique minimal Hilbert basis of $\ker_{\mathbb{R}^d}(A) \cap \mathbb{O}_j$. Then we define $\mathcal{G}_{IP}(A) := \bigcup H_j \setminus \{0\}$ to be the IP Graver test set (or IP Graver basis) of A.*

As $\mathcal{G}_{\mathrm{IP}}(A)$ is a finite union of finite sets, $\mathcal{G}_{\mathrm{IP}}(A)$ has finite cardinality. By construction, the elements of $\mathcal{G}_{\mathrm{IP}}(A)$ are exactly all elements of $\ker_{\mathbb{Z}^d}(A) \setminus \{0\}$ that are minimal with respect to the partial ordering $\sqsubseteq$ on $\mathbb{Z}^d$. Moreover, if $v \in \mathcal{G}_{\mathrm{IP}}(A)$ then also $-v \in \mathcal{G}_{\mathrm{IP}}(A)$, that is, G is symmetric.

Lemma 2.1.3 *$\mathcal{G}_{IP}(A)$ has the positive sum property with respect to $\ker_{\mathbb{Z}^d}(A)$.*

Proof. Take any non-zero element $z \in \ker_{\mathbb{Z}^d}(A)$. Then z belongs to some orthant $\mathbb{O}_j$ and thus can be written as a positive integer linear combination of elements of the Hilbert basis $H_j \subseteq \mathcal{G}_{\mathrm{IP}}(A)$ of $\ker_{\mathbb{Z}^d}(A) \cap \mathbb{O}_j$. □

Corollary 2.1.4 *$\mathcal{G}_{IP}(A)$ is a universal test set for $(IP)_{c,b}$.*

In Section 1.3 we have already seen how a generating set of $\ker_{Z^d}(A)$ over $\mathbb{Z}$ can be completed with respect to the positive sum property. The resulting set G has to contain the IP Graver test set which consists of all $\sqsubseteq$-minimal elements in G.

In fact, this algorithm computes the set $\mathcal{G}(\Lambda)$ of all $\sqsubseteq$-minimal elements in $\Lambda \setminus \{0\}$ for any integer lattice $\Lambda := \{z \in \mathbb{Z}^d : Az = 0, \bar{A}z \equiv 0 \pmod{p}\}$. Now we show that $\mathcal{G}(\Lambda)$ has the positive sum property with respect to Λ and thus, constitutes a universal test set for the family of optimization problems

$$(\mathrm{MOD})_{c,b,\bar{b}}: \qquad \min\{c^\intercal z : Az = b, \bar{A}z \equiv \bar{b} \pmod{p}, z \in \mathbb{Z}^d_+\},$$

as $c \in \mathbb{R}^d$, $b \in \mathbb{R}^l$, and $\bar{b} \in \mathbb{R}^{\bar{l}}$ vary.

Lemma 2.1.5 *The set $\mathcal{G}(\Lambda)$ has the positive sum property with respect to Λ.*

Proof. We have to prove that every non-zero $z \in \Lambda$ can be written as a finite linear combination $z = \sum \alpha_i g_i$ with $\alpha_i \in \mathbb{Z}_{>0}$, $g_i \in \mathcal{G}(\Lambda)$, and $g_i \sqsubseteq z$ for all i.

Let $z \in \Lambda \setminus \{0\}$ be given. By definition of $\mathcal{G}(\Lambda)$, there is some $g_1 \in \mathcal{G}(\Lambda)$ with $g_1 \sqsubseteq z$. Thus, $z - g_1 \sqsubseteq z$ and $\|z - g_1\|_1 < \|z\|_1$. Going on with the same argument for $z - g_1$ instead, the latter relation implies that we must eventually arrive at 0. But then $z = \sum_{i=1}^k g_i$, $g_i \in \mathcal{G}(\Lambda)$, and $g_i \sqsubseteq z$ for all i, which completes the proof. □

Moreover, employing the same arguments as in the proof of Lemma 1.1.2, we conclude the following.

Corollary 2.1.6 *The set $\mathcal{G}(\Lambda)$ constitutes a universal test set for the family of optimization problems $(MOD)_{c,b,\bar{b}}$ as $c \in \mathbb{R}^d$, $b \in \mathbb{R}^l$, and $\bar{b} \in \mathbb{R}^{\bar{l}}$ vary.*

Universal test sets (universal Gröbner bases in particular) for $(\mathrm{MOD})_{c,b,\bar{b}}$ were already considered in [50]. Here, we only want to emphasize that all algorithmic results in this thesis for $(\mathrm{IP})_{c,b}$ readily extend to $(\mathrm{MOD})_{c,b,\bar{b}}$, since only the lattice structure of $\ker_{\mathbb{Z}^d}(A)$ and the positive sum property with respect to this lattice are exploited in the proofs. However, for ease of exposition only, we will restrict our attention to the special case $\Lambda = \ker_{\mathbb{Z}^d}(A)$.

2.2 Truncated Graver Test Sets for $(\mathrm{IP})_{c,b}$

The notion of a truncated Graver test set already appeared in a paper by Thomas and Weismantel [53]. Our algorithm to compute a truncated Graver test set could be deduced from their approach as well by defining their partial order not on all components of the right-hand side (b, u) but only on the components of u, the upper bounds.

Consider the family of optimization problems $\min\{c^\intercal z : Az = b, z \leq u, z \in \mathbb{Z}^d_+\}$ where only the vectors $b \in \mathbb{R}^l$ and $c \in \mathbb{R}^d$ are allowed to vary. The upper bounds $u \in (\mathbb{Z} \cup \{\infty\})^d$ are assumed to remain fixed. Let $u^{(i)} = \infty$ if $z^{(i)}$ is not bounded from above. Prominent examples arise in combinatorial optimization as 0-1 problems.

Example. Before we introduce truncated Graver test sets, let us demonstrate how the algorithm presented in Section 1.3 deals with the upper bounds when it computes a Graver test set for this problem. Introduction of slack variables leads to the computation of $\mathcal{G}_{\mathrm{IP}}(B)$, where

$$B = \begin{pmatrix} A & 0 \\ E & E \end{pmatrix}.$$

Herein, E denotes the identity matrix of size d. Each element of $\ker_{\mathbb{Z}^{2d}}(B)$ has the form $(v, -v)$. Moreover, $(w, -w) \sqsubseteq (v, -v)$ if and only if $w \sqsubseteq v$. This shows that the computations of $\mathcal{G}_{\mathrm{IP}}(A)$ and of $\mathcal{G}_{\mathrm{IP}}(B)$ are isomorphic using the correspondence $v \leftrightarrow (v, -v)$. That is, to speed up the calculation of $\mathcal{G}_{\mathrm{IP}}(B)$, we may first compute $\mathcal{G}_{\mathrm{IP}}(A)$ and finally map each element $v \in \mathcal{G}_{\mathrm{IP}}(A)$ to $(v, -v)$.

Thus, essentially, the bounds $z \leq u$ are ignored during the computation, which is due to the fact that u is assumed to vary as well. Therefore, computations are done in $\mathbb{Z}^{2d}$ and do not exploit the given fixed upper bounds. □

However, the fixed upper bounds on the variables allow a truncation of the Graver test set $\mathcal{G}_{\mathrm{IP}}(A)$ for $\min\{c^\intercal z : Az = b, z \in \mathbb{Z}^d_+\}$, since only those vectors v with $|v^{(i)}| \leq u^{(i)}$ for all $i = 1, \ldots, d$, may transform a feasible solution $0 \leq z_1 \leq u$ into a feasible solution $z_2 = z_1 - v$ satisfying $0 \leq z_2 \leq u$, since $v = z_1 - z_2$.

The following lemma is an immediate consequence from the positive sum property of $\mathcal{G}_{\mathrm{IP}}(A)$ with respect to $\ker_{\mathbb{Z}^d}(A)$.

Lemma 2.2.1 *The truncated Graver test set*

$$\mathcal{G}_u(A) := \mathcal{G}_{IP}(A) \cap \{z \in \mathbb{Z}^d : -u \leq z \leq u\}$$

is a universal test set for $\min\{c^\intercal z : Az = b, z \leq u, z \in \mathbb{Z}^d_+\}$*, where only* $b \in \mathbb{R}^l$ *and* $c \in \mathbb{R}^d$ *may vary.*

Proof. Take an arbitrary $v \in \ker_{\mathbb{Z}^d}(A) \cap \{z \in \mathbb{Z}^d : -u \leq z \leq u\}$. By the positive sum property of $\mathcal{G}_{\mathrm{IP}}(A)$ with respect to $\ker_{\mathbb{Z}^d}(A)$, v can be written as a positive integer linear combination $v = \sum \alpha_i g_i$ of elements $g_i \in \mathcal{G}_{\mathrm{IP}}(A)$, $\alpha_i \in \mathbb{Z}_{>0}$, and $g_i \sqsubseteq v$. The latter condition, however, together with $-u \leq v \leq u$ implies that also $-u \leq g_i \leq u$

and thus, $g_i \in \mathcal{G}_u(A)$ for all i. Thus, $\mathcal{G}_u(A)$ has the positive sum property with respect to $\ker_{\mathbb{Z}^d}(A) \cap \{z \in \mathbb{Z}^d : -u \leq z \leq u\}$. As in the proof of Lemma 1.1.2, this immediately implies that $\mathcal{G}_u(A)$ is a universal test set for $\min\{c^\intercal z : Az = b, z \leq u, z \in \mathbb{Z}^d_+\}$, where only $b \in \mathbb{R}^l$ and $c \in \mathbb{R}^d$ may vary. □

Example. To see that the truncated Graver test set $\mathcal{G}_u(A)$ may be much smaller than $\mathcal{G}_{\text{IP}}(A)$ consider the problem $\min\{c^\intercal z : (k\ 1\ 1)z = b,\ z \leq 1,\ z \in \mathbb{Z}^3_+\}$ for given given integer $k > 2$. Then we have

$$\mathcal{G}_{\text{IP}}(A) = \{\pm(0,1,-1), \pm(1,0,-k), \pm(1,-1,-(k-1)), \ldots, \pm(1,-k,0)\}$$

but $\mathcal{G}_{(1,1,1)}(A) = \{\pm(0,1,-1)\}$. Note that this truncated test set does not change if we do not impose a finite upper bound on z_1. □

Therefore, truncation may drastically reduce the number of interesting vectors in a Graver test set. Of major importance, however, is the fact that $\mathcal{G}_u(A)$ can be computed directly, without computing the much bigger set $\mathcal{G}_{\text{IP}}(A)$ first. In the following, we present an algorithm which computes all $\sqsubseteq$-minimal solutions in the set $\{z \in \mathbb{Z}^d : Az = 0, l \leq z \leq u\} \setminus \{0\}$, where $l \leq 0 \leq u$. Thus, for special choices of l and u this includes algorithmic solutions to the following problems:

- For $l = 0$ and $u = 1$ (component-wise) the algorithm computes (partial) test sets for combinatorial optimization problems.
- For $l = -u$ the algorithm computes the desired truncated Graver test set $\mathcal{G}_u(A)$.
- For $l = -\infty$ and $u = +\infty$ the algorithm is similar to Algorithm 2.1.7 in Urbaniak [54] and computes $\mathcal{G}_{\text{IP}}(A)$. However, it is much slower than Algorithm 1.3.1 (Section 1.3), since the computation is done in a higher dimensional space.
- For $b = 0, l = 0$, and $u = +\infty$ the algorithm computes the unique minimal Hilbert basis for the pointed rational cone $\ker_{\mathbb{R}^d}(A) \cap \mathbb{R}^d_+$.

Analogously to integer lattices, Lemma 2.1.5, one can prove that the set of all $\sqsubseteq$-minimal solutions in the set $\{z \in \mathbb{Z}^d : Az = 0, l \leq z \leq u\} \setminus \{0\}$ has the positive sum property with respect to $\{z \in \mathbb{Z}^d : Az = 0, l \leq z \leq u\}$. Thus, it is not surprising that the computation of these $\sqsubseteq$-minimal solutions follows the pattern of a completion procedure as given in Algorithm 1.3.1. Now we have to specify the input set, the reduction needed in the normalForm algorithm, and the set of S-vectors.

As input set we choose $F = \bigcup_{i:l^{(i)}<0} \{(-e_i, -Ae_i)\} \cup \bigcup_{i:u^{(i)}>0} \{(e_i, Ae_i)\}$, where e_i denotes the i^{th} unit vector in $\mathbb{R}^d$. Moreover, $s \in \mathbb{Z}^{d+l}$ reduces by $g \in \mathbb{Z}^{d+l}$ to $s - g$ if $g \sqsubseteq s$.

Finally, we define

$$\text{S-vectors}((v,Av),(w,Aw)) := \begin{cases} \{(v+w, A(v+w))\} & \text{if } l \leq v+w \leq u \text{ and both,} \\ & v \text{ and } w, \text{ lie in the same} \\ & \text{orthant of } \mathbb{R}^d \\ \emptyset & \text{otherwise.} \end{cases}$$

Lemma 2.2.2 *With the above definitions of input set,* normalForm*, and* S-vectors*, the Completion Algorithm 1.3.1 terminates and returns a set G which contains $(z_0, 0)$ for all $\sqsubseteq$-minimal integer solutions z_0 of the system $Az = 0$, $l \leq z \leq u$, and $z \neq 0$.*

The set of all minimal integer solutions to $Az = 0$, $l \leq z \leq u$, and $z \neq 0$, is the set of all those vectors z such that $(z,0) \in G$ and such that $(z,0)$ is irreducible with respect to $G \setminus \{(z,0)\}$.

The following proof is similar to that of Lemma 1.2.1. However, we use the special input set F to show that much fewer S-vectors compared to those in Algorithm 1.3.1 need to be considered here.

Proof. Again, termination of the above algorithm follows immediately by applying the Gordan-Dickson Lemma to the sequence $\{(v^+, v^-) : v \in G \setminus F\}$ as it is generated by the algorithm.

For the correctness proof let G denote the set that is returned by the algorithm. We show that any non-zero vector $(z,0)$ with $z \in \ker_{\mathbb{Z}^d}(A)$ and $l \leq z \leq u$ can be written as a positive linear integer combination of elements of G, where each vector in this combination lies in the same orthant as $(z,0)$. From this the claim immediately follows since there is exactly one such representation $(z,0) = 1 \cdot (z,0)$ for all those $z \in \ker_{\mathbb{Z}^d}(A) \setminus \{0\}$ that are minimal with respect to $\sqsubseteq$.

Since G contains the (particularly nice) symmetric input set F, the vector $(z,0)$ can be written as a positive linear integer combination $(z,0) = \sum \alpha_i (v_i, Av_i)$ with $\alpha_i \in \mathbb{Z}_{>0}$ and vectors $(v_i, Av_i) \in G$, $v_i \sqsubseteq z$ for all i. From the set of all such positive linear integer combinations $\sum \alpha_i (v_i, Av_i)$ choose one such that $\sum \alpha_i \|Av_i\|_1$ is minimal. Note that $\sum \alpha_i \|Av_i\|_1 \geq 0$ with equality if and only if $\|Av_i\|_1 = 0$ for all i.

If $\sum \alpha_i \|Av_i\|_1 = 0$ were true, we could conclude $Av_i = 0$ for all i and we had found a desired linear integer representation of z. Thus, assume on the contrary that $\sum \alpha_i \|Av_i\|_1 > 0$ holds.

Therefore, there have to exist (v_{i_1}, Av_{i_1}), (v_{i_2}, Av_{i_2}) such that $(Av_{i_1})^{(k_0)}(Av_{i_2})^{(k_0)} < 0$ for some component k_0. By construction, v_{i_1} and v_{i_2} lie in the same orthant of $\mathbb{R}^d$ and from $v_{i_1} + v_{i_2} \sqsubseteq z$ we conclude $l \leq -z^- \leq v_{i_1} + v_{i_2} \leq z^+ \leq u$. Thus, the vector

$(v_{i_1}, Av_{i_1}) + (v_{i_2}, Av_{i_2})$ is an S-vector and was reduced to 0 by G in the course of the algorithm. This gives a representation $(v_{i_1}, Av_{i_1}) + (v_{i_2}, Av_{i_2}) = \sum \beta_j (v'_j, Av'_j)$ for some positive integers β_j and some vectors $(v'_j, Av'_j) \in G$. Moreover, it holds $\beta_j(v'_j, Av'_j) \sqsubseteq (v_{i_1}, Av_{i_1}) + (v_{i_2}, Av_{i_2})$ for all j, implying that $v'_j \sqsubseteq z$ and that for each component k

$$\begin{aligned} \sum \beta_j \left|(Av'_j)^{(k)}\right| &= \left|\sum \beta_j (Av'_j)^{(k)}\right| \\ &= \left|(A(v_{i_1} + v_{i_2}))^{(k)}\right| \\ &\leq \left|(Av_{i_1})^{(k)}\right| + \left|(Av_{i_2})^{(k)}\right|, \end{aligned}$$

holds, where the last inequality is strict for $k = k_0$ by construction. Summation over k now gives

$$\sum \beta_j \|Av'_j\|_1 = \|A(v_{i_1} + v_{i_2})\|_1 < \|Av_{i_1}\|_1 + \|Av_{i_2}\|_1.$$

This implies that

$$\sum \beta_i (v'_i, Av'_i) + (\alpha_{i_1} - 1)(v_{i_1}, Av_{i_1}) + (\alpha_{i_2} - 1)(v_{i_2}, Av_{i_2}) + \sum_{i \neq i_1, i_2} \alpha_i (v_i, Av_i)$$

is a valid linear integer representation of $(z, 0)$ that contradicts the minimality of $\sum \alpha_i \|Av_i\|_1$. We conclude that already $\sum \alpha_i \|Av_i\|_1 = 0$ and the claim follows. □

Lemma 2.2.3 *If $A = (A'|E)$ then no additional variables are needed.*

Proof. The input vectors of the above algorithm become $(e_i, A'e_i, e_i)$. Thus, each vector that appears in the algorithm will have the form $(v, A'v, v)$. Since we have $(w, A'w, w) \sqsubseteq (v, A'v, v)$ if and only if $(w, A'w) \sqsubseteq (v, A'v)$, we can ignore the additional variables. □

A similar proof works for matrices $A = (A'| - E)$ or for the case, where only some equations arise from inequalities with the help of slack-variables. Thus, we can avoid all or at least some of the additional variables, which leads to an algorithmic speed up.

2.3 LP Graver Test Sets

For every orthant $\mathbb{O}_j$ of $\mathbb{R}^d$ consider the pointed rational cone $C_j := \ker_{\mathbb{R}^d}(A) \cap \mathbb{O}_j$. Up to scalar factors this cone has a unique inclusion minimal generating set over $\mathbb{R}$. Since C_j is a (pointed) rational cone we may scale each generator to have only integer components with greatest common divisor 1.

Definition 2.3.1 *For every orthant $\mathbb{O}_j$ of $\mathbb{R}^d$ let H_j be the unique minimal generating set over $\mathbb{R}$ of the pointed rational cone $\ker_{\mathbb{R}^d}(A) \cap \mathbb{O}_j$, where the components of each generator are scaled to integers with greatest common divisor 1. Then we define $\mathcal{G}_{LP}(A) := \bigcup H_j$ to be the LP Graver test set (or LP Graver basis) of A.*

As $\mathcal{G}_{\mathrm{LP}}(A)$ is a finite union of finite sets, $\mathcal{G}_{\mathrm{LP}}(A)$ has finite cardinality.

Lemma 2.3.2 *$\mathcal{G}_{LP}(A)$ has the positive sum property with respect to $\ker_{\mathbb{R}^d}(A)$. Thus, $\mathcal{G}_{LP}(A)$ is a universal test set for $(LP)_{c,b}$.*

Proof. Take any non-zero element $z \in \ker_{\mathbb{R}^d}(A)$. Thus, z belongs to some orthant $\mathbb{O}_j$ and can be written as a positive linear combination of elements of the generating set $H_j \subseteq \mathcal{G}_{\mathrm{LP}}(A)$ of $\ker_{\mathbb{R}^d}(A) \cap \mathbb{O}_j$. The second part follows from Lemma 1.1.2. □

Lemma 2.3.3 *$\mathcal{G}_{LP}(A)$ coincides with the set of all circuits of A.*

Proof. We show that $\mathcal{G}_{\mathrm{LP}}(A)$ contains every circuit. The claim then follows from the positive sum property of the set of circuits, Lemma 1.2.6.

Let q be a circuit of A belonging to some orthant $\mathbb{O}_j$ of $\mathbb{R}^d$. Then q can be written as $q = \sum \alpha_i g_i$ where $\alpha_i > 0$ and $g_i \in H_j$. This implies $\alpha_i g_i \sqsubseteq q$ for all i. We conclude that $\mathrm{supp}(g_i) \subseteq \mathrm{supp}(q)$ and thus $q = g_i$ by Lemma 1.2.4. □

In Section 1.3 we have already seen how a generating set of $\ker_{\mathbb{R}^d}(A)$ over $\mathbb{R}$ can be completed with respect to the positive sum property. The resulting set G has to contain a scalar multiple of each circuit of A. Thus, the LP Graver test set consists of all support minimal elements in G, normalized to integer vectors whose components have a greatest common divisor 1. In that way we may compute the LP Graver test set.

2.4 MIP Graver Test Sets

2.4.1 Introduction

Let $A = (A_1|A_2)$, where the columns of A_1 and A_2 correspond to the integer and continuous variables, respectively. Throughout this section A_1 and A_2 are assumed to be integer matrices of sizes $l \times d_1$ and $l \times d_2$. Analogously, we subdivide $c = (c_z, c_q)$, where $c_z \in \mathbb{R}^{d_1}$ and $c_q \in \mathbb{R}^{d_2}$. Let $z \in \mathbb{Z}^{d_1}$ and $q \in \mathbb{R}^{d_2}$ denote the integer and

continuous variables, respectively, and let $d = d_1 + d_2$. Our aim is to construct a universal test set for the family of optimization problems $(\mathrm{P})_{c,b}$ as $c \in \mathbb{R}^d$ and $b \in \mathbb{R}^l$ vary.

Again, a Graver test set in the mixed-integer situation is defined to be an inclusion minimal subset of $\ker_{\mathbb{X}}(A)$ which has the positive sum property with respect to $\ker_{\mathbb{X}}(A)$. This leads us to the following set of vectors.

Definition 2.4.1 *The MIP Graver test set $\mathcal{G}_{MIP}(A)$ contains all vectors*

- $(0, q)$, $q \in \mathcal{G}_{LP}(A_2)$, *and*
- $(z, q) \in \ker_{\mathbb{X}}(A)$, $z \neq 0$, *and such that there is no* $(z', q') \in \ker_{\mathbb{X}}(A)$ *satisfying* $(z', q') \sqsubseteq (z, q)$.

The proof of the following lemma can be found in [17]. (To be self-contained we reproduce its proof in the appendix, Section 2.7.)

Lemma 2.4.2 *$\mathcal{G}_{MIP}(A)$ is an inclusion minimal set which has the positive sum property with respect to* $\ker_{\mathbb{X}}(A)$. *Consequently, $\mathcal{G}_{MIP}(A)$ is a universal test set for $(P)_{c,b}$.*

The set $\mathcal{G}_{\mathrm{MIP}}(A)$, however, need not be finite in general. To demonstrate this, we present an example similar to the one by Cook et al. [14].

Example. Consider the family of optimization problems

$$\min\{c_0 z + c_1 q_1 + c_2 q_2 : z + q_1 + q_2 = b, z \in \mathbb{Z}_+, q_1, q_2 \in \mathbb{R}_+\}$$

as $c = (c_0, c_1, c_2) \in \mathbb{R}^3$ and $b \in \mathbb{R}$ vary. Thus, we have $A = (A_1|A_2)$ where $A_1 = (1)$ and $A_2 = (1\ 1)$. Every element in $\ker_{\mathbb{Z}\times\mathbb{R}^2}(A)$ can be written as linear combination $\alpha_1(1, -1, 0) + \alpha_2(0, 1, -1)$ with $\alpha_1 \in \mathbb{Z}$ and $\alpha_2 \in \mathbb{R}$. We will prove that for every $\beta \in \mathbb{R}$, $0 < \beta < 1$, the vector

$$u := (1, -1, 0) + \beta(0, 1, -1) = (1, -1 + \beta, -\beta) \in \ker_{\mathbb{Z}\times\mathbb{R}^2}(A) \setminus \{0\}$$

cannot be written as a sum $v + w$ with $v, w \in \ker_{\mathbb{Z}\times\mathbb{R}^2}(A) \setminus \{0\}$ and $v, w \sqsubseteq u$.

Suppose on the contrary that such vectors v, w exist. Since $v, w \sqsubseteq u$, one of the two vectors, say v, must have a zero integer component. Hence $v = \gamma(0, 1, -1)$ for some $\gamma \in \mathbb{R}$. But, clearly, for no choice of $\gamma \in \mathbb{R}$ and of $\beta \in \mathbb{R}$, $0 < \beta < 1$, the vectors $(1, -1 + \beta, -\beta)$ and $\gamma(0, 1, -1)$ belong to the same orthant of $\mathbb{R}^3$, which is a contradiction to $v \sqsubseteq u$. We conclude that for all $\beta \in \mathbb{R}$, $0 < \beta < 1$, the vector $(1, -1 + \beta, -\beta)$ is $\sqsubseteq$-minimal and thus, it belongs to $\mathcal{G}_{\mathrm{MIP}}(A)$. □

Having only an infinite test set available, we cannot yet make algorithmic use of the Augmentation Algorithm 0.0.2 to improve a feasible initial solution to optimality. However, we will construct a finite set $\mathcal{G}_{\mathbb{Z}}(A) \subseteq \mathbb{Z}^{d_1}$ from which an improving vector to a non-optimal solution of the given problem can be reconstructed in finitely many steps. Thus, the Augmentation Algorithm 0.0.2 can be employed again to find an optimal solution.

2.4.2 Finitely many Integer Parts

Consider the projection $\phi : \mathbb{Z}^{d_1} \times \mathbb{R}^{d_2} \to \mathbb{Z}^{d_1}$ which maps each mixed-integer vector onto its d_1 integer components. Define $\mathcal{G}_{\mathbb{Z}}(A) = \mathcal{G}_{\mathbb{Z}}(A_1|A_2) := \phi(\mathcal{G}_{\text{MIP}}(A))$ to be the set of images of the elements in $\mathcal{G}_{\text{MIP}}(A)$.

Lemma 2.4.3 *$\mathcal{G}_{\mathbb{Z}}(A_1|A_2)$ is finite for every matrix $A \in \mathbb{Z}^{l\times d}$ and for any subdivision $A = (A_1|A_2)$.*

Proof. Each element $(z, q) \in \mathcal{G}_{\text{MIP}}(A)$ satisfies $\|z\|_1 \leq \Delta(A_1, A_2)$ for some scalar $\Delta(A_1, A_2)$ which depends only on A_1 and A_2 which can be true only for finitely many vectors $z \in \mathbb{Z}^{d_1}$ ([17], again, to be self-contained, this is reproduced in the appendix, Section 2.7). □

In [28], Köppe obtained the above finiteness result by a geometric description of the rational parts of mixed-integer test set elements when the integer parts are kept fixed. The analysis in [28] then also led to the conclusions we obtain in the following lemma.

Lemma 2.4.4 *Given vectors b, c, and a non-optimal feasible solution (z_0, q_0) to $A_1z + A_2q = b$, there exists $z_1 \in \mathcal{G}_{\mathbb{Z}}(A)$ from which an improving vector (z_1, q_1) to (z_0, q_0) can be reconstructed.*

Proof. Since (z_0, q_0) is not optimal there is an improving vector $(z', q') \in \mathcal{G}_{\text{MIP}}(A)$. Thus, an improving vector can be reconstructed from $z_1 := z' \in \mathcal{G}_{\mathbb{Z}}(A)$. It remains to show that we can find z_1 and a suitable q_1 algorithmically.

For every $z_1 \in \mathcal{G}_{\mathbb{Z}}(A)$ with $z_0 - z_1 \geq 0$ we try to find q_1 such that $(z_0 - z_1, q_0 - q_1)$ is feasible and such that the objective value $c^{\intercal}(z_0 - z_1, q_0 - q_1)$ is as small as possible. If $c^{\intercal}(z_0 - z_1, q_0 - q_1) < c^{\intercal}(z_0, q_0)$ then (z_1, q_1) is an improving vector. This problem is equivalent to the optimization problem

$$q_1 \in \arg\max_q \{c_z^{\intercal} z_1 + c_q^{\intercal} q : A_1(z_0 - z_1) + A_2(q_0 - q) = b, q_0 - q \geq 0, q \in \mathbb{R}^{d_2}\}. \quad (2.1)$$

Clearly, there is an improving vector for (z_0, q_0) in $\mathcal{G}_{MIP}(A)$ starting with z_1 if and only if (z_1, q_1) is an improving vector, that is, if and only if the above maximization problem 2.1 has a feasible solution with strictly positive cost function value. □

Corollary 2.4.5 *If for all $z_1 \in \mathcal{G}_{\mathbb{Z}}(A)$ the maximal value of (2.1) is non-positive then (z_0, q_0) is optimal.*

Proof. If the solution (z_0, q_0) is not optimal, then there has to exist an improving vector $(z_1, \bar{q}) \in \mathcal{G}_{\text{MIP}}(A_1|A_2)$. But for this z_1 the maximal value of (2.1) is at least $c_z^\intercal z_1 + c_q^\intercal \bar{q} > 0$, contradicting the assumption that no such z_1 exists. □

We need to solve many LP subproblems in order to reconstruct an improving vector. However, compared to the original mixed-integer problem these LP computations can be considered to be cheap. The advantages of our approach are clear. $\mathcal{G}_{\mathbb{Z}}(A)$ is always a finite set of vectors from which for every given $b \in \mathbb{R}^l$ improving vectors to non-optimal solutions can be reconstructed. Moreover, this approach allows us to use powerful LP solvers to take care of the continuous part of our mixed-integer problem once the set $\mathcal{G}_{\mathbb{Z}}(A)$ has been computed. Finally, note that this project-and-lift approach is not restricted to MIP Graver test sets. It can be used for any other type of mixed-integer test sets, as well.

2.4.3 Computation

In the following we show how to compute a superset of $\mathcal{G}_{\mathbb{Z}}(A)$. To this end, we assume that we have available an algorithm to solve the following problem.

Problem 2.4.6 *Let $A \in \mathbb{Z}^{l \times d}$. For any $b \in \mathbb{R}^l$ define $P_{A,b} := \{z : Az = b, z \in \mathbb{R}^d_+\}$. For given $b_1, b_2 \in \mathbb{R}^l$ decide, whether $P_{A,b_1+b_2} = P_{A,b_1} + P_{A,b_2}$, where $P_{A,b_1} + P_{A,b_2}$ denotes the Minkowski sum of P_{A,b_1} and P_{A,b_2}.*

To compute $\mathcal{G}_{\mathbb{Z}}(A)$, we will again employ the Completion Algorithm 1.3.1 by defining the necessary notions.

Definition 2.4.7 *Let $\bar{F}$ be a symmetric integer generating set for the integer lattice*

$$\{z \in \mathbb{Z}^k : \exists q \in \mathbb{R}^{d-k} \text{ with } (z, q) \in \ker_{\mathbb{Z}^k \times \mathbb{R}^{d-k}}(A)\}.$$

Let $F := \{(v, P_{(A_2|-A_2),-A_1 v}) : v \in \bar{F}\}$ be the input set to the completion algorithm. For better readability, we will write P_v instead of $P_{(A_2|-A_2),-A_1 v}$.

Thus, we apply the completion procedure to more complicated structures than vectors. Therefore, the test $f \neq 0$ in the Completion Algorithm 1.3.1 has to be replaced by $f \neq (0, P_0)$.

Definition 2.4.8 *We define* S-vectors$((u, P_u), (u', P_{u'})) := \{(u, P_u) \oplus (u', P_{u'})\}$, *where*

$$(u, P_u) \oplus (u', P_{u'}) := (u + u', P_{u+u'}).$$

We say that $(u', P_{u'})$ *reduces* (u, P_u), *or* $(u', P_{u'}) \sqsubseteq (u, P_u)$ *for short, if* $u' \sqsubseteq u$ *and* $P_u = P_{u'} + P_{u-u'}$. *In case of reducibility,* (u, P_u) *is reduced to*

$$(u, P_u) \ominus (u', P_{u'}) := (u - u', P_{u-u'}).$$

In order to prove termination of the algorithm below, the following sufficient condition to check $P_{A,b_1+b_2} = P_{A,b_1} + P_{A,b_2}$ will turn out very useful.

Lemma 2.4.9 *Let* $A \in \mathbb{Z}^{l\times d}$ *be of full row rank, and* $b_1, b_2 \in \mathbb{R}^l$, *such that* $P_{A,b_1} \neq \emptyset$ *and* $P_{A,b_2} \neq \emptyset$. *Moreover, suppose that for every invertible* $l \times l$ *submatrix* B *of* A *the vectors* $B^{-1}b_1$ *and* $B^{-1}b_2$ *lie in the same orthant of* $\mathbb{R}^l$. *Then* $P_{A,b_1+b_2} = P_{A,b_1} + P_{A,b_2}$.

A proof of this lemma can be found for example in [28]. We will now collect some results that will turn out useful when proving termination of the completion algorithm.

Corollary 2.4.10 *Let* $A \in \mathbb{Z}^{l\times d}$ *be of full row rank and denote by* $B_1, \ldots, B_M$ *all invertible* $l \times l$ *submatrices of* A. *Moreover, we define for each* $z \in \mathbb{Z}^d$ *the vector* $f(A, z) := (z, \det(B_1)B_1^{-1}(Az), \ldots, \det(B_M)B_M^{-1}(Az)) \in \mathbb{Z}^{d+Ml}$. *Then the relation* $(z, P_{A,Az}) \not\sqsubseteq (z', P_{A,Az'})$ *implies* $f(A, z) \not\sqsubseteq f(A, z')$.

Proof. Suppose that $f(A, z) \sqsubseteq f(A, z')$. Therefore, the vectors $f(A, z)$, $f(A, z')$, and $f(A, z'-z) = f(A, z') - f(A, z)$ all lie in the same orthant of $\mathbb{R}^{d+Ml}$. Thus, $z \sqsubseteq z'$, and the vectors $\det(B_i)B_i^{-1}(Az)$ and $\det(B_i)B_i^{-1}(A(z'-z))$ lie in the same orthant of $\mathbb{R}^l$ for all $i = 1, \ldots, M$. Hence, $P_{A,Az'} = P_{A,A(z'-z)} + P_{A,Az}$, by Lemma 2.4.9, and therefore $(z, P_{A,Az}) \sqsubseteq (z', P_{A,Az'})$. Thus, $(z, P_{A,Az}) \not\sqsubseteq (z', P_{A,Az'})$ implies $f(A, z) \not\sqsubseteq f(A, z')$, as claimed. □

Corollary 2.4.11 *Let* $A \in \mathbb{Z}^{l\times d}$ *and let* $\{u_1, u_2, \ldots\}$ *be a sequence in* $\mathbb{Z}^d$. *Then every sequence* $\{(u_1, P_{A,Au_1}), (u_2, P_{A,Au_2}), \ldots\}$ *with* $(u_i, P_{A,Au_i}) \not\sqsubseteq (u_j, P_{A,Au_j})$ *whenever* $i < j$, *is finite.*

Proof. Consider the sequence $\{f(A,u_1), f(A,u_2), \ldots\}$ of vectors in $\mathbb{Z}^{d+Ml}$, which satisfies $f(A,u_i) \not\sqsubseteq f(A,u_j)$ whenever $i<j$, by Corollary 2.4.10. Applying the Gordan-Dickson Lemma to the sequence

$$\{(f(A,u_1)^+, f(A,u_1)^-), (f(A,u_2)^+, f(A,u_2)^-), \ldots\} \subseteq \mathbb{Z}^{2(d+Ml)},$$

which satisfies $(f(A,u_i)^+, f(A,u_i)^-) \not\leq (f(A,u_j)^+, f(A,u_j)^-)$ whenever $i<j$, we conclude that this sequence and, thus, also the sequences $\{f(A,u_1), f(A,u_2), \ldots\}$ and $\{(u_1, P_{A,Au_1}), (u_2, P_{A,Au_2}), \ldots\}$ are finite. □

Proposition 2.4.12 *If the input set, the procedure* normalForm*, and the set* S-vectors *are defined as above, then the Completion Algorithm 1.3.1 terminates and computes a set G such that $\mathcal{G}_{\mathbb{Z}}(A) \subseteq \{u : (u,P_u) \in G\}$.*

Proof. In the course of the algorithm, a sequence of pairs in $G \setminus F$ is generated that satisfies the conditions of Corollary 2.4.11. Therefore, the algorithm terminates.

To show correctness, we have to prove that

$$\bar{G} := \{(\bar{u},\bar{v}) \in \mathbb{Z}^{d_1} \times \mathbb{R}^{d_2} : (\bar{u}, P_{\bar{u}}) \in G, (\bar{v}^+, \bar{v}^-) \in P_{\bar{u}}\}$$

has the positive sum property with respect to $\ker_{\mathbb{Z}^{d_1}\times\mathbb{R}^{d_2}}(A)$. To this end, we construct for arbitrarily chosen $(z,q) \in \ker_{\mathbb{Z}^{d_1}\times\mathbb{R}^{d_2}}(A)$ a representation

$$(z,q) = \sum_{(u,P_u)\in G} \alpha_u (u, q_u),$$

where $\alpha_u \in \mathbb{Z}_+$, $(q_u^+, q_u^-) \in P_u$, and $\alpha_u(u,q_u) \sqsubseteq (z,q)$ for all u. Since G contains F, at least an integer linear combination

$$(z,q) = \sum_{(u,P_u)\in G} \alpha_u (u, q_u)$$

with $\alpha_u \in \mathbb{Z}_+$ and $(q_u^+, q_u^-) \in P_u$ for all u, is possible. For each such integer linear combination consider the value $\sum \alpha_u \|(u,q_u)\|_1$, which is always non-negative. Thus, this sum is bounded from below by some non-negative infimum. Let us assume for the moment that there is a linear integer combination $\sum \alpha_u(u,q_u)$ that attains this infimum.

If $\sum_u \alpha_u \|(u,q_u)\|_1 = \|(z,q)\|_1$ then all vectors (u,q_u) with $\alpha_u > 0$ lie in the same orthant as (z,q). Hence $\alpha_u(u,q_u) \sqsubseteq (z,q)$ for all u with $\alpha_u > 0$. But since (z,q) was chosen arbitrarily, this implies that $\bar{G}$ has the positive sum property with respect to $\ker_{\mathbb{Z}^{d_1}\times\mathbb{R}^{d_2}}(A)$ and thus, $\bar{G}$ contains $\mathcal{G}_{\mathrm{MIP}}(A)$. The claim then follows.

Assume on the contrary that we have $\sum_u \alpha_u \|(u,q_u)\|_1 > \|(z,q)\|_1$ for such a minimal combination. Then there exist u_1, u_2 with $\alpha_{u_1} > 0$ and $\alpha_{u_2} > 0$, and a component m, such that $(u_1,q_{u_1})^{(m)}(u_2,q_{u_2})^{(m)} < 0$. Thus,

$$\|(u_1+u_2, q_{u_1}+q_{u_2})\|_1 < \|(u_1,q_{u_1})\|_1 + \|(u_2,q_{u_2})\|_1.$$

During the run of the algorithm, $(u_1+u_2, P_{u_1+u_2})$ was reduced to $(0,P_0)$ by elements (v_1,P_{v_1}), ..., $(v_s,P_{v_s}) \in G$, giving a representation $u_1+u_2 = \sum_{j=1}^s v_j$, $v_j \sqsubseteq u_1+u_2$, and $P_{u_1+u_2} = P_0 + P_{v_1} + \ldots + P_{v_s}$. The latter implies that there are vectors q_{v_j} with $(q_{v_j}^+, q_{v_j}^-) \in P_{v_j}$, $j = 1,\ldots,s$, and $(q_{v_0}^+, q_{v_0}^-) \in P_0$ such that

$$(q_{u_1}+q_{u_2})^+ = \sum_{j=0}^s q_{v_j}^+$$

and

$$(q_{u_1}+q_{u_2})^- = \sum_{j=0}^s q_{v_j}^-.$$

Thus $q_{v_j} \sqsubseteq q_{u_1}+q_{u_2}$, $j = 0,\ldots,s$, and therefore,

$$\|q_{u_1}+q_{u_2}\|_1 = \sum_{j=0}^s \|q_{v_j}\|_1.$$

Altogether, we obtain

$$\sum_{j=0}^s \|(v_j,q_{v_j})\|_1 = \|\sum_{j=0}^s (v_j,q_{v_j})\|_1 = \|(u_1+u_2,q_{u_1}+q_{u_2})\|_1 < \|(u_1,q_{u_1})\|_1 + \|(u_2,q_{u_2})\|_1,$$

where $v_0 = 0$. Now rewrite

$$(z,q) = \sum_{(u,P_u)\in G} \alpha_u (u,q_u)$$

as

$$(z,q) = \sum_{(u,P_u)\in G, u\neq u_1,u_2} \alpha_u(u,q_u) + (\alpha_{u_1}-1)(u_1,q_{u_1}) + (\alpha_{u_2}-1)(u_2,q_{u_2}) + \sum_{j=1}^s (v_j,q_{v_j})$$

where

$$\sum_{u\neq u_1,u_2} \alpha_u \|(u,q_u)\|_1 + (\alpha_{u_1}-1)\|(u_1,q_{u_1})\|_1 + (\alpha_{u_2}-1)\|(u_2,q_{u_2})\|_1 + \sum_{j=1}^s \|(v_j,q_{v_j})\|_1$$

is strictly less than $\sum_u \alpha_u \|(u,q_u)\|_1$, in contradiction to the minimality assumption on the integer linear combination $\sum_{(u,P_u)\in G} \alpha_u(u,q_u)$.

It remains to show that the infimum of $\sum_{(u,P_u)\in G} \alpha_u \|(u,q_u)\|_1$ is indeed attained by some integer linear combination. This can be seen as follows. Since z, q, and all u with $(u,P_u) \in G$ are given, we have to find $\alpha_u \in \mathbb{Z}_+$ and vectors $q_u \in \mathbb{R}^{d-k}$ with $(q_u^+, q_u^-) \in P_u$ such that $\sum_{(u,P_u)\in G} \alpha_u \|(u,q_u)\|_1$ attains the infimum value.

First, since there is at least one such linear combination, we have an upper bound K for the infimum. Thus, there are at most finitely many choices for the non-negative integers α_u such that $\sum_{(u,P_u)\in G} \alpha_u \|u\|_1 \leq K$. It remains to show that for fixed integers α_u the infimum of $\sum_{(u,P_u)\in G} \alpha_u \|(u,q_u)\|_1$ is indeed attained for some choice of the vectors q_u. Then, also the global infimum, as the least value of the finitely many minimal values of $\sum_{(u,P_u)\in G} \alpha_u \|(u,q_u)\|_1$, wherein the α_u are fixed, is attained by some combination.

Therefore, suppose in the following that the integers α_u are fixed and that there exists at least one linear integer representation of (z,q) for this fixed choice of the numbers α_u. Since α_u and all u are fixed, we have to minimize $\sum_{(u,P_u)\in G} \alpha_u \|q_u\|_1$ where $\sum_{(u,P_u)\in G} \alpha_u q_u = q$ and $A_2 q_u = -A_1 u$, for all u with $(u,P_u) \in G$. Writing $q_u = q_u^+ - q_u^-$ we obtain a linear minimization problem with non-empty feasible region, whose objective is bounded from below by 0. Thus, the minimum is attained for some choice of the vectors q_u. This concludes the proof. □

2.5 Termination of Augmentation Algorithm

Let us now have a look at termination of the Augmentation Algorithm 0.0.2 in the LP, IP, and MIP situations. Assume that we are given a feasible solution z_0 to our problem. Moreover, assume that $(LP)_{c,b}$ is bounded with respect to c, which can be checked by standard techniques from LP. Therefore, if we assume integer (or rational) entries in A and b, also $(IP)_{c,b}$ and $(P)_{c,b}$ are bounded with respect to c, see Schrijver [42].

In the LP case, circuits provide improving directions to non-optimal solutions. However, a zig-zagging effect, even to a non-optimal point, is possible and we have to take some care on how to choose the next circuit for improvement.

Example. Consider the problem

$$\min\{z_1 + z_2 - z_3 : 2z_1 + z_2 \leq 2, z_1 + 2z_2 \leq 2, z_3 \leq 1, (z_1,z_2,z_3) \in \mathbb{R}^3_{\geq 0}\}$$

with optimal solution $(0,0,1)$. Introducing slack variables z_4, z_5, z_6 we obtain the

problem $\min\{c^\intercal z : Az = (2,2,1)^\intercal, z \in \mathbb{R}^6_{\geq 0}\}$ with $c^\intercal = (1,1,-1,0,0,0)$ and

$$A = \begin{pmatrix} 2 & 1 & 0 & 1 & 0 & 0 \\ 1 & 2 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

$\ker_{\mathbb{R}^6}(A)$ is generated over $\mathbb{R}$ by the vectors $(1,0,0,-2,-1,0)$, $(0,1,0,-1,-2,0)$, and $(0,0,1,0,0,-1)$. We obtain $(1,0,0,-2,-1,0)$, $(0,1,0,-1,-2,0)$, $(1,-2,0,0,3,0)$, $(2,-1,0,-3,0,0)$, $(0,0,1,0,0,-1)$ together with their negatives as the circuits of A. The improving directions are given by all circuits v for which $c^\intercal v > 0$. These are the following: $(1,0,0,-2,-1,0)$, $(0,1,0,-1,-2,0)$, $(-1,2,0,0,-3,0)$, $(2,-1,0,-3,0,0)$, $(0,0,-1,0,0,1)$.

Now start with the feasible solution $z_0 = (0,1,0,1,0,1)$. Going along the directions $(0,1,0,-1,-2,0)$ and $(0,0,-1,0,0,1)$ as far as possible, we immediately arrive at $(0,0,1,2,2,0)$ which corresponds to the desired optimal solution $(0,0,1)$ of our problem. However, alternatively choosing only the vectors $(-1,2,0,0,-3,0)$ and $(2,-1,0,-3,0,0)$ as improving directions, the augmentation process does not terminate. In our original space $\mathbb{R}^3$ this reads as

$$(0,1,0) \to \left(\frac{1}{2},0,0\right) \to \left(0,\frac{1}{4},0\right) \to \left(\frac{1}{8},0,0\right) \to \left(0,\frac{1}{16},0\right) \to \dots$$

clearly showing the zig-zagging effect to the non-optimal point $(0,0,0)$. Thus, we have to impose certain constraints on the circuit which is chosen next in the augmentation algorithm. □

To ensure termination of the Augmentation Algorithm 0.0.2 we split the augmentation process into two phases.

Algorithm 2.5.1 *(Augmentation Strategy to reach Optimum in finitely many Steps)*

1. *In phase* 1 *only those circuits are allowed for improvement which lead to a better feasible solution with an additional zero entry. Repeat this step until there is no circuit with this property and go to phase* 2.

2. *In phase* 2 *any appropriate circuit is chosen for improvement. Stop, if there is no such circuit, otherwise improve and go back to phase* 1.

Proposition 2.5.2 *Under the assumption that the $(LP)_{c,b}$ is bounded with respect to c, the LP Augmentation Algorithm 0.0.2 which uses the Augmentation Strategy 2.5.1 terminates with an optimal solution.*

Proof. At the end of phase 1 the current feasible solution z' has to be optimal with respect to all feasible solutions whose support is contained in $\mathrm{supp}(z')$, that is, there is no feasible solution z'' with $\mathrm{supp}(z'') \subseteq \mathrm{supp}(z')$ and $c^\intercal z'' < c^\intercal z'$. Assume on the contrary that there is such a solution z''. Since $z' - z'' \in \ker_{\mathbb{R}^d}(A)$, and by the positive sum property of the set of circuits with respect to $\ker_{\mathbb{R}^d}(A)$, there exists some circuit q with $\mathrm{supp}(q) \subseteq \mathrm{supp}(z' - z'') \subseteq \mathrm{supp}(z')$ and $c^\intercal q > 0$. This circuit, however, improves z': Choose the largest positive scalar α such that $z' - \alpha q$ is still feasible. This scalar is finite since $LP_{c,b}$ is bounded with respect to the given cost function. We conclude that $z' - \alpha q$ contains an additional zero entry and $c^\intercal(z' - \alpha q) < c^\intercal z'$, contradicting the fact that the first phase finished with z'.

Now, if z' is optimal then the algorithm terminates in phase 2. Otherwise, the zero pattern of z' will never reappear after z' was improved in phase 2. Thus, no zero pattern reappears at the end of the first phase and the augmentation process has to terminate.

The feasible solution upon termination has to be optimal since in the second phase no augmenting circuit was found. The claim now follows by the universal test set property of circuits. □

Finally, let us look at the IP and the MIP situations. Here, the Augmentation Algorithm 0.0.2 always terminates with an optimal solution.

Proposition 2.5.3 *If the relaxed problem $(LP)_{c,b}$ is bounded with respect to c, the IP Augmentation Algorithm 0.0.2 terminates with an optimal solution.*

Proof. Let $\epsilon := \min\{|c^\intercal g| : g \in \mathcal{G}_{\mathrm{IP}}(A), c^\intercal g \neq 0\}$. Then each time we improve a given feasible solution z_0 by an element $g \in \mathcal{G}_{\mathrm{IP}}(A)$ to $z_0 - \alpha g$ the cost function value drops by at least $\alpha(c^\intercal g) \geq \epsilon > 0$ since α is a positive integer. Since $(LP)_{c,b}$ is bounded with respect to c this can happen only finitely often and the IP augmentation algorithm has to terminate. The returned solution has to be optimal, since it cannot be improved along some test set direction. □

Proposition 2.5.4 *If the relaxed problem $(LP)_{c,b}$ is bounded with respect to c, the MIP Augmentation Algorithm 0.0.2 terminates with an optimal solution.*

Proof. First transform the problem $(P)_{c,b}$ by Gaussian elimination into an equivalent optimization problem $(P)'_{c,b'}$ with problem matrix

$$A' := \begin{pmatrix} A'_1 & A'_2 \\ A''_1 & 0 \end{pmatrix},$$

wherein A_2' has full row rank. Note that the right-hand side b will change to some b' as well, but the cost function c remains fixed. Moreover, since the kernel of the problem matrix does not change, the finite set $\mathcal{G}_\mathbb{Z}(A_1|A_2)$ remains unchanged after this transformation.

Now suppose on the contrary that the augmentation process does not terminate, that is, it generates an infinite sequence of feasible solutions to $(P)'_{c,b'}$ with strictly decreasing objective value. Divide this sequence into consecutive subsequences of length $K+1$, where K denotes the number of invertible submatrices B of A_2' of full rank. As we will see, within each of these subsequences the cost function value drops by at least some constant $\epsilon > 0$ depending only A_1', A_1'', A_2', and c. Since $(LP)_{c,b'}$ is bounded with respect to c this can happen only finitely often, and the MIP Augmentation Algorithm 0.0.2 has to terminate after finitely many steps.

Since A' is an integer matrix, the set $\{q \in \mathbb{R}_+^{d_2} : A'(z,q) = b'\}$ is a polyhedron for every fixed $z \in \mathbb{Z}^{d_1}$. After each augmentation step the continuous part q_0 of the current feasible solution (z_0, q_0) is a vertex of $\{q \in \mathbb{R}_+^{d_2} : A'(z_0,q) = b'\}$ (or a point with the same cost function value). Thus, after possible rearrangement of continuous variables, we can write A_2' as $(B|\bar{A}_2)$ where B is an invertible $l \times l$-matrix and the optimal point is given by $(z_0, B^{-1}(b' - A_1'z_0), 0)$. Let $c = (c_z, c_{q,1}, c_{q,2})$ be divided analogously. Then the cost function value of $(z_0, B^{-1}(b' - A_1'z_0), 0)$ is given by

$$c_z^\intercal z_0 + c_{q,1}^\intercal B^{-1}(b' - A_1'z_0) = (c_z^\intercal - c_{q,1}^\intercal B^{-1}A_1')z_0 + c_{q,1}^\intercal B^{-1}b' =: \tilde{c}_B^\intercal z_0 + \tilde{c}_{0,B}$$

where $\tilde{c}_B$ and $\tilde{c}_{0,B}$ depend only on the given problem data and on the specific choice of B.

Consider a sequence of $K+1$ consecutive feasible solutions as generated in the MIP Augmentation Algorithm 0.0.2. By the pigeon-hole principle there are two solutions $(z_1, B^{-1}(b' - A_1'z_1), 0)$ and $(z_2, B^{-1}(b' - A_1'z_2), 0)$ whose continuous parts are determined by the same submatrix B of A_2' as already indicated by the notation. Moreover, let the second solution have a better cost function value than the first one.

Thus $0 < \tilde{c}_B^\intercal z_1 + \tilde{c}_{0,B} - (\tilde{c}_B^\intercal z_2 + \tilde{c}_{0,B}) = \tilde{c}_B^\intercal(z_1 - z_2)$. In the MIP augmentation algorithm the vector $z_1 - z_2$ was represented as a sum $\sum_{i=1}^{\leq K} g_i$ of at most K elements from $\mathcal{G}_\mathbb{Z}(A') = \mathcal{G}_\mathbb{Z}(A)$, since z_2 was obtained from z_1 by at most K augmentation steps. Since $\mathcal{G}_\mathbb{Z}(A')$ is finite, there are only finitely many possible positive values of $\tilde{c}_B^\intercal(\sum_{i=1}^{\leq K} g_i) > 0$. Thus, there exists a constant $\epsilon_B > 0$ depending only on the choices of B and the given data A_1', A_1'', A_2', and c, such that

$$\tilde{c}_B^\intercal(z_1 - z_2) = \tilde{c}_B^\intercal \left(\sum_{i=1}^{\leq K} g_i \right) \geq \epsilon_B > 0$$

for all choices of the $g_i \in \mathcal{G}_{\mathbb{Z}}(A')$. Therefore, the cost function value drops by at least $\epsilon_B > 0$ between the two mixed-integer solutions $(z_1, B^{-1}(b' - A'_1 z_1), 0)$ and $(z_2, B^{-1}(b' - A'_1 z_2), 0)$ in the augmentation algorithm.

Since there are only finitely many invertible submatrices B of A'_2 of full rank, the value $\epsilon := \min_B\{\epsilon_B\} > 0$ is well defined and depends only on A'_1, A''_1, A'_2, and c, the given problem data. Moreover, we proved that within each subsequence of length $K+1$ the cost function value drops by at least $\epsilon_B \geq \epsilon > 0$, which implies that the MIP Augmentation Algorithm 0.0.2 has to terminate provided that the relaxed problem $(LP)_{c,b'}$ is bounded with respect to c. The returned solution has to be optimal, since it cannot be improved along some test set direction. □

2.6 Feasible Initial Solutions

So far we have addressed the problem of improving a given feasible solution for $(\mathrm{P})_{c,b}$ using directions given by test set elements. In doing so, the optimization problem can be solved in finitely many augmentation steps. But, particularly in the IP and MIP cases, it is often already a very hard problem to find an initial feasible solution at all.

However, one can at least compute a (mixed-) integer solution to $Az = b$ in polynomial time, ignoring the lower bounds 0 on the variables [42]. Computing a solution in the LP case can be done for example by Gaussian elimination or by even faster methods.

In the following we will demonstrate how universal test sets can be used to transform any given solution to $Az = b, z \in \mathbb{X}$, into a feasible solution of $(\mathrm{P})_{c,b}$, that is, a solution to $Az = b, z \in \mathbb{X}$, satisfying also the lower bounds $z \geq 0$. The proposed algorithm always terminates and returns either a feasible solution or the answer that no such solution exists.

Algorithm 2.6.1 *(Algorithm to find a Feasible Solution)*

Input: a solution $z_1 \in \mathbb{X}$ to $Az = b$, a universal test set $\mathcal{T}$ for $(P)_{c,b}$

Output: a feasible solution or "FAIL" if no such solution exists

while there exist $t \in \mathcal{T}$ and $\alpha \in \mathbb{R}_{>0}$ such that $z_1 - \alpha t \in \mathbb{X}$, $\|(z_1 - \alpha t)^-\|_1 < \|z_1^-\|_1$, and $(z_1 - \alpha t)^{(k)} \geq 0$ whenever $z_1^{(k)} \geq 0$ do

$z_1 := z_1 - \alpha t$, where α is chosen maximal such that all three conditions are still satisfied

if $\|z_1^-\|_1 > 0$ then return "FAIL" else return z_1

In the MIP case we have to decide for every $g_z \in \mathcal{G}_{\mathbb{Z}}(A)$ whether we can find g_q and a scalar $\alpha > 0$ such that $t := (g_z, g_q)$ has the desired properties. For this we distinguish two cases. Either we have $g_z = 0$ or $g_z \neq 0$. In both cases we can fix $\alpha = 1$ and are left with a pure LP problem: Find $(g_z, g_q) \in \ker_{\mathbb{X}}(A)$ minimizing $\|(z_1 - \alpha g)^-\|_1 = \|z_1 - (g_z, g_q)^-\|_1$, where z_1 and g_z are given.

Lemma 2.6.2 *Algorithm 2.6.1 satisfies its specifications.*

Proof. Suppose that the algorithm terminates and that at the end z_1 is still infeasible although there is some feasible solution z_0 to our problem. Consider the optimization problem

$$\min\{ \sum_{i:z_1^{(i)}<0} (-z)^{(i)} : Az = b + Az_1^-, z \in \mathbb{X}_+ \}. \tag{2.2}$$

Then z_1^+ is feasible for problem (2.2) and it admits an objective function value of 0. Moreover, $z_0 + z_1^-$ is feasible, too, with strictly negative objective function value, since $(z_0 + z_1^-)^{(k)} > 0$ whenever $z_1^{(k)} < 0$ (and $z_0^{(k)} \geq 0$). By the universal test set property of $\mathcal{T}$, there have to exist a vector $t \in \mathcal{T}$ and a scalar $\alpha > 0$ such that $z_1^+ - \alpha t$ is feasible for problem (2.2) with strictly negative objective function value. In other words, $z^+ - \alpha t \geq 0$ and $\sum_{k:z_1^{(k)}<0} (-t)^{(k)} > 0$. Therefore, $(-t)^{(k)} > 0$ for at least some $k = k_0$ with $z_1^{(k_0)} < 0$. Moreover $(-t)^{(k)} \geq 0$ whenever $z_1^{(k)} < 0$, since $(z^+ - \alpha t)^{(k)} \geq 0$, $\alpha > 0$, and $(z_1^+)^{(k)} = 0$. Thus, $\|(z_1 - \alpha t)^-\|_1 < \|z_1^-\|_1$ and if $z_1^{(k)} \geq 0$ then also $(z_1 - \alpha t)^{(k)} = (z_1^+ - \alpha t)^{(k)} \geq 0$. Thus, the vector t and the scalar α satisfy the conditions of the while-loop which is a contradiction to z_1 being the output of the Algorithm 2.6.1. □

Lemma 2.6.3 *Algorithm 2.6.1 terminates in the IP and MIP cases. The algorithm terminates in the LP case if the following strategy similar to the Augmentation Strategy 2.5.1 is followed.*

1. *In phase* 1 *only those circuits are allowed for improvement which lead to a better solution with an additional zero entry. Repeat this step until there is no circuit with this property and go to phase* 2.

2. *In phase* 2 *any appropriate circuit is chosen for improvement. Stop, if there is no such circuit, otherwise improve and go back to phase* 1.

Proof. The algorithm terminates in the IP case since in each step $\|z_1^-\|_1$ drops by at least 1. The algorithm terminates in the LP case since, analogously to the termination

proof of the LP Augmentation Strategy 2.5.1, every support of z_1 appears at most once at the end of phase 1.

Suppose that the algorithm does not terminate in the MIP case and that it generates an infinite sequence $z_1, z_2, \ldots$ of points in $\mathbb{X}$. Note that all non-negative components of a point z_j remain non-negative for all subsequent points during the run of the above algorithm. Thus, since the algorithm does not terminate, there is a number N such that $\emptyset \neq \operatorname{supp}(z_N^-) = \operatorname{supp}(z_j^-)$ for all $j \geq N$. Now consider the problem

$$\min\{ \sum_{i: z_N^{(i)} < 0} (-z)^{(i)} : Az = b + Az_N^-, z \in \mathbb{X}_+ \}. \tag{2.3}$$

As can be seen from the proof of Lemma 2.6.2, the Algorithm 2.6.1 produces a sequence $z_1, z_2, \ldots$ of points in $\mathbb{X}$ such that $z_1^- \geq z_2^- \geq \ldots$, that is, $z_i^- \geq z_j^-$ whenever $i < j$. Thus we have $z_j + z_N^- = z_j^+ + (z_N^- - z_j^-) \geq z_j^+ \geq 0$ and we can conclude that $z_j + z_N^-$, $j \geq N$, are all feasible solutions of (2.3). Since by our assumption on N, we have $z_j^{(k)} < 0$ whenever $z_N^{(k)} < 0$, all these solutions have a strictly positive objective value. However, since the algorithm does not terminate, we can use a similar argument as in the termination proof of the MIP augmentation algorithm, Proposition 2.5.4, to show that the cost function value drops below any given value within a finite number of steps. Thus, in particular, the cost function value becomes eventually negative for some z_j with $j \geq N$. From this contradiction we can conclude that the algorithm terminates in the MIP case, too. □

Remark 2.6.4 *In order to find a feasible solution to $(P)_{c,b}$ we have to construct improving vectors to problems of type (2.2). Thus, we can and we will focus our attention in the subsequent exposition to finding improving directions.*

2.7 Appendix

Lemma 2.7.1 *(Foroudi and Graver [17], General Decomposition Theorem)*

$\mathcal{G}_{MIP}(A)$ has the positive sum property with respect to $\ker_{\mathbb{X}}(A)$.

Proof. Suppose that $(z, q) \in \ker_{\mathbb{X}}(A)$ cannot be written as a positive linear combination of elements in $\mathcal{G}_{\mathrm{MIP}}(A)$. Thus, $(z, q) \notin \mathcal{G}_{\mathrm{MIP}}(A)$. We know that $\|z\|_1 > 0$ since $\mathcal{G}_{\mathrm{LP}}(A_2)$ has the positive sum property with respect to $\ker_{\mathbb{R}^{d_2}}(A_2)$. From all such vectors $(z, q) \in \ker_{\mathbb{X}}(A)$ choose one such that $\|z\|_1 + |\operatorname{supp}(q)|$ is minimal.

Now suppose that there is some circuit $q_1 \in \mathcal{G}_{\mathrm{LP}}(A_2)$ such that $\mathrm{supp}(q_1) \subseteq \mathrm{supp}(q)$ and such that $(0, q_1)$ lies in the same orthant as (z, q). Choose the largest scalar $\alpha_1 \in \mathbb{R}_{>0}$ such that $(z, q - \alpha_1 q_1)$ still belongs to the same orthant as (z, q). Therefore, $\mathrm{supp}(q - \alpha_1 q_1) \subsetneq \mathrm{supp}(q)$ and consequently

$$\|z\|_1 + |\operatorname{supp}(q - \alpha_1 q_1)| < \|z\|_1 + |\operatorname{supp}(q)|.$$

By the minimality required on $\|z\|_1 + |\operatorname{supp}(q)|$ we can conclude that there is a linear representation $(z, q - \alpha_1 q_1) = \sum \beta_j (z_j, q_j)$ where for all j we have $\beta_j(z_j, q_j) \in \mathbb{X}$, $\beta_j(z_j, q_j) \sqsubseteq (z, q - \alpha_1 q_1)$, and $\beta_j \in \mathbb{R}_{>0}$. Hence $(z, q) = \alpha_1 q_1 + \sum \beta_j (z_j, q_j)$ is a valid representation of (z, q) in contrast to our initial assumption. Thus, we may assume that there is no vector $(0, q_1) \in \ker_{\mathbb{X}}(A)$ with $(0, q_1) \sqsubseteq (z, q)$.

From $(z, q) \notin \mathcal{G}_{\mathrm{MIP}}(A)$ we conclude that there is some $(z', q') \in \ker_{\mathbb{X}}(A) \setminus \{0\}$ with $(z', q') \sqsubseteq (z, q)$, $(z', q') \neq (z, q)$. Hence $(z, q) = (z', q') + (z - z', q - q')$, where also $(z - z', q - q') \sqsubseteq (z, q)$, $(z - z', q - q') \neq (z, q)$. This implies in particular, that $\mathrm{supp}(q') \subseteq \mathrm{supp}(q)$ and $\mathrm{supp}(q - q') \subseteq \mathrm{supp}(q)$.

But neither $z' = 0$ nor $z - z' = 0$, by construction. Therefore, we have $\|z'\|_1 < \|z\|_1$ and $\|z - z'\|_1 < \|z\|_1$. Thus, by the minimality assumption on $\|z\|_1 + |\operatorname{supp}(q)|$, both (z', q') and $(z - z', q - q')$ can be written as valid positive linear combinations of elements from $\mathcal{G}_{\mathrm{MIP}}(A)$, all of which belong to the same orthant as (z', q') and as $(z - z', q - q')$, respectively.

Substituting these representations into $(z, q) = (z', q') + (z - z', q - q')$ gives a valid positive linear combination representing (z, q), contradicting our initial assumption that no such representation exists. □

Lemma 2.7.2 *(Foroudi and Graver [17], Lemma 13)*

Given $A = (A_1|A_2)$ the following inequality holds for any $(z_0, q_0) \in \mathcal{G}_{MIP}(A_1|A_2)$:

$$\|z_0\|_1 \leq \sum_{(z,q) \in \mathcal{G}_{LP}(A)} \|z\|_1.$$

Proof. Let $(z_0, q_0) \in \mathcal{G}_{MIP}(A_1|A_2) \subseteq \ker_{\mathbb{R}^d}(A)$. The positive sum property of $\mathcal{G}_{\mathrm{LP}}(A)$ with respect to $\ker_{\mathbb{R}^d}(A)$ yields a finite linear representation $(z_0, q_0) = \sum \alpha_i (z_i, q_i)$ where $(z_i, q_i) \in \mathcal{G}_{\mathrm{LP}}(A)$, $\alpha_i (z_i, q_i) \sqsubseteq (z_0, q_0)$, and $\alpha_i \in \mathbb{R}_{>0}$.

Suppose that there exists a summand $\alpha_j (z_j, q_j)$ in this representation with $z_j \neq 0$ and $\alpha_j > 1$. From $(z_j, q_j) \in \mathcal{G}_{\mathrm{LP}}(A) \subseteq \mathbb{Z}^d$ we conclude that

$$(z_j, q_j) \in \mathbb{X} \qquad \text{and} \qquad (\alpha_j - 1)(z_j, q_j) + \sum_{i \neq j} \alpha_i (z_i, q_i) = (z_0, q_0) - (z_j, q_j) \in \mathbb{X}.$$

By construction, $z_j \neq 0$ and $z_0 - z_j \neq 0$, $(z_j, q_j), (z_0 - z_j, q_0 - q_j) \sqsubseteq (z_0, q_0)$, and $(z_0, q_0) = (z_j, q_j) + (z_0 - z_j, q_0 - q_j)$. Therefore, $(z_0, q_0) \notin \mathcal{G}_{\mathrm{MIP}}(A_1|A_2)$. Thus, if $(z_0, q_0) \in \mathcal{G}_{\mathrm{MIP}}(A_1|A_2)$ then any linear representation $(z_0, q_0) = \sum \alpha_i(z_i, q_i)$ with $(z_i, q_i) \in \mathcal{G}_{\mathrm{LP}}(A)$, $\alpha_i(z_i, q_i) \sqsubseteq (z_0, q_0)$, and $\alpha_i \in \mathbb{R}_{>0}$, must satisfy $\alpha_j \leq 1$ whenever $z_j \neq 0$. Consequently,

$$\|z_0\|_1 = \sum_i \|\alpha_i z_i\|_1 \leq \sum_{i:z_i \neq 0} \|z_i\|_1 + \sum_{i:z_i=0} \|\alpha_i z_i\|_1 = \sum_{i:z_i \neq 0} \|z_i\|_1 \leq \sum_{(z,q) \in \mathcal{G}_{\mathrm{LP}}(A)} \|z\|_1$$

and the claim is proved. □

With the additional information that z_0 belongs to some given orthant $\mathbb{O}_j$ the above estimate can be strengthened to

$$\|z_0\|_1 \leq \sum_{(z,q) \in \mathcal{G}_{\mathrm{LP}}(A), z \in \mathbb{O}_j} \|z\|_1.$$

2.8 Conclusions

We have discussed the positive sum property, a property inherent to Graver test sets in LP, IP, and MIP. Already Graver [19] showed that this property implies the universal test set property. We added to this analysis by presenting two criteria which allow an algorithmic test of the positive sum property with respect to $\ker_{\mathbb{R}^d}$ or with respect to an integer lattice. These criteria led us immediately to a completion algorithm: an initial set of vectors is completed with respect to the positive sum property by adding new vectors to the set as long as necessary. Critical pair/completion procedures arise in automated theorem proving, polynomial ideal theory, and in the solution of word problems in universal algebras, see Buchberger [7] for a survey.

Then we have demonstrated how fixed (finite) upper bounds on variables can be used to truncate the Graver test set of $(\mathrm{IP})_{c,b}$ and to speed up its computation. The presented algorithm can also be used to compute the unique minimal Hilbert basis of $\ker_{\mathbb{R}^d}(A) \cap \mathbb{R}^d_+$ if the input set is chosen accordingly.

Via the positive sum property the notion of a Graver test sets can be extended to the mixed-integer situation. Although these test sets need not be finite in general we have identified a finite set of integer vectors from which an improving vector can be computed. In order to do so, one has to solve a finite number of LP's.

When optimizing with the help of test sets, attention has to be paid in the LP situation. An example shows that simple augmentation may result in zig-zagging towards a non-optimal point. We have presented a strategy to prevent this.

It is usually already a hard problem to find an initial feasible point of $(\mathrm{P})_{c,b}$ when some or all variables are required to be integer. Although there are algorithmic alternatives, we have shown that universal test sets contain enough information to solve this problem, too. Thus, once a universal test set is available, the problem $(\mathrm{P})_{c,b}$ can be solved entirely by test set methods.

In the following we will focus our attention to two- and to multi-stage stochastic programs whose problem matrices are highly structured. As we will see, the problem structure can be exploited for a novel decomposition approach. The presented algorithms also follow the pattern of a completion procedure.

Chapter 3

Decomposition of Test Sets in Two-Stage Stochastic Programming

As we have seen in the previous chapter, given solvability, the optimization problem $(\mathrm{IP})_{c,b}$ can be solved to optimality with the help of augmenting vectors from the corresponding Graver test set. Theoretically, this procedure can be used to solve stochastic integer programs (4), page 13, as well. However, due to the huge amount of stored information, Graver test sets are quite large already for small problems. Therefore, a direct test set approach to (4) is not advisable.

As we will see, the block angular structure of the problem matrix in (4) induces a symmetry structure on the elements of the Graver basis, telling us, that these test set vectors are formed by a comparably small number of building blocks. We show that these building blocks can be computed without computing the Graver test set of (4) itself and that we can construct an improving vector to a given non-optimal feasible solution to (4), scenario by scenario, using building blocks only. Incorporating this into the Augmentation Algorithm 0.0.2, we find an optimal solution with comparably small effort, once the building blocks have been computed.

We present the decomposition approach for the pure integer case. The same decomposition ideas, however, readily extend to the continuous and the mixed-integer situations.

To study Graver test sets of (4) we consider the matrix

$$A_N := \begin{pmatrix} A & 0 & 0 & \cdots & 0 \\ T & W & 0 & \cdots & 0 \\ T & 0 & W & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ T & 0 & 0 & \cdots & W \end{pmatrix}$$

together with the objective function vector $c = (c_0, c_1, \ldots, c_N)^\intercal := (h, \pi_1 q, \ldots, \pi_N q)^\intercal$ and the right-hand side $b = (a, \xi^1, \ldots, \xi^N)^\intercal$, where the subscript N corresponds to the number of scenarios, that is, the number of T's and W's used. Problem (4) then may be written as $\min\{c^\intercal z : A_N z = b, z \in \mathbb{Z}_+^{d_N}\}$ with $d_N = m + Nn$ and m, n as in (1)-(3), see the introductory chapter. We assume all entries in A, T, and W to be integer.

When referring to components of z, the notation $z = (u, v_1, \ldots, v_N)$ will be used throughout. Herein, u corresponds to the first-stage and is always of dimension m, whereas $v_1, \ldots, v_N$ are the second-stage vectors whose dimension is n.

3.1 Building Blocks of Graver Test Sets

The following simple observation is the basis for the decomposition of test set vectors presented below.

Lemma 3.1.1 *$(u, v_1, \ldots, v_N) \in \ker_{\mathbb{Z}^{d_N}}(A_N)$ if and only if $(u, v_i) \in \ker_{\mathbb{Z}^{d_1}}(A_1)$, for $i = 1, \ldots, N$.*

Proof. The claim follows from $0 = A_N z = (Au, Tu + Wv_1, \ldots, Tu + Wv_N)$. □

This guarantees that by permuting the v_i we do not leave $\ker_{\mathbb{Z}^{d_N}}(A_N)$. Moreover, every $\sqsubseteq$-minimal element of $\ker_{\mathbb{Z}^{d_N}}(A_N) \setminus \{0\}$ will always be transformed into another $\sqsubseteq$-minimal element of $\ker_{\mathbb{Z}^{d_N}}(A_N) \setminus \{0\}$. Thus, a Graver test set vector is transformed into a Graver test set vector by such a permutation. This leads us to the following definition.

Definition 3.1.2 *(Building blocks)*

Let $z = (u, v_1, \ldots, v_N) \in \ker_{\mathbb{Z}^{d_N}}(A_N)$ and call the vectors $u, v_1, \ldots, v_N$ the building blocks of z. Denote by $\mathcal{G}_N$ the Graver test set associated with A_N and collect into $\mathcal{H}_N$ all those vectors arising as building blocks of some $z \in \mathcal{G}_N$. By $\mathcal{H}_\infty$ denote the set $\bigcup_{N=1}^{\infty} \mathcal{H}_N$.

The set $\mathcal{H}_\infty$ contains both m-dimensional vectors u associated with the first-stage in (4) and n-dimensional vectors v related to the second-stage in (4). For convenience, we will arrange the vectors in $\mathcal{H}_\infty$ into pairs (u, V_u). For fixed $u \in \mathcal{H}_\infty$, all those vectors $v \in \mathcal{H}_\infty$ are collected into V_u for which $(u, v) \in \ker_{\mathbb{Z}^{d_1}}(A_1)$. In what follows, we will employ this arrangement into pairs to arbitrary sets of m- and n-dimensional building blocks, not necessarily belonging to $\mathcal{H}_\infty$.

The set $\mathcal{H}_\infty$ is of particular interest, since, by definition, it contains all building blocks of test set vectors of (4) for an arbitrary number N of scenarios. Next, we will address finiteness of $\mathcal{H}_\infty$, computation of $\mathcal{H}_\infty$, and reconstruction of improving vectors using $\mathcal{H}_\infty$.

3.2 Finiteness of $\mathcal{H}_\infty$

As two main results of this thesis we will prove finiteness of $\mathcal{H}_\infty$ in the integer and continuous situations.

3.2.1 IP case

To prove our claim for the integer case we have to use some further algebraic machinery. For an introduction into basic notions we refer to [16]. In order to prove the subsequent Theorem 3.2.11 we will use the following recent theorem on monomial ideals.

Theorem 3.2.1 *(Maclagan, [32, 33])*

Let $\mathcal{I}$ be an infinite collection of monomial ideals in a polynomial ring. Then there are two ideals $I, J \in \mathcal{I}$ with $I \subseteq J$.

Assuming that each monomial ideal is generated by only one monomial, it can be seen that Maclagan's theorem includes as a special case the Gordan-Dickson Lemma, Lemma 1.3.3, which was used in Section 1.3 to show finiteness of the Completion Algorithm 1.3.1 to compute IP Graver test sets.

We now add two useful corollaries (cf. [32]) and, for convenience, include their short proofs.

Corollary 3.2.2 *Let $\mathcal{I}$ be an infinite collection of pair-wise different monomial ideals in a polynomial ring. Then $\mathcal{I}$ contains only finitely many inclusion maximal monomial ideals.*

Proof. Let M denote the set of all monomial ideals in $\mathcal{I}$ that are maximal with respect to inclusion. If M were not finite, then it contained two different ideals $I, J \in \mathcal{I}$ with $I \subseteq J$. Since $I \neq J$ this last relation is strict and thus, I is not inclusion maximal in $\mathcal{I}$. □

Corollary 3.2.3 *Let $(I_1, I_2, \ldots)$ be a sequence of monomial ideals in a polynomial ring with $I_j \not\subseteq I_i$ whenever $i < j$. Then this sequence is finite.*

Proof. Let M denote the set of all inclusion maximal monomial ideals in the set $\{I_j : j = 1, \ldots\}$. Then M is finite by Corollary 3.2.2. Thus, there is some $k \in \mathbb{Z}_+$ such that $M \subseteq \{I_1, \ldots, I_k\}$. Therefore, for all $j > k$ there is some $i \in \{1, \ldots, k\}$ satisfying $I_j \subseteq I_i$, in contradiction to the assumption that no such pair i, j exists. □

We will use this last corollary to obtain a similar result on sequences of pairs (u, V_u) which will be employed to prove termination of the subsequent algorithm to compute $\mathcal{H}_\infty$. To this end, we define the following notions.

Definition 3.2.4 *We say that $(u', V_{u'})$ reduces (u, V_u), or $(u', V_{u'}) \sqsubseteq (u, V_u)$ for short, if the following conditions are satisfied:*

- $u' \sqsubseteq u$,
- *for every $v \in V_u$ there exists a $v' \in V_{u'}$ with $v' \sqsubseteq v$,*
- $u' \neq 0$ *or there exist vectors $v \in V_u$ and $v' \in V_{u'}$ with $0 \neq v' \sqsubseteq v$.*

The last condition above is necessary to avoid the situation that u' and all occurring v' are zero, which forms a trivial zero reduction. In other words, for every N and for every vector $z = (u, v_1, \ldots, v_N)$ constructable from the building blocks in (u, V_u) there is a non-zero (!) vector $z' = (u', v'_1, \ldots, v'_N)$ constructable from the building blocks in $(u', V_{u'})$ such that $z' \sqsubseteq z$. Therefore, $(u', V_{u'}) \sqsubseteq (u, V_u)$ implies that (at least not all) the building blocks in (u, V_u) are not needed for the description of Graver basis elements for any N.

Definition 3.2.5 *We associate with every pair (u, V_u), $u \neq 0$, the monomial ideal $I(u, V_u) \in Q[x_1, \ldots, x_{2m+2n}]$ generated by all the monomials $x^{(u^+, u^-, v^+, v^-)}$ with $v \in V_u$, whereas we associate with $(0, V_0)$ the monomial ideal $I(0, V_0) \in Q[x_1, \ldots, x_{2n}]$ generated by all the monomials $x^{(v^+, v^-)}$ with $v \in V_0 \setminus \{0\}$.*

Lemma 3.2.6 *Let $(u, V_u), (u', V_{u'})$ with $u, u' \neq 0$ be given. Then $I(u, V_u) \subseteq I(u', V'_u)$ implies $(u', V_{u'}) \sqsubseteq (u, V_u)$. Therefore, $(u', V_{u'}) \not\sqsubseteq (u, V_u)$ implies $I(u, V_u) \not\subseteq I(u', V'_u)$.*

Proof. Since $I(u, V_u)$ and $I(u', V_{u'})$ are monomial ideals, we have $I(u, V_u) \subseteq I(u', V_{u'})$ if and only if every generator $x^{(u^+, u^-, v^+, v^-)}$ of $I(u, V_u)$ is divisible by some generator $x^{((u')^+, (u')^-, (v')^+, (v')^-)}$ of $I(u', V_{u'})$ (cf. [16]). The latter implies that $u' \sqsubseteq u$ and that for every $v \in V_u$ there exists vectors $v' \in V_{u'}$ with $v' \sqsubseteq v$. In other words, we have $(u', V_{u'}) \sqsubseteq (u, V_u)$, and the proof is complete. □

Lemma 3.2.7 *$(0, V'_0) \not\sqsubseteq (0, V_0)$ implies $I(0, V_0) \not\subseteq I(0, V'_0)$.*

Proof. $(0, V'_0) \not\sqsubseteq (0, V_0)$ implies that there is some $v \in V_0$ such that there is no $v' \in V'_0 \setminus \{0\}$ with $v' \sqsubseteq v$. But this means that $x^{(v^+, v^-)} \notin I(0, V'_0)$ and, therefore, we have $I(0, V_0) \not\subseteq I(0, V'_0)$. □

Now we are in the position to prove the above mentioned lemma on sequences of pairs.

Lemma 3.2.8 *Let $((u_1, V_{u_1}), (u_2, V_{u_2}), \ldots)$ be a sequence of pairs such that $u_i \neq 0$ for all $i = 1, 2, \ldots$, and such that $(u_i, V_{u_i}) \not\sqsubseteq (u_j, V_{u_j})$ whenever $i < j$. Then this sequence is finite.*

Proof. Consider the sequence $(I(u_1, V_{u_1}), I(u_2, V_{u_2}), \ldots)$ of monomial ideals. By the above Lemma 3.2.6, it fulfills $I(u_j, V_{u_j}) \not\subseteq I(u_i, V_{u_i})$ whenever $i < j$. Thus, by Corollary 3.2.3, this sequence is finite implying that the sequence $((u_1, V_{u_1}), (u_2, V_{u_2}), \ldots)$ is finite, as well. □

Lemma 3.2.9 *Let $((0, V_1), (0, V_2), \ldots)$ be a sequence of pairs with $(0, V_i) \not\sqsubseteq (0, V_j)$ whenever $i < j$. Then this sequence is finite.*

Proof. Consider the sequence $(I(0, V_1), I(0, V_2), \ldots)$ of monomial ideals. By Lemma 3.2.7, it fulfills $I(0, V_j) \not\subseteq I(0, V_i)$ whenever $i < j$. Thus, by Corollary 3.2.3, this sequence is finite implying that the sequence $((0, V_1), (0, V_2), \ldots)$ is finite, as well. □

As a consequence of the last two lemmas we obtain the following.

Lemma 3.2.10 *Let $((u_1, V_{u_1}), (u_2, V_{u_2}), \ldots)$ be a sequence of pairs in the IP situation such that $(u_i, V_{u_i}) \not\sqsubseteq (u_j, V_{u_j})$ whenever $i < j$. Then this sequence is finite.*

Proof. Suppose the sequence $((u_1, V_{u_1}), (u_2, V_{u_2}), \ldots)$ is not finite. Consider the two subsequences where all u_i are non-zero and where all u_i are zero, respectively. At least one of these subsequences is not finite and satisfies $(u_i, V_{u_i}) \not\sqsubseteq (u_j, V_{u_j})$ whenever $i < j$. But this contradicts one of the two preceding Lemmas 3.2.8 and 3.2.9. □

Lemma 3.2.10 will imply finiteness of the subsequent algorithm to compute $\mathcal{H}_\infty$ in the IP case. Below we will prove correctness of this algorithm. Then, termination and correctness together imply finiteness of $\mathcal{H}_\infty$ in the IP situation. Thus, we have the following.

Theorem 3.2.11 *Given integer matrices A, T, and W of appropriate dimensions, and let $\mathcal{H}_\infty$ be defined for the IP case as above. Then $\mathcal{H}_\infty$ is a finite set.*

3.2.2 LP case

For the LP case we have to define the relation $\sqsubseteq$ differently.

Definition 3.2.12 *We say that the pair (u, V_u) can be reduced by the pair $(u', V_{u'})$, or $(u', V_{u'}) \sqsubseteq (u, V_u)$ for short, if the following conditions are satisfied:*

1. $\operatorname{supp}(u') \subseteq \operatorname{supp}(u)$,
2. *for all $v_i \in V_u$ there exists $v_i' \in V_{u'}$ such that $\operatorname{supp}(v_i') \subseteq \operatorname{supp}(v_i)$, and*
3. *$u \neq 0$ or there exist $v_i \in V_u$ and $v_i' \in V_{u'}$ such that $\emptyset \neq \operatorname{supp}(v_i') \subseteq \operatorname{supp}(v_i)$.*

The third condition is necessary to avoid the situation that u' and all occurring v_i' are zero, which forms a trivial zero reduction.

Lemma 3.2.13 *Let $((u_1, V_{u_1}), (u_2, V_{u_2}), \ldots)$ be a sequence of pairs in the LP situation such that $(u_i, V_{u_i}) \not\sqsubseteq (u_j, V_{u_j})$ whenever $i < j$. Then this sequence is finite.*

Proof. Define V_u and $V_{u'}$ to be equivalent if for each $v \in V_u$ there is some $v' \in V_{u'}$ with $\operatorname{supp}(v) = \operatorname{supp}(v')$ and vice-versa. This defines an equivalence relation with 2^{2^n} equivalence classes.

Define (u, V_u) and $(u', V_{u'})$ to be equivalent if $\operatorname{supp}(u) = \operatorname{supp}(u')$ and V_u is equivalent to $V_{u'}$. This defines an equivalence relation with $2^m 2^{2^n}$ equivalence classes.

Moreover, if (u, V_u) and $(u', V_{u'})$ are equivalent it holds that $(u, V_u) \sqsubseteq (u', V_{u'})$ and $(u', V_{u'}) \sqsubseteq (u, V_u)$. Therefore, there are no two equivalent pairs in the sequence $((u_1, V_{u_1}), (u_2, V_{u_2}), \ldots)$. Thus, this sequence is finite. □

This lemma will imply finiteness of the subsequent algorithm to compute $\mathcal{H}_\infty$ in the LP case. Below we will prove correctness of this algorithm. Then, termination and correctness together imply finiteness of $\mathcal{H}_\infty$ in the LP situation. Thus, we have the following.

Theorem 3.2.14 *Given integer matrices A, T, and W of appropriate dimensions, and let $\mathcal{H}_\infty$ be defined for the LP case as above. Then $\mathcal{H}_\infty$ is a finite set.*

3.3 Computation of $\mathcal{H}_\infty$

If $\mathcal{H}_\infty$ were finite, we could find this set by computing the Graver test set $\mathcal{G}_N$ for sufficiently large N and by decomposing its elements into their building blocks. Even when disregarding that we do not know in advance how big N then has to be taken, this approach is not very practical, due to the size of $\mathcal{G}_N$. The idea now is to retain the pattern of Graver test set computations from Section 1.3, but to work with pairs (u, V_u) instead and to define the main ingredients, input set, normalForm, and S-vectors, to the completion procedure appropriately. In what follows, the objects f, g, and s all are pairs of the form (u, V_u).

Algorithm 3.3.1 *(Algorithm to compute $\mathcal{H}_\infty$)*

Input: a symmetric generating set F of $\ker_{\mathbb{Z}^{d_1}}(A_1)$ over $\mathbb{Z}$ or of $\ker_{\mathbb{R}^{d_1}}(A_1)$ over $\mathbb{R}$ in (u, V_u)-notation to be specified below

Output: a set $G \supseteq \mathcal{H}_\infty$

$G := F$

$C := \bigcup_{f,g \in G} \{f \oplus g\}$ *(forming* S-vectors*)*

while $C \neq \emptyset$ *do*

$s :=$ *an element in* C

$C := C \setminus \{s\}$

$f := \text{normalForm}(s, G)$

if $f \neq (0, \{0\})$ *then*

$$C := C \cup \bigcup_{g \in G \cup \{f\}} \{f \oplus g\} \qquad \textit{(adding S-vectors)}$$
$$G := G \cup \{f\}$$

return G.

Behind the function normalForm(s, G) there is the following algorithm.

Algorithm 3.3.2 *(Normal form algorithm)*

Input: a pair s, a set G of pairs

Output: a normal form of s with respect to G

while there is some $g \in G$ such that $g \sqsubseteq s$ do

$s := s \ominus g$

return s

It remains to define an appropriate input set, the sum $\oplus$, and the difference $\ominus$ of two pairs (u, V_u) and $(u', V_{u'})$. To get good candidates we may think of a computation of $\mathcal{G}_N$, N sufficiently large, where every vector is decomposed into its building blocks.

Lemma 3.3.3 *Let F be a symmetric generating set for $\ker_{\mathbb{Z}^{d_1}}(A_1)$ over $\mathbb{Z}$ which contains a generating set for $\{(0, v) : Wv = 0\} \subseteq \ker_{\mathbb{Z}^{d_1}}(A_1)$ consisting only of vectors with zero as first-stage component. Moreover, let F_N be the set of all those vectors in $\ker_{\mathbb{Z}^{d_N}}(A_N)$ whose building blocks are also building blocks of vectors in $F \cup \{0\}$. Then, for any N, the vectors F_N generate $\ker_{\mathbb{Z}^{d_N}}(A_N)$ over $\mathbb{Z}$.*

The same result holds for the continuous kernel $\ker_{\mathbb{R}^{d_N}}(A_N)$.

Proof. Let $z = (u, v_1, \ldots, v_N) \in \ker_{\mathbb{Z}^{d_N}}(A_N)$. We have to show that z can be written as a linear integer combination of elements from F_N.

From $(u, v_1) \in \ker_{\mathbb{Z}^{d_1}}(A_1)$ we conclude that there is a finite linear integer combination $(u, v_1) = \sum \alpha_i (u_i, v_{i,1})$, $(u_i, v_{i,1}) \in F$. Hence $(u, v_1, \ldots, v_1) = \sum_i \alpha_i (u_i, v_{i,1}, \ldots, v_{i,1})$ with $(u_i, v_{i,1}, \ldots, v_{i,1}) \in F_N$. Moreover, for $j = 2, \ldots, N$, we have $W(v_j - v_1) = 0$, that is, $(0, v_j - v_1) \in \ker_{\mathbb{Z}^{d_1}}(A_1)$, since $Tu + Wv_1 = Tu + Wv_j$. Since F contains an integer generating set for $\{(0, v) : Wv = 0\} \subseteq \ker_{\mathbb{Z}^{d_1}}(A_1)$ consisting only of vectors with zero as first-stage component, we can construct linear integer combinations $(0, v_j - v_1) = \sum_i \beta_{i,j} (0, v_{i,j})$ with $(0, v_{i,j}) \in F$ and $j = 2, \ldots, N$.

Thus, we have $(0,0,\ldots,0,v_j - v_1,0,\ldots,0) = \sum_i \beta_{i,j}(0,0,\ldots,0,v_{i,j},0,\ldots,0)$ with $(0,0,\ldots,0,v_{i,j},0,\ldots,0) \in F_N$. But now we get

$$\begin{aligned} z &= (u, v_1, \ldots, v_1) + \sum_j (0,0,\ldots,0,v_j - v_1,0,\ldots,0) \\ &= \sum_i \alpha_i (u_i, v_{i,1}, \ldots, v_{i,1}) + \sum_{j>1} \sum_i \beta_{i,j}(0,0,\ldots,0,v_{i,j},0,\ldots,0), \end{aligned}$$

as desired.

An analogous argumentation for the continuous case shows the result for $\ker_{\mathbb{R}^{d_N}}(A_N)$. □

Before we define the input set, the sum $\oplus$, and the difference $\ominus$ for the IP and the LP cases let us introduce the following notation.

Definition 3.3.4 *For $\alpha \in \mathbb{R}$ let*

$$(u, V_u) + \alpha(u', V_{u'}) := (u + \alpha u', V_u + \alpha V_{u'}),$$

where

$$V_u + \alpha V_{u'} := \{v + \alpha v' : v \in V_u, v' \in V_{u'}\}.$$

3.3.1 IP case

Lemma 3.3.3 suggests the following input set.

Definition 3.3.5 *Let F be a symmetric generating set for $\ker_{\mathbb{Z}^{d_1}}(A_1)$ over $\mathbb{Z}$ which contains a generating set for $\{(0,v) : Wv = 0\} \subseteq \ker_{\mathbb{Z}^{d_1}}(A_1)$ consisting only of vectors with zero as first-stage components. With this define the building blocks of all vectors in $F \cup \{0\}$ in (u, V_u)-notation to be the input set to Algorithm 3.3.1 for the integer case.*

Definition 3.3.6 *For the computation of $\mathcal{H}_\infty$ for two-stage stochastic integer programs define*

$$(u, V_u) \oplus (u', V_{u'}) := (u, V_u) + (u', V_{u'}) = (u + u', V_u + V_{u'})$$

and

$$(u, V_u) \ominus (u', V_{u'}) := (u - u', \{v - v' : v \in V_u, v' \in V_{u'}, v' \sqsubseteq v\}).$$

Remark. In $(u, V_u) \ominus (u', V_{u'})$ it suffices to collect only one difference $v - v'$ for every $v \in V_u$. The proof of the subsequent proposition also shows that Algorithm 3.3.1 still terminates and works correctly if we defined $(u, V_u) \ominus (u', V_{u'})$ in this way. □

Proposition 3.3.7 *If the input set, the procedure* normalForm*, $f \oplus g$, and $s \ominus g$ are defined as in Definitions 3.2.4, 3.3.5, and 3.3.6, then Algorithm 3.3.1 terminates and returns a set containing $\mathcal{H}_\infty$ for two-stage stochastic integer programs.*

Proof. In the course of the algorithm, a sequence of pairs in $G \setminus F$ is generated that satisfies the conditions of Lemma 3.2.10. Therefore, Algorithm 3.3.1 terminates.

Denote by G the set that is returned by Algorithm 3.3.1. To show that $\mathcal{H}_\infty \subseteq G$, we have to prove that $\mathcal{H}_N \subseteq G$ for all $N \in \mathbb{Z}_+$. Fix N and start a Graver test set computation (see Section 1.3) with

$$\bar{F} := \{(u, v_1, \ldots, v_N) : (u, V_u) \in G, v_i \in V_u, i = 1, \ldots, N\}$$

as input set. $\bar{F}$ generates $\ker_{\mathbb{Z}^{d_N}}(A_N)$ over $\mathbb{Z}$ for all $N \in \mathbb{Z}_+$, since $F_N \subseteq \bar{F}$ by the assumptions on the input set to Algorithm 3.3.1 and by Lemma 3.3.3.

We will now show that all sums $z + z' \in$ S-vectors(z, z') of two elements $z, z' \in \bar{F}$ reduce to 0 with respect to $\bar{F}$. In this case, Algorithm 1.3.1 returns the input set $\bar{F}$ which implies $\mathcal{G}_N \subseteq \bar{F}$. Therefore, $\mathcal{H}_N \subseteq G$ as desired.

Take two arbitrary elements $z = (u, v_1, \ldots, v_N)$ and $z' = (u', v'_1, \ldots, v'_N)$ from $\bar{F}$, and consider the vector $z + z' = (u + u', v_1 + v'_1, \ldots, v_N + v'_N)$.

In the above algorithm, $(u, V_u) \oplus (u', V_{u'})$ was reduced to $(0, \{0\})$ by elements (u_1, V_{u_1}), $\ldots$, $(u_k, V_{u_k}) \in G$. From this sequence we can construct a sequence $z_1, \ldots, z_k$ of vectors in $\bar{F}$ which reduce $z + z'$ to zero as follows.

$(u_1, V_{u_1}) \sqsubseteq (u, V_u) \oplus (u', V_{u'})$ implies that $u_1 \sqsubseteq u + u'$ and that there exist vectors $v_{1,1}, \ldots, v_{1,N} \in V_{u_1}$ satisfying $v_{1,i} \sqsubseteq v_i + v'_i$ for $i = 1, \ldots, N$. Therefore, we conclude $z_1 := (u_1, v_{1,1}, \ldots, v_{1,N}) \sqsubseteq z + z'$ and thus, $z + z'$ can be reduced to $z + z' - z_1$. Moreover, $z_1 \in \bar{F}$ and all the building blocks of $z + z' - z_1$ are contained in the pair $((u, V_u) \oplus (u', V_{u'})) \ominus (u_1, V_{u_1})$.

But $((u, V_u) \oplus (u', V_{u'})) \ominus (u_1, V_{u_1})$ was further reduced by $(u_2, V_{u_2}), \ldots, (u_k, V_{u_k}) \in G$. Therefore, we can construct from (u_2, V_{u_2}) a vector $z_2 \in \bar{F}$ with $z_2 \sqsubseteq z + z' - z_1$. Thus, $z + z' - z_1$ can be further reduced to $z + z' - z_1 - z_2$.

Repeating this construction, we arrive in the k^{th} step at $z + z' - z_1 - \ldots - z_{k-1}$ whose building blocks all lie in $((u, V_u) \oplus (u', V_{u'})) \ominus (u_1, V_{u_1}) \ominus \ldots \ominus (u_{k-1}, V_{u_{k-1}})$. The latter

can be reduced to $(0, \{0\})$ by the pair $(u_k, V_{u_k}) \in G$. Therefore, there exists a vector $z_k \in \bar{F}$ such that $z_k \sqsubseteq z + z' - z_1 - \ldots - z_{k-1}$ and $0 = z + z' - z_1 - \ldots - z_k$. $\square$

Note that termination and correctness of the above algorithm imply finiteness of $\mathcal{H}_\infty$. Therefore, the above proposition finally proves Theorem 3.2.11.

3.3.2 LP case

Lemma 3.3.3 suggests the following input set.

Definition 3.3.8 *Let F be a generating set for $\ker_{\mathbb{R}^{d_1}}(A_1)$ over $\mathbb{R}$ which contains a generating set for $\{(0, v) : Wv = 0\} \subseteq \ker_{\mathbb{R}^{d_1}}(A_1)$ consisting only of vectors with zero as first-stage components. With this define the building blocks of all vectors in $F \cup \{0\}$ in (u, V_u)-notation to be the input set to Algorithm 3.3.1 for the continuous case.*

Definition 3.3.9 *For the computation of $\mathcal{H}_\infty$ for two-stage stochastic linear programs define $\oplus$ and $\ominus$ as follows. For given (u, V_u), $(u', V_{u'})$, and all α such that*

1. *$u + \alpha u'$ contains a zero entry at some component i at which neither u nor u' have a zero entry, that is, $(u + \alpha u')^{(i)} = 0$ but $u^{(i)}(u')^{(i)} \neq 0$, or*
2. *for some vectors $v \in V_u, v' \in V_{u'}$ the vector $v + \alpha v'$ contains a zero entry at some component i at which neither v nor v' have a zero entry, that is, $(v + \alpha v')^{(i)} = 0$ but $v^{(i)}(v')^{(i)} \neq 0$,*

let

$$(u, V_u) \oplus (u', V_{u'}) := \{(u, V_u) + \alpha(u', V_{u'}) : \alpha \in \mathbb{R} \text{ satisfying the above conditions } 1, 2\}.$$

In case of reduction, that is, if $(u', V_{u'}) \sqsubseteq (u, V_u)$ (see Definition 3.2.12), (u, V_u) reduces to

$$(u, V_u) \ominus \beta(u', V_{u'}) := (u - \beta u', \{v_i - \beta v_i' : v_i \in V_u, v_i' \in V_{u'} \text{ as in Definition 3.2.12}\}),$$

where β takes one of the finitely many values such that $\operatorname{supp}(u - \beta u') \subsetneq \operatorname{supp}(u)$ or such that $\operatorname{supp}(v_i - \beta v_i') \subsetneq \operatorname{supp}(v_i)$ for at least one i.

Since $(u - \beta u', \{v_i - \beta v_i' : v_i \in V_u, v_i' \in V_{u'}\})$ contains at least one more zero entry in u or in some element of V_u, normalForm(s, G) always terminates.

Proposition 3.3.10 *If the input set, the procedure* normalForm*, $f \oplus g$, and $s \ominus g$ are defined as above then Algorithm 3.3.1 terminates and returns a set containing $\mathcal{H}_\infty$ for two-stage stochastic linear programs.*

Proof. In the course of the algorithm, a sequence of pairs in $G \setminus F$ is generated that satisfies the conditions of Lemma 3.2.13. This means that $G \setminus F$ contains at most $2^m 2^{2^n}$ different pairs (u, V_u). Therefore, Algorithm 3.3.1 terminates.

Let G denote the set that is returned by Algorithm 3.3.1. To show correctness, that is $\mathcal{H}_\infty \subseteq G$, we have to prove $\mathcal{H}_N \subseteq G$ for all $N \in \mathbb{Z}_+$. Fix N and start a Graver test set computation with $\bar{F} := \{(u, v_1, \ldots, v_N) : (u, V_u) \in G, v_i \in V_u, i = 1, \ldots, N\}$ as input set. Note that $\bar{F}$ generates $\ker_{\mathbb{R}^{d_N}}(A_N)$ over $\mathbb{R}$ for all $N \in \mathbb{Z}_+$, since $F_N \subseteq \bar{F}$ by the assumptions on the input set to Algorithm 3.3.1 and by Lemma 3.3.3. If all S-vectors $f + \alpha g \in$ S-vectors(f, g) of two elements $f, g \in \bar{F}$ can be written as a linear combination of elements of $\bar{F}$ whose support is contained in $\operatorname{supp}(f + \alpha g)$, then the Graver basis $\mathcal{G}_N$ is contained in $\bar{F}$ (see Remark 1.3.5). Since N was chosen arbitrarily, $\mathcal{G}_N \subseteq \bar{F}$ implies $\mathcal{H}_N \subseteq G$ for all $N \in \mathbb{Z}_+$ and we are done.

Take two arbitrary elements $z = (u, v_1, \ldots, v_N)$ and $z' = (u', v'_1, \ldots, v'_N)$ from $\bar{F}$ and choose $\alpha \in \mathbb{R}$ such that $z + \alpha z' \in$ S-vectors(z, z'). In the above algorithm, $(u, V_u) + \alpha(u', V_{u'}) \in (u, V_u) \oplus (u', V_{u'})$ was successively reduced to $(0, \{0\})$ by (u_1, V_{u_1}), $\ldots$, $(u_k, V_{u_k}) \in G$. From this sequence we will now construct a sequence $z_1, \ldots, z_k$ of vectors in $\bar{F}$ such that $z + \alpha z' = \sum \alpha_i z_i$ and $\operatorname{supp}(z_i) \subseteq \operatorname{supp}(z + \alpha z')$ as desired.

In the very first reduction step, the pair $(u, V_u) + \alpha(u', V_{u'})$ was reduced to the pair $((u, V_u) + \alpha(u', V_{u'})) \ominus \alpha_1(u_1, V_{u_1})$. But $(u_1, V_{u_1}) \sqsubseteq (u, V_u) + \alpha(u', V_{u'})$ implies that $\operatorname{supp}(u_1) \subseteq \operatorname{supp}(u + \alpha u')$ and that there exist vectors $v_{1,1}, \ldots, v_{1,N} \in V_{u_1}$ with $\operatorname{supp}(v_{1,i}) \subseteq \operatorname{supp}(v_i + \alpha v'_i)$, $i = 1, \ldots, N$. Thus, $z_1 := (u_1, v_{1,1}, \ldots, v_{1,N}) \in \bar{F}$, $\operatorname{supp}(z_1) \subseteq \operatorname{supp}(z + \alpha z')$, and all the building blocks of $z + \alpha z' - \alpha_1 z_1$ lie in the pair $((u, V_u) + \alpha(u', V_{u'})) \ominus \alpha_1(u_1, V_{u_1})$.

But $((u, V_u) + \alpha(u', V_{u'})) \ominus (u_1, V_{u_1})$ was further reduced by $(u_2, V_{u_2}) \in G$ to the pair $((u, V_u) + \alpha(u', V_{u'})) \ominus \alpha_1(u_1, V_{u_1}) \ominus \alpha_2(u_2, V_{u_2})$. Thus, we can construct from (u_2, V_{u_2}) a vector $z_2 \in \bar{F}$ such that $\operatorname{supp}(z_2) \subseteq \operatorname{supp}(z + \alpha z' - \alpha_1 z_1)$, and all building blocks of $z + \alpha z' - \alpha_1 z_1 - \alpha_2 z_2$ lie in $((u, V_u) + \alpha(u', V_{u'})) \ominus \alpha_1(u_1, V_{u_1}) \ominus \alpha_2(u_2, V_{u_2})$.

Repeating this construction, we arrive in the k^{th} step at $z + \alpha z' - \sum_{i=1}^{k-1} \alpha_i z_i$ whose building blocks all lie in $((u, V_u) + \alpha(u', V_{u'})) \ominus \alpha_1(u_1, V_{u_1}) \ominus \ldots \ominus \alpha_{k-1}(u_{k-1}, V_{u_{k-1}})$. The latter can be reduced to $(0, \{0\})$ by the pair $(u_k, V_{u_k}) \in G$. Therefore, there has to exist a vector $z_k \in \bar{F}$ such that $\operatorname{supp}(z_k) \subseteq \operatorname{supp}(z + \alpha z' - \sum_{i=1}^{k-1} \alpha_i z_i)$ and $0 = z + \alpha z' - \sum_{i=1}^{k} \alpha_i z_i$. Thus, $z + \alpha z' = \sum_{i=1}^{k} \alpha_i z_i$ with $\operatorname{supp}(z_i) \subseteq \operatorname{supp}(z + \alpha z')$ for $i = 1, \ldots, k$, which completes the proof. $\square$

Note that termination and correctness of the above algorithm imply finiteness of $\mathcal{H}_\infty$. Therefore, the above proposition finally proves Theorem 3.2.14.

3.4 Solving the Optimization Problem with the Help of $\mathcal{H}_\infty$

As we have seen in Section 2.6, universal test sets can help to find an initial feasible solution to our optimization problem and to augment it to optimality. For our optimization problem $\min\{c^\intercal z : A_N z = b, z \in \mathbb{Z}_+^{d_N}\}$, however, the universal test set is not given explicitly. Only its set of building blocks $\mathcal{H}_\infty$ is available. As mentioned in Remark 2.6.4, we may concentrate on finding improving directions. In the following, we will see how these directions can be reconstructed from $\mathcal{H}_\infty$.

3.4.1 IP case

Suppose we are given $\mathcal{H}_\infty$, a cost function c and a feasible solution $z_0 = (u, v_1, \ldots, v_N)$.

Lemma 3.4.1 *Suppose there exists no pair $(u', V_{u'}) \in \mathcal{H}_\infty$ with the properties*

1. *$u' \leq u$,*
2. *for all $i = 1, \ldots, N$, there exists $\bar{v}_i \in V_{u'} : \bar{v}_i \leq v_i$,*
3. *$c^\intercal z' > 0$, where $z' = (u', v'_1, \ldots, v'_N)$ and $v'_i \in \operatorname{argmax}\{c_i^\intercal \bar{v}_i : \bar{v}_i \leq v_i, \bar{v}_i \in V_{u'}\}$ for $i = 1, \ldots, N$.*

Then $z_0 = (u, v_1, \ldots, v_N)$ is optimal for $\min\{c^\intercal z : A_N z = b, z \in \mathbb{Z}_+^{d_N}\}$.

If there exists such a pair $(u', V_{u'}) \in \mathcal{H}_\infty$ then $z_0 - z'$ is a feasible solution and it holds $c^\intercal(z_0 - z') < c^\intercal z_0$.

Proof. Suppose that z_0 is not optimal. Then there has to exist some improving vector $z'' = (u'', v''_1, \ldots, v''_N) \in \mathcal{G}_N$ such that $z_0 - z''$ is feasible and $c^\intercal(z_0 - z'') < c^\intercal z_0$. Feasibility of $z_0 - z''$ implies $z_0 - z'' \geq 0$, hence $z'' \leq z_0$. Therefore, $u'' \leq u$ and $v''_i \leq v_i$, $i = 1, \ldots, N$, the latter implying that for any $i = 1, \ldots, N$ there exists a $\bar{v}_i \in V_{u''}$ such that $\bar{v}_i \leq v_i$.

Let $z' := (u'', v'_1, \ldots, v'_N)$ where $v'_i \in \operatorname{argmax}\{c_i^\intercal \bar{v}_i : \bar{v}_i \leq v_i, \bar{v}_i \in V_{u''}\}$. But now $c^\intercal(z_0 - z'') < c^\intercal z_0$ implies that $c^\intercal z'' > 0$. Moreover, $c^\intercal z' \geq c^\intercal z'' > 0$. In conclusion,

the pair $(u', V_{u'})$ with $u' := u''$ fulfills conditions $1. - 3.$ proving the first claim of the lemma.

With $z' = (u', v'_1, \ldots, v'_N)$ according to 3. we obtain $c^\intercal(z_0 - z') < c^\intercal z_0$. Moreover $v'_i \leq v_i$, $i = 1, \ldots, N$, and $u' \leq u$ together imply $z' \leq z_0$, and $z_0 - z' \geq 0$. Finally, $(u', v'_1, \ldots, v'_N) \in \ker_{\mathbb{Z}^{d_N}}(A_N)$, and therefore $A_N(z_0 - z') = A_N z_0 + 0 = b$ which completes the proof. □

Theorem 3.4.2 *If the problem* $\min\{c^\intercal z : A_N z = b, z \in Z_+^{d_N}\}$ *is solvable, an optimal solution can be computed in finitely many steps by application of the Algorithms 0.0.2 and 2.6.1 together with the above reconstruction procedures for finding improving vectors.*

Moreover, the reconstruction procedure in the above Lemma 3.4.2 is linear with respect to the number N of scenarios. In accordance with that, we observed in our preliminary test runs (cf. Section 3.6) that the method is fairly insensitive with respect to growing of the number N of scenarios. Of course, this becomes effective only after $\mathcal{H}_\infty$ has been computed.

3.4.2 LP case

Suppose we are given $\mathcal{H}_\infty$, a cost function c and a feasible solution $z_0 = (u, v_1, \ldots, v_N)$.

Lemma 3.4.3 *Suppose there exists no pair* $(u', V_{u'}) \in \mathcal{H}_\infty$ *with the properties*

1. $\operatorname{supp}(u'^+) \subseteq \operatorname{supp}(u)$,
2. *for all* $i = 1, \ldots, N$, *there exists* $\bar{v}_i \in V_{u'} : \operatorname{supp}(\bar{v}_i^+) \subseteq \operatorname{supp}(v_i)$,
3. $c^\intercal z' > 0$, *where* $z' = (u', v'_1, \ldots, v'_N)$ *and*

$$v'_i \in \operatorname{argmax}\{c_i^\intercal \bar{v}_i : \operatorname{supp}(\bar{v}_i^+) \subseteq \operatorname{supp}(v_i), \bar{v}_i \in V_{u'}\} \text{ for } i = 1, \ldots, N.$$

Then $z_0 = (u, v_1, \ldots, v_N)$ *is optimal for* $\min\{c^\intercal z : A_N z = b, z \in \mathbb{R}_+^{d_N}\}$.

If there exists such a pair $(u', V_{u'}) \in \mathcal{H}_\infty$ *then there is some positive scalar* α *such that* $z_0 - \alpha z'$ *is feasible and it holds* $c^\intercal(z_0 - \alpha z') < c^\intercal z_0$.

Proof. Suppose that z_0 is not optimal. Then there has to exist some improving vector $z'' = (u'', v''_1, \ldots, v''_N) \in \mathcal{G}_N$ and a positive scalar β such that $z_0 - \beta z''$ is

feasible and $c^\intercal(z_0 - \beta z'') < c^\intercal z_0$. Feasibility of $z_0 - \beta z''$ implies $z_0 - \beta z'' \geq 0$, and therefore $\text{supp}((z'')^+) \subseteq \text{supp}(z_0)$. Therefore, we have $\text{supp}((u'')^+) \subseteq \text{supp}(u)$ and $\text{supp}((v_i'')^+) \subseteq \text{supp}(v_i)$, $i = 1, \ldots, N$, the latter implying that for any $i = 1, \ldots, N$, there exists $\bar{v}_i \in V_{u''}$ such that $\text{supp}(\bar{v}_i^+) \subseteq \text{supp}(v_i)$. Let $z' := (u'', v_1', \ldots, v_N')$ where $v_i' \in \text{argmax}\{c_i^\intercal \bar{v}_i : \text{supp}(\bar{v}_i^+) \subseteq \text{supp}(v_i), \bar{v}_i \in V_{u''}\}$.

Now $c^\intercal(z_0 - \beta z'') < c^\intercal z_0$ and $\beta > 0$ imply that $c^\intercal z'' > 0$. Moreover, $c^\intercal z' \geq c^\intercal z'' > 0$. In conclusion, the pair $(u', V_{u'})$ with $u' := u''$ fulfills conditions $1. - 3.$ proving the first claim of the lemma.

With $z' = (u', v_1', \ldots, v_N')$ according to condition 3. above, $c^\intercal z' > 0$ implies that $c^\intercal(z_0 - \alpha z') < c^\intercal z_0$ for any scalar $\alpha > 0$. Moreover, $\text{supp}((v_i')^+) \subseteq \text{supp}(v_i)$, for $i = 1, \ldots, N$, and $\text{supp}((u')^+) \subseteq \text{supp}(u)$ together imply that we can indeed choose a scalar $\alpha > 0$ with $z_0 - \alpha z' \geq 0$. Finally, $(u', v_1', \ldots, v_N') \in \ker_{\mathbb{R}^{d_N}}(A_N)$ and therefore $A_N(z_0 - \alpha z') = A_N z_0 - \alpha \cdot 0 = b$, which completes the proof. □

Theorem 3.4.4 *If the problem* $\min\{c^\intercal z : A_N z = b, z \in \mathbb{R}_+^{d_N}\}$ *is solvable, an optimal solution can be computed in finitely many steps by application of Algorithm 0.0.2 together with the Augmentation Strategy 2.5.1 and the above reconstruction procedure for finding improving vectors.*

Note that the Augmentation Strategy 2.5.1 can also be applied at the building block level, since in phase 1 we have to look for an improving vector v to our feasible solution z_0 for which $c^\intercal v > 0$ and $\text{supp}(v^+) \subseteq \text{supp}(z_0)$ holds.

Moreover, the reconstruction procedure in the above Lemma 3.4.4 is linear with respect to the number N of scenarios.

3.5 Simple Recourse

For two-stage stochastic programs with simple recourse the matrix W takes the particular simple form $W = (D|-D)$, where D is an invertible square matrix of appropriate dimension. Simple recourse instances arise both in two-stage linear [3, 4, 26, 38] and two-stage integer linear programs [27]. In the following we show that for stochastic programs with simple recourse we have $\mathcal{H}_\infty = \mathcal{H}_1$. Thus, it suffices to compute the Graver basis of A_1 in order to reconstruct an improving vector for the full simple recourse problem. To show this, we have to state another property of Graver bases. A similar lemma for IP Graver test sets of knapsack problems can be found in [15].

3.5.1 IP case

Lemma 3.5.1 *Given a matrix* $B = \begin{pmatrix} A & a & -a \end{pmatrix}$*, with two columns which differ only in their respective signs. Then the IP Graver basis of* B *can be constructed from the IP Graver basis of* $B' = \begin{pmatrix} A & a \end{pmatrix}$ *in the following way:*

$$\mathcal{G}_{IP}(B) = \{(u, v, w) : vw \leq 0, (u, v - w) \in \mathcal{G}_{IP}(B')\} \cup \{\pm(0, 1, 1)\}.$$

Proof. Let $(u, v, w) \in \mathcal{G}_{\mathrm{IP}}(B)$. Since $\pm(0, 1, 1)$ are the only minimal elements in $\ker_{\mathbb{Z}^{n+2}}(B) \setminus \{0\}$ with $u = 0$, we may assume that $u \neq 0$. Moreover, it holds $vw \leq 0$, since otherwise either $(u, v + 1, w + 1) \sqsubseteq (u, v, w)$ or $(u, v - 1, w - 1) \sqsubseteq (u, v, w)$ contradicts the $\sqsubseteq$-minimality property of (u, v, w). Thus, without loss of generality, we assume in the following that $v \geq 0$ and $w \leq 0$. Next we show $(u, v - w) \in \mathcal{G}_{\mathrm{IP}}(B')$.

Suppose that $(u, v - w) \notin \mathcal{G}_{\mathrm{IP}}(B')$. Then there is some vector $(u', z') \in \mathcal{G}_{\mathrm{IP}}(B')$ with $(u', z') \sqsubseteq (u, v - w)$ and $u' \neq 0$. (If $u' = 0$ then $z' = 0$ contradicting $(u', z') \in \mathcal{G}_{\mathrm{IP}}(B')$ since only non-zero elements are in $\mathcal{G}_{\mathrm{IP}}(B')$.) Of course, $(u', z') \neq (u, v - w)$, since $(u, v - w) \notin \mathcal{G}_{\mathrm{IP}}(B')$. From $v - w \geq 0$ we get $0 \leq z' \leq v - w$. Next we show that $(u, v, w) \notin \mathcal{G}_{\mathrm{IP}}(B)$ which will contradict our initial assumption $(u, v, w) \in \mathcal{G}_{\mathrm{IP}}(B)$.

To prove this, note that $0 \leq \min(z', v) \leq v$ and $0 \geq -z' + \min(z', v) \geq w$. Whereas the first chain of inequalities holds because of $0 \leq z'$, the second can be seen as follows. If $z' \leq v$, we get $0 \geq -z' + z' = 0 \geq w$ by our assumption on w. If, on the contrary, $z' > v$ we get $0 \geq -z' + v \geq -(v - w) + v = w$. But this implies that $(u', \min(z', v), -z' + \min(z', v)) \sqsubseteq (u, v, w)$. Moreover, we know that $u' \neq 0$ and $(u', \min(z', v), -z' + \min(z', v)) \in \ker_{\mathbb{Z}^{n+2}}(B)$. Thus, it remains to prove that $(u', \min(z', v), -z' + \min(z', v)) \neq (u, v, w)$, which implies that (u, v, w) is not $\sqsubseteq$-minimal in $\ker_{\mathbb{Z}^{n+2}}(B) \setminus \{0\}$ and therefore $(u, v, w) \notin \mathcal{G}_{\mathrm{IP}}(B)$.

Suppose that $(u', \min(z', v), -z' + \min(z', v)) = (u, v, w)$ is true. This yields $u = u'$, $\min(z', v) = v$, and $w = -z' + \min(z', v) = -z' + v \geq -(v - w) + v = w$. But this implies that $z' = v - w$ and therefore $(u', z') = (u, v - w)$ in contrast to our assumption $(u', z') \neq (u, v - w)$ above. Thus $(u, v, w) \notin \mathcal{G}_{\mathrm{IP}}(B)$.

Therefore, it remains to prove that every vector (u, v, w) with $u \neq 0$, $v \geq 0$, $w \leq 0$, and $(u, v - w) \in \mathcal{G}_{\mathrm{IP}}(B')$ belongs to $\mathcal{G}_{\mathrm{IP}}(B)$. Clearly, $(u, v, w) \in \ker_{\mathbb{Z}^{n+2}}(B)$. Suppose that there exists a vector $(u', v', w') \in \ker_{\mathbb{Z}^{n+2}}(B)$ with $u' \neq 0$, $(u', v', w') \sqsubseteq (u, v, w)$, and $(u, v, w) \neq (u', v', w')$. But then we conclude $(u', v' - w') \in \ker_{\mathbb{Z}^{n+1}}(B')$ and $(u', v' - w') \sqsubseteq (u, v - w)$. If $(u, v - w) \neq (u', v' - w')$ were true, this would contradict $(u, v - w) \in \mathcal{G}_{\mathrm{IP}}(B')$.

Therefore, suppose that we have $(u, v - w) = (u', v' - w')$, $(u', v', w') \sqsubseteq (u, v, w)$, and $(u, v, w) \neq (u', v', w')$. But then $0 \leq v' \leq v$ and $0 \geq w' \geq w$ together imply that

$0 \leq v' - w' \leq v - w$. Since $u = u'$ and $(u, v, w) \neq (u', v', w')$, at least one of the inequalities $v' \leq v$ and $-w' \leq -w$ holds strictly. Therefore, $0 \leq v' - w' < v - w$, a contradiction to $v - w = v' - w'$. □

Corollary 3.5.2 *Given a matrix* $B = \begin{pmatrix} A & D & -D \end{pmatrix}$, *with a number of paired columns which differ only in their respective signs. Then the IP Graver basis of* B *can be constructed from the IP Graver basis of* $B' = \begin{pmatrix} A & D \end{pmatrix}$ *in the following way:*

$$\mathcal{G}_{IP}(B) = \{(u, v, w) : v^{(i)} w^{(i)} \leq 0, \text{ for all } i, (u, v - w) \in \mathcal{G}_{IP}(B')\} \cup \{\pm(0, 1, 1)\}.$$

Herein, v, w, *and* 1 *are vectors whose dimensions equal the number of columns of* D.

Proof. The proof is analogous to the proof of Lemma 3.5.1, since we can consider the pairs of columns in D and $-D$ independently. □

Lemma 3.5.3 $\mathcal{H}_\infty$ *coincides with* $\mathcal{H}_1$ *for all two-stage stochastic integer programs with simple recourse.*

Proof. The Graver basis of A_N can be reconstructed (in particular each building block of it can be reconstructed) from the Graver basis of the matrix

$$A'_N := \begin{pmatrix} A & 0 & 0 & \cdots & 0 \\ T & D & 0 & \cdots & 0 \\ T & 0 & D & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ T & 0 & 0 & \cdots & D \end{pmatrix}.$$

Thus, it suffices to compute the set $\mathcal{H}'_\infty$ associated with this two-stage stochastic program. However, applying our decomposition approach to A'_N we notice, that all occuring sets V_u are singletons with $V_u = \{D^{-1}(-Tu)\}$. Therefore, our completion algorithm to compute $\mathcal{H}'_\infty$ is isomorphic to the completion algorithm to compute the Graver basis of

$$\begin{pmatrix} A & 0 \\ T & D \end{pmatrix}$$

via the bijection $(u, \{v\}) \leftrightarrow (u, v)$. Hence, both algorithms lead to the same set of building blocks. Since the algorithm to reconstruct $\mathcal{H}_\infty$ from $\mathcal{H}'_\infty$ works entirely on the building block level, this already proves our claim. □

To reconstruct $\mathcal{H}_\infty$ from $\mathcal{H}'_\infty$ we replace every pair $(u', V_{u'}) \in \mathcal{H}'_\infty$, $u' \neq 0$, by the pair (u, V_u) given by $u = u'$ and $V_u := \{(v, w) : v^{(i)} w^{(i)} \leq 0, \text{ for all } i, v - w \in V_{u'}\}$. Furthermore, note that for $u = 0$ the set V_u consists of the building blocks (e_i, e_i) and their negatives. Herein, the e_i are the unit vectors of appropriate dimension.

3.5.2 LP case

Lemma 3.5.4 *Given a matrix* $B = \begin{pmatrix} A & a & -a \end{pmatrix}$*, with two columns which differ only in their respective signs. Then the LP Graver basis of* B *can be constructed from the LP Graver basis of* $B' = \begin{pmatrix} A & a \end{pmatrix}$ *in the following way:*

$$\mathcal{G}_{LP}(B) = \{(u,v,0) : (u,v) \in \mathcal{G}_{LP}(B')\} \cup \{(u,0,-v) : (u,v) \in \mathcal{G}_{LP}(B')\} \cup \{\pm(0,1,1)\}.$$

Proof. Since $\pm(0,1,1)$ are the only support minimal elements in $\ker_{\mathbb{R}^{n+2}}(B) \setminus \{0\}$ with $u = 0$, we may assume that $u \neq 0$. Let $(u,v,w) \in \mathcal{G}_{\mathrm{LP}}(B)$ be given. Because of $\pm(0,1,1) \in \ker_{\mathbb{R}^{n+2}}(B)$, (u,v,w) can have minimal support in $\ker_{\mathbb{R}^{n+2}}(B) \setminus \{0\}$ only if $v = 0$ or $w = 0$. If $w = 0$, then $(u,v,0) \in \mathcal{G}_{\mathrm{LP}}(B)$ implies that there is no non-zero vector $(u',v',w') \in \ker_{\mathbb{R}^{n+2}}(B)$ with $\mathrm{supp}(u',v',w') \subsetneq \mathrm{supp}(u,v,0)$ and thus, with $w' = 0$. Therefore, $(u,v) \in \ker_{\mathbb{R}^{n+1}}(B')$ has inclusion minimal support in $\ker_{\mathbb{R}^{n+1}}(B') \setminus \{0\}$, which implies $(u,v) \in \mathcal{G}_{\mathrm{LP}}(B')$. Analogously, we conclude that $(u,-w) \in \mathcal{G}_{\mathrm{LP}}(B')$ in case of $v = 0$.

It remains to show that

$$\{(u,v,0) : (u,v) \in \mathcal{G}_{\mathrm{LP}}(B')\} \cup \{(u,0,-v) : (u,v) \in \mathcal{G}(B')\} \cup \{\pm(0,1,1)\} \subseteq \mathcal{G}_{\mathrm{LP}}(B).$$

Clearly, $\pm(0,1,1) \in \mathcal{G}_{\mathrm{LP}}(B)$. Let $(u,v) \in \mathcal{G}_{\mathrm{LP}}(B')$, $u \neq 0$. That means that there is no non-zero vector $(u',v') \in \ker_{\mathbb{R}^{n+1}}(B')$ with $\mathrm{supp}(u',v') \subsetneq \mathrm{supp}(u,v)$. This implies that there are no vectors $(u',v',0)$ and $(u',0,-v')$ with $\mathrm{supp}(u',v',0) \subsetneq \mathrm{supp}(u,v,0)$ and $\mathrm{supp}(u',0,-v') \subsetneq \mathrm{supp}(u,0,-v)$, respectively. Therefore, both vectors $(u,v,0)$ and $(u,0,v)$ belong to $\mathcal{G}_{\mathrm{LP}}(B)$. □

Corollary 3.5.5 *Given a matrix* $B = \begin{pmatrix} A & D & -D \end{pmatrix}$*, with a number of paired columns which differ only in their respective signs. Then the LP Graver basis of* B *can be constructed from the LP Graver basis of* $B' = \begin{pmatrix} A & D \end{pmatrix}$ *in the following way:*

$$\mathcal{G}_{LP}(B) = \{(u,v,w) : (u,v-w) \in \mathcal{G}_{LP}(B'), \mathrm{supp}(v) \cap \mathrm{supp}(w) = \emptyset\} \cup \{\pm(0,1,1)\}.$$

Herein, v*,* w*, and* 1 *are vectors whose dimensions equal the number of columns of* D*.*

Proof. The proof is analogous to the proof of Lemma 3.5.4, since we can consider the pairs of columns in D and $-D$ independently. □

Lemma 3.5.6 *For two-stage stochastic linear programs with simple recourse* $\mathcal{H}_\infty$ *coincides with* $\mathcal{H}_1$*.*

Proof. The proof follows exactly the same lines as the proof of Lemma 3.5.3. □

To construct $\mathcal{H}_\infty$ from ${\mathcal{H}'}_\infty$ we replace every pair $(u', V_{u'}) \in {\mathcal{H}'}_\infty$, $u' \neq 0$, by (u, V_u), where $u = u'$ and $V_u := \{(u, v, w) : (u, v - w) \in \mathcal{G}_{\mathrm{LP}}(B'), \mathrm{supp}(v) \cap \mathrm{supp}(w) = \emptyset\}$. Furthermore, note that for $u = 0$ the set V_u consists of the building blocks (e_i, e_i) and their negatives.

3.6 Computations

Algorithm 3.3.1 as well as the initialization and augmentation procedures from Section 3.4 have been implemented for the integer case. The current version of that implementation can be obtained from [22]. In Chapter 7 we give a more detailed description of that implementation. To indicate the principal behaviour of our method, we report on test runs with an academic example. The algorithmic bottleneck of the method is the completion procedure in Algorithm 3.3.1. Therefore, the sizes of the matrices T and W are very moderate in this initial phase of testing. On the other hand, the method is fairly insensitive with respect to the number of scenarios.

Consider the two-stage program

$$\begin{aligned}\min\{35x_1 + 40x_2 + \tfrac{1}{N}\sum_{\nu=1}^{N} 16y_1^\nu + 19y_2^\nu + 47y_3^\nu + 54y_4^\nu : \\ x_1 + y_1^\nu + y_3^\nu &\geq \xi_1^\nu, \\ x_2 + y_2^\nu + y_4^\nu &\geq \xi_2^\nu, \\ 2y_1^\nu + y_2^\nu &\leq \xi_3^\nu \\ y_1^\nu + 2y_2^\nu &\leq \xi_4^\nu, \\ x_1, x_2, y_1^\nu, y_2^\nu, y_3^\nu, y_4^\nu &\in \mathbb{Z}_+\}\end{aligned}$$

Here, the random vector $\xi \in \mathbb{R}^s$ is given by the scenarios $\xi^1, \ldots, \xi^N$, all with equal probability $1/N$. The realizations of (ξ_1^ν, ξ_2^ν) and (ξ_3^ν, ξ_4^ν) are given by uniform grids (of differing granularity) in the squares $[300, 500] \times [300, 500]$ and $[0, 2000] \times [0, 2000]$, respectively. Timings are given in CPU seconds on a SUN Enterprise 450, 300 MHz Ultra-SPARC.

It took 3.3 seconds to compute $\mathcal{H}_\infty$ altogether consisting of 1464 building blocks arranged into 25 pairs (u, V_u). The column Aug then gives the times needed to augment the solution $x_1 = x_2 = y_1^\nu = y_2^\nu = 0$, $y_3^\nu = \xi_1^\nu$, and $y_4^\nu = \xi_2^\nu$, $\nu = 1, \ldots N$, to optimality.

(ξ_1, ξ_2)	(ξ_3, ξ_4)	scenarios	variables	optimum	Aug	CPLEX	dualdec
5×5	3×3	225	902	$(100, 150)$	1.52	0.63	> 1800
5×5	21×21	11025	44102	$(100, 100)$	66.37	696.10	–
9×9	21×21	35721	142886	$(108, 96)$	180.63	> 1 day	–

Although further exploration is necessary, the above table seems to indicate linear dependence of the computing time on the number N of scenarios, once $\mathcal{H}_\infty$ has been computed. Existing methods in stochastic integer programming are much more sensitive on N. The implemented dual decomposition algorithm for two-stage stochastic mixed-integer programs [10] involves a non-smooth convex minimization in a space of dimension $(N-1)m$, where m denotes the dimension of the first-stage vector x. This explains the failure of the dual decomposition algorithm in the two bigger examples, as indicated in the row dualdec. On the other hand, the dual decomposition algorithm is a more general method in that it works for mixed-integer problems as well. Of course, it is possible to tackle the full-size integer linear program (4) by standard solvers, such as CPLEX, directly. Not surprisingly, this strategy comes to its limit if N is sufficiently large, see the last example in the above table.

3.7 Conclusions

We have presented a novel approach for the solution of two-stage (integer) linear stochastic programs based on test set methods. To this end, we constructed a finite object, $\mathcal{H}_\infty$, which depends only on the matrices A, T, and W, and which allows an algorithmic solution to (4), page 13, for any number N of scenarios, any specific cost function, any specific right-hand side, and any initial feasible solution to the problem. To the best of our knowledge, the set $\mathcal{H}_\infty$, so far, has no counterpart in the existing stochastic programming literature.

Moreover, we have presented an algorithm to compute $\mathcal{H}_\infty$. First computational tests indicate that, once the bottleneck of finding $\mathcal{H}_\infty$ has been passed, the augmentation process acts less sensitive on the number N of scenarios than hitherto methods do.

In the next chapter we will show how to extend the presented decomposition approach to the multi-stage extension of (4). Again, we define a set $\mathcal{H}_\infty$ of building blocks not depending on the number of scenarios and give an algorithm which, upon termination, returns $\mathcal{H}_\infty$. The in building blocks $\mathcal{H}_\infty$ can again be employed to reconstruct improving vectors to a given feasible solution. In this more general setting, however, finiteness of $\mathcal{H}_\infty$ in the integer situation is still an open question.

Chapter 4

Decomposition of Test Sets in Multi-Stage Stochastic Programming

In the following, we generalize the decomposition approach from Chapter 3 to the multi-stage situation. To this end, we will present the details for the 3-stage case only. The treatment below, however, should make clear how to define the decomposition, the notions, and the algorithms for the higher multi-stage situations. This treatment and the proofs follow the same counting or construction arguments as in the two-stage situation. For an introduction to multi-stage stochastic programming we refer to [40] and the references therein.

The constraint matrix of the 3-stage extension of (4), page 13, has the following form:

$$A_{N_1,(N_{2,1},\ldots,N_{2,N_1})} := \begin{pmatrix} A_1 & & & & & & & & & & \\ T_1' & A_2 & & & & & & & & & \\ T_1'' & T_2 & W_2 & & & & & & & & \\ \vdots & \vdots & & \ddots & & & & & & & \\ T_1'' & T_2 & & & W_2 & & & & & & \\ \vdots & & & & & \ddots & & & & & \\ T_1' & & & & & & A_2 & & & & \\ T_1'' & & & & & & T_2 & W_2 & & & \\ \vdots & & & & & & \vdots & & \ddots & & \\ T_1'' & & & & & & T_2 & & & W_2 & \end{pmatrix}.$$

All remaining entries in the above matrix are 0 and there are N_1 blocks of the form

$$\begin{pmatrix} A_2 & & & \\ T_2 & W_2 & & \\ \vdots & & \ddots & \\ T_2 & & & W_2 \end{pmatrix}.$$

The j^{th} block contains $N_{2,j}$ copies of T_2 and of W_2. For simplicity we assume that $N_{2,1} = \ldots = N_{2,N_1} =: N_2$. This can always be achieved by introducing additional scenarios (or scenario trees in the multi-stage case) with zero conditional probability. We remark that this symmetrification is done for ease of exposition only. The subsequent analysis done with A_{N_1,N_2} readily extends to $A_{N_1,(N_{2,1},\ldots,N_{2,N_1})}$.

Therefore, we may assume that matrices of 3-stage stochastic programs have the following structure:

$$A_{N_1,N_2} := \begin{pmatrix} A_1 & 0 & 0 & \cdots & 0 \\ T_1 & W_1 & 0 & \cdots & 0 \\ T_1 & 0 & W_1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ T_1 & 0 & 0 & \cdots & W_1 \end{pmatrix},$$

where $T_1 = (T_1', T_1'', \ldots, T_1'')^\intercal$ and where W_1 is the matrix of a two-stage stochastic program of the form

$$W_1 := \begin{pmatrix} A_2 & 0 & 0 & \cdots & 0 \\ T_2 & W_2 & 0 & \cdots & 0 \\ T_2 & 0 & W_2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ T_2 & 0 & 0 & \cdots & W_2 \end{pmatrix}.$$

The subscripts N_i, $i = 1, 2$, correspond to the numbers of T_i's and W_i's used. The matrices A_1, A_2, T_1, T_2, W_2 have integer entries and are of appropriate dimensions in order to fit into the above matrix structure. In the following let u, v, and w denote variables corresponding to the first, second, and third stages, respectively. Moreover, let their dimensions be n_1, n_2, and n_3. Thus, the dimension of the full 3-stage problem is given by $d_{N_1,N_2} = n_1 + N_1 n_2 + N_1 N_2 n_3$.

Although it would be more convenient to use $u^{(1)}$, $u^{(2)}$, and $u^{(3)}$ for an inductive generalization of all notions to more than two stages, we prefer u, v, w for a better readability.

4.1 Building Blocks of Graver Test Sets

As in Chapter 3 we present the decomposition approach for the pure integer case. However, the same decomposition ideas readily extend to the continuous and the mixed-integer situations. Again, a simple observation is the basis for the decomposition of test set vectors.

Lemma 4.1.1 *The vector* $z = (u, v_1, w_{1,1}, \ldots, w_{1,N_2}, \ldots, v_{N_1}, w_{N_1,1}, \ldots, w_{N_1,N_2})$ *belongs to* $\ker_{\mathbb{Z}^{d_{N_1,N_2}}}(A_{N_1,N_2})$ *if and only if* $(u, v_i, w_{i,j}) \in \ker_{\mathbb{Z}^{d_{1,1}}}(A_{1,1})$ *for* $i = 1, \ldots, N_1$, $j = 1, \ldots, N_2$.

Proof. The claim follows immediately if we write $0 = A_{N_1,N_2} z$ explicitly. □

Definition 4.1.2 *(Building blocks)*

Let $z = (u, v_1, w_{1,1}, \ldots, w_{1,N_2}, \ldots, v_{N_1}, w_{N_1,1}, \ldots, w_{N_1,N_2}) \in \ker_{\mathbb{Z}^{d_{N_1,N_2}}}(A_{N_1,N_2})$ *and call the vectors* $u, v_1, \ldots, v_{N_1}, w_{1,1}, \ldots, w_{N_1,N_2}$ *the building blocks of* z. *Denote by* $\mathcal{G}_{N_1,N_2}$ *the Graver test set associated with* A_{N_1,N_2} *and collect into* $\mathcal{H}_{N_1,N_2}$ *all those vectors arising as building blocks of some* $z \in \mathcal{G}_{N_1,N_2}$.

By $\mathcal{H}_\infty$ *denote the set* $\bigcup_{N_1=1}^{\infty} \bigcup_{N_2=1}^{\infty} \mathcal{H}_{N_1,N_2}$.

The set $\mathcal{H}_\infty$ contains vectors u, v, w associated with the first, second, and third stages, respectively. Again, for convenience we will arrange the vectors in $\mathcal{H}_\infty$ into a certain structure:

For fixed $u, v \in \mathcal{H}_\infty$ collect into $W_{u,v}$ all those vectors $w \in \mathcal{H}_\infty$ which satisfy $(u, v, w) \in \ker_{\mathbb{Z}^{d_{1,1}}}(A_{1,1})$. Now, for fixed $u \in \mathcal{H}_\infty$, collect into V_u all those pairs $(v, W_{u,v})$ such that $v \in \mathcal{H}_\infty$ and $W_{u,v} \neq \emptyset$.

Therefore, we will again arrange the building blocks of $\mathcal{H}_\infty$ into pairs (u, V_u) but now each element of V_u is not a vector but a pair $(v, W_{u,v})$. In what follows we will extend this arrangement into pairs to arbitrary sets of building blocks of appropriate dimensions, not necessarily belonging to $\mathcal{H}_\infty$.

Again, the set $\mathcal{H}_\infty$ is of particular interest, since by definition it contains all building blocks of Graver test set vectors for A_{N_1,N_2} for arbitrary numbers N_1 and N_2. Thus, $\mathcal{H}_\infty$ does not depend on N_1 and N_2.

4.2 Computation of $\mathcal{H}_\infty$

Assuming for the moment that $\mathcal{H}_\infty$ is finite also in the 3-stage situation, let us present the necessary notions for the Completion Algorithm 4.2.1 below to compute this set.

We will prove that this algorithm, upon termination, returns a set containing $\mathcal{H}_\infty$ in both the LP and the IP cases. Employing a simple counting argument we will show termination of this completion algorithm in the LP situation. Correctness and termination of the algorithm implies finiteness of $\mathcal{H}_\infty$ for all 3-stage stochastic linear programs. Moreover, this counting argument readily extends to higher multi-stage stochastic linear programs, proving finiteness of the analogous sets $\mathcal{H}_\infty$ in these more general cases.

For stochastic integer programs this finiteness question is still unanswered already for the 3-stage case. In order to prove finiteness of $\mathcal{H}_\infty$ and termination of its computation for 3-stage or even higher multi-stage stochastic integer programs, generalizations of Maclagan's Theorem, Theorem 3.2.1, are needed. We will deal with these generalizations in Chapter 6.

To compute $\mathcal{H}_\infty$ in the 3-stage case we use the same algorithmic pattern as for Graver test set computations and the computation of $\mathcal{H}_\infty$ for the two-stage case. Thus, we have to define the main ingredients to this completion algorithm: the input set, normalForm, $\oplus$, and $\ominus$.

Note that the objects f, g, and s occurring in the following algorithm have the form (u, V_u), wherein V_u is a set of pairs $(v, W_{u,v})$.

Algorithm 4.2.1 *(Algorithm to compute $\mathcal{H}_\infty$)*

Input: a symmetric generating set F of $\ker_{\mathbb{Z}^{d_{1,1}}}(A_{1,1})$ over $\mathbb{Z}$ or of $\ker_{\mathbb{R}^{d_{1,1}}}(A_{1,1})$ over $\mathbb{R}$ in (u, V_u)-notation to be specified below

Output: a set $G \supseteq \mathcal{H}_\infty$

$G := F$

$C := \bigcup_{f,g \in G} \{f \oplus g\}$ *(forming* S-vectors*)*

while $C \neq \emptyset$ *do*

$s :=$ *an element in* C

$C := C \setminus \{s\}$

$f := \text{normalForm}(s, G)$

if $f \neq (0, \{(0, \{0\})\})$ *then*

$C := C \cup \bigcup_{g \in G \cup \{f\}} \{f \oplus g\}$ *(adding* S-vectors*)*

$G := G \cup \{f\}$

return G.

Behind the function normalForm(s, G) there is again the following algorithm.

Algorithm 4.2.2 *(Normal form algorithm)*

Input: a pair s, a set G of pairs

Output: a normal form of s with respect to G

while there is some $g \in G$ such that $g \sqsubseteq s$ do

$s := s \ominus g$

return s

In the following we define the necessary notions for the above completion algorithm in the integer and continuous cases.

Lemma 4.2.3 *Let F be a symmetric generating set for $\ker_{\mathbb{Z}^{d_{1,1}}}(A_{1,1})$ over $\mathbb{Z}$ which contains both a generating set for $\{(0, v, w) : T_2 v + W_2 w = 0\} \subseteq \ker_{\mathbb{Z}^{d_{1,1}}}(A_{1,1})$ over $\mathbb{Z}$ consisting only of vectors with zero as first-stage component and a generating set for $\{(0, 0, w) : W_2 w = 0\} \subseteq \ker_{\mathbb{Z}^{d_{1,1}}}(A_{1,1})$ over $\mathbb{Z}$ consisting only of vectors with zero as first- and as second-stage components.*

Moreover, let F_{N_1,N_2} be the set of all those vectors in $\ker_{\mathbb{Z}^{d_{N_1,N_2}}}(A_{N_1,N_2})$ whose building blocks are also building blocks of vectors in $F \cup \{0\}$.

Then for any N_1, N_2 the vectors F_{N_1,N_2} generate $\ker_{\mathbb{Z}^{d_{N_1,N_2}}}(A_{N_1,N_2})$ over $\mathbb{Z}$.

The same result holds for the continuous kernel $\ker_{\mathbb{R}^{d_{N_1,N_2}}}(A_{N_1,N_2})$.

Proof. Let $z = (u, v_1, w_{1,1}, \ldots, w_{1,N_2}, \ldots, v_{N_1}, w_{N_1,1}, \ldots, w_{N_1,N_2}) \in \ker_{\mathbb{Z}^{d_{N_1,N_2}}}(A_{N_1,N_2})$. We have to show that z can be written as a linear combination of elements from F_{N_1,N_2}.

Since $(u, v_1, w_{1,1}) \in \ker_{\mathbb{Z}^{d_{1,1}}}(A_{1,1})$ there has to exist a finite integer linear combination $(u, v_1, w_{1,1}) = \sum \alpha_i(\bar{u}_i, \bar{v}_i, \bar{w}_i)$ with $(\bar{u}_i, \bar{v}_i, \bar{w}_i) \in F$. Hence

$$(u, v_1, w_{1,1}, \ldots, w_{1,1}, \ldots, v_1, w_{1,1}, \ldots, w_{1,1}) = \sum_i \alpha_i(\bar{u}, \bar{v}_i, \bar{w}_i, \ldots, \bar{w}_i, \ldots, \bar{v}_i, \bar{w}_i, \ldots, \bar{w}_i)$$

with $(\bar{u}, \bar{v}_i, \bar{w}_i, \ldots, \bar{w}_i, \ldots, \bar{v}_i, \bar{w}_i, \ldots, \bar{w}_i) \in F_{N_1,N_2}$.

Now we have

$$\begin{aligned} z &= (u, v_1, w_{1,1}, \ldots, w_{1,1}, \ldots, v_1, w_{1,1}, \ldots, w_{1,1}) + \\ &\quad \sum_{j=1}^{N_1} (0,0,0,\ldots,0,\ldots,v_j - v_1, w_{j,1} - w_{1,1}, \ldots, w_{j,N_2} - w_{1,1}, \ldots, 0, 0, \ldots, 0) \\ &= \sum_i \alpha_i (\bar{u}, \bar{v}_i, \bar{w}_i, \ldots, \bar{w}_i, \ldots, \bar{v}_i, \bar{w}_i, \ldots, \bar{w}_i) + \\ &\quad \sum_{j=1}^{N_1} (0,0,0,\ldots,0,\ldots,v_j - v_1, w_{j,1} - w_{1,1}, \ldots, w_{j,N_2} - w_{1,1}, \ldots, 0, 0, \ldots, 0) \end{aligned}$$

Note that $(0,0,0,\ldots,0,\ldots,v_j - v_1, w_{j,1} - w_{1,1}, \ldots, w_{j,N_2} - w_{1,1}, \ldots, 0, 0, \ldots, 0) \in \ker_{\mathbb{Z}^{d_{N_1,N_2}}}(A_{N_1,N_2})$ and $(v_j - v_1, w_{j,1} - w_{1,1}, \ldots, w_{j,N_2} - w_{1,1}) \in \ker_{\mathbb{Z}^{n_2+N_2 n_3}}(W_1)$. But in the two-stage case (cf. Lemma 3.3.3) we have already seen that we can write $(v_j - v_1, w_{j,1} - w_{1,1}, \ldots, w_{j,N_2} - w_{1,1}) \in \ker_{\mathbb{Z}^{n_2+N_2 n_3}}(W_1)$ as a finite integer linear combination of the given set of vectors. Therefore, we have proved that z can be written as a finite integer linear combination of vectors from F_{N_1,N_2}.

An analogous argumentation shows the same result for the continuous case. □

Before we define the input set, the sum $\oplus$, and the difference $\ominus$ for the IP and the LP cases let us introduce the following notation.

Definition 4.2.4 *For $\alpha \in \mathbb{R}$ let*

$$(u, V_u) + \alpha(u', V_{u'}) := (u + \alpha u', V_u + \alpha V_{u'}),$$

where

$$V_u + \alpha V_{u'} := \{V + \alpha V' : V \in V_u, V' \in V_{u'}\}$$

with $V + \alpha V'$ as in the the 2-stage situation, Definition 3.3.4.

4.2.1 IP case

Lemma 4.2.3 suggests the following input set.

Definition 4.2.5 *Let F be a symmetric generating set for $\ker_{\mathbb{Z}^{d_{1,1}}}(A_{1,1})$ over $\mathbb{Z}$ which contains both a generating set for $\{(0,v,w) : T_2 v + W_2 w = 0\} \subseteq \ker_{\mathbb{Z}^{d_{1,1}}}(A_{1,1})$ over $\mathbb{Z}$ consisting only of vectors with zero as first-stage component and a generating set for $\{(0,0,w) : W_2 w = 0\} \subseteq \ker_{\mathbb{Z}^{d_{1,1}}}(A_{1,1})$ over $\mathbb{Z}$ consisting only of vectors with zero as first- and as second-stage components.*

With this define the building blocks of all vectors in $F \cup \{0\}$ in (u, V_u)-notation to be the input set to Algorithm 4.2.1 for the integer case.

It remains to define $(u, V_u) \oplus (u', V_{u'})$ and $(u, V_u) \ominus (u', V_{u'})$.

Definition 4.2.6 *Let $(u, V_u) \oplus (u', V_{u'}) := (u, V_u) + (u', V_{u'})$. Moreover, we say that $(u', V_{u'})$ reduces (u, V_u), or $(u', V_{u'}) \sqsubseteq (u, V_u)$ for short, if the following conditions are satisfied:*

1. *$u' \sqsubseteq u$*
2. *For every $(v, W_{u,v}) \in V_u$ there exists $(v', W_{u',v'}) \in V_{u'}$ with*
   - *$v' \sqsubseteq v$, and*
   - *for all $w \in W_{u,v}$ there is some vector $w' \in W_{u',v'}$ with $w' \sqsubseteq w$.*
3. *If we have $u' = 0$ and $v' = 0$ in the condition above, then there are vectors $w \in W_{u,v}$ and $w' \in W_{u',v'}$ with $0 \neq w' \sqsubseteq w$.*

If $(u', V_{u'}) \sqsubseteq (u, V_u)$, let (u, V_u) reduce to

$$(u, V_u) \ominus (u', V_{u'}) := (u - u', \{V \ominus V' : V \in V_u, V' \in V_{u'}, V' \sqsubseteq V\}),$$

where $V \ominus V'$ is defined as in the 2-stage case, Definition 3.3.6.

Again, the third condition above is necessary to avoid the situation that u' and all occurring v' and w' are zero, which forms a trivial zero reduction.

Remark. In $(u, V_u) \ominus (u', V_{u'})$ it suffices again to collect only one difference $V \ominus V'$ for every $V \in V_u$. In addition, it suffices to take only one difference $w - w'$ when computing $V \ominus V'$ for every $w \in V$. As can be seen from the proof of the following proposition, Algorithm 4.2.1 still works correctly if we defined $(u, V_u) \ominus (u', V_{u'})$ in this way. □

Proposition 4.2.7 *If the input set, the procedure* normalForm*, $f \oplus g$, and $s \ominus g$ are defined as above for the IP case, then Algorithm 4.2.1, upon termination, returns a set of vectors that contains $\mathcal{H}_\infty$.*

Proof. Suppose that Algorithm 4.2.1 terminates.

To show that $\mathcal{H}_\infty \subseteq G$, we have to prove that $\mathcal{H}_{N_1,N_2} \subseteq G$ for arbitrary $N_1, N_2 \in \mathbb{Z}_+$. Fix N_1, N_2 and start a Graver test set computation (see Section 1.3) with

$$\begin{aligned}\bar{F} \;:=\; \{&(u, v_1, w_{1,1}, \ldots, w_{1,N_2}, \ldots, v_{N_1}, w_{N_1,1}, \ldots, w_{N_1,N_2}) : \\ &(u, V_u) \in G, (v_i, W_{u,v_i}) \in V_u, w_{i,j} \in W_{u,v_i}, i = 1, \ldots, N_1, j = 1, \ldots, N_2\}\end{aligned}$$

as input set. $\bar{F}$ generates $\ker_{\mathbb{Z}^{d_{N_1,N_2}}}(A_{N_1,N_2})$ over $\mathbb{Z}$ for all $N_1, N_2 \in \mathbb{Z}_+$, since, by the assumption on the input set to Algorithm 4.2.1, $F_{N_1,N_2} \subseteq \bar{F}$.

We show next that all sums $z + z' \in$ S-vectors(z, z') of two elements $z, z' \in \bar{F}$ reduce to 0 with respect to $\bar{F}$. In this case, Algorithm 1.3.1 returns the input set $\bar{F}$ which implies $\mathcal{G}_{N_1,N_2} \subseteq \bar{F}$. Therefore, $\mathcal{H}_{N_1,N_2} \subseteq G$ as desired.

Take two arbitrary elements

$$z = (u, v_1, w_{1,1}, \ldots, w_{1,N_2}, \ldots, v_{N_1}, w_{N_1,1}, \ldots, w_{N_1,N_2})$$

and

$$z' = (u', v'_1, w'_{1,1}, \ldots, w'_{1,N_2}, \ldots, v'_{N_1}, w'_{N_1,1}, \ldots, w'_{N_1,N_2})$$

from $\bar{F}$ and consider the vector $z + z' \in$ S-vectors(z, z').

In the above algorithm, $(u, V_u) \oplus (u', V_{u'})$ was reduced to $(0, \{(0, \{0\})\})$ by elements $(u_1, V_{u_1}), \ldots, (u_k, V_{u_k}) \in G$. From this sequence we can construct a sequence $z_1, \ldots, z_k$ of vectors in $\bar{F}$ which reduce $z + z'$ to zero as follows.

$(u_1, V_{u_1}) \sqsubseteq (u, V_u) \oplus (u', V_{u'})$ implies that $u_1 \sqsubseteq u + u'$ and that there exist pairs $(v_{1,1}, W_{u_1,v_{1,1}})$, …, $(v_{1,N_1}, W_{u_1,v_{1,N_1}}) \in V_{u_1}$ and $w_{1,j,1}, \ldots, w_{1,j,N_2} \in W_{u_1,v_{1,j}}$ such that

- $v_{1,i} \sqsubseteq v_i + v'_i$ for $i = 1, \ldots, N_1$, and
- $w_{1,j,k} \sqsubseteq w_{j,k} + w'_{j,k}$ for $j = 1, \ldots, N_1$, $k = 1, \ldots, N_2$.

Therefore, $z_1 := (u_1, v_{1,1}, w_{1,1,1}, \ldots, w_{1,1,N_2}, \ldots, v_{1,N_1}, w_{1,N_1,1}, \ldots, w_{1,N_1,N_2}) \sqsubseteq z + z'$ and $z + z'$ can be reduced to $z + z' - z_1$. Moreover, $z_1 \in \bar{F}$ and all the building blocks of $z + z' - z_1$ lie in $((u, V_u) \oplus (u', V_{u'})) \ominus (u_1, V_{u_1})$.

But $((u, V_u) \oplus (u', V_{u'})) \ominus (u_1, V_{u_1})$ was further reduced to $(0, \{(0, \{0\})\})$ by the pairs $(u_2, V_{u_2}), \ldots, (u_k, V_{u_k}) \in G$. Therefore, we can construct from (u_2, V_{u_2}) a vector $z_2 \in \bar{F}$ with $z_2 \sqsubseteq z + z' - z_1$. Thus, $z + z' - z_1$ can be further reduced to $z + z' - z_1 - z_2$.

Repeating this construction, we arrive in the k^{th} step at $z + z' - z_1 - \ldots - z_{k-1}$ whose building blocks all lie in $((u, V_u) \oplus (u', V_{u'})) \ominus (u_1, V_{u_1}) \ominus \ldots \ominus (u_{k-1}, V_{u_{k-1}})$. The latter can be reduced to $(0, \{(0, \{0\})\})$ by the pair $(u_k, V_{u_k}) \in G$. Therefore, there exists a vector $z_k \in \bar{F}$ such that $z_k \sqsubseteq z + z' - z_1 - \ldots - z_{k-1}$ and $0 = z + z' - z_1 - \ldots - z_k$. $\square$

4.2.2 LP case

Lemma 4.2.3 suggests the following input set.

Definition 4.2.8 *Let F be a symmetric generating set for $\ker_{\mathbb{R}^{d_{1,1}}}(A_{1,1})$ over $\mathbb{R}$ which contains both a generating set for $\{(0,v,w) : T_2v + W_2w = 0\} \subseteq \ker_{\mathbb{R}^{d_{1,1}}}(A_{1,1})$ over $\mathbb{R}$ consisting only of vectors with zero as first-stage component and a generating set for $\{(0,0,w) : W_2w = 0\} \subseteq \ker_{\mathbb{R}^{d_{1,1}}}(A_{1,1})$ over $\mathbb{R}$ consisting only of vectors with zero as first- and as second-stage components.*

With this define the building blocks of all vectors in $F \cup \{0\}$ in (u, V_u)-notation to be the input set to Algorithm 4.2.1 for the continuous case.

For the computation of $\mathcal{H}_\infty$ for 3-stage stochastic linear programs we define $\oplus$ and $\ominus$ as follows.

Definition 4.2.9 *For given (u, V_u), $(u', V_{u'})$, and all $\alpha \in \mathbb{R}$ such that*

1. *$u + \alpha u'$ contains a zero entry at some component i at which neither u nor u' have a zero entry, that is, $(u + \alpha u')^{(i)} = 0$ but $u^{(i)}(u')^{(i)} \neq 0$, or*
2. *for some vectors $v \in V_u, v' \in V_{u'}$ the vector $v + \alpha v'$ contains a zero entry at some component i at which neither v nor v' have a zero entry, that is, $(v + \alpha v')^{(i)} = 0$ but $v^{(i)}(v')^{(i)} \neq 0$, or*
3. *for some vectors $w \in W_{u,v}, w' \in W_{u',v'}$ the vector $w + \alpha w'$ contains a zero entry at some component i at which neither w nor w' have a zero entry, that is, $(w + \alpha w')^{(i)} = 0$ but $w^{(i)}(w')^{(i)} \neq 0$,*

define $(u, V_u) \oplus (u', V_{u'})$ to be the pair

$$\{(u, V_u) + \alpha(u', V_{u'}) : \alpha \in \mathbb{R} \text{ satisfying the above conditions } 1, 2, 3\}.$$

Definition 4.2.10 *We say that $(u', V_{u'})$ reduces (u, V_u), or $(u', V_{u'}) \sqsubseteq (u, V_u)$ for short, if the following conditions are satisfied:*

1. $\operatorname{supp}(u') \subseteq \operatorname{supp}(u)$
2. *For every $(v, W_{u,v}) \in V_u$ there exists $(v', W_{u',v'}) \in V_{u'}$ with*
   - $\operatorname{supp}(v') \subseteq \operatorname{supp}(v)$, *and*

- *for all* $w \in W_{u,v}$ *there is some* $w' \in W_{u',v'}$ *with* $\operatorname{supp}(w') \subseteq \operatorname{supp}(w)$.

3. *If we have* $u' = 0$ *and* $v' = 0$ *in the condition above, then there are vectors* $w \in W_{u,v}$ *and* $w' \in W_{u',v'}$ *with* $\emptyset \neq \operatorname{supp}(w') \subseteq \operatorname{supp}(w)$.

In case of reduction (u, V_u) *reduces to* $(u, V_u) \ominus \beta(u', V_{u'})$ *which is defined to be the pair*

$$(u - \beta u', \{v - \beta v', \{w - \beta w' : w \in W_{u,v}, w' \in W_{u',v'}\} : v \in V_u, v' \in V_{u'}\}),$$

where all v' *and* w' *satisfy the three conditions above, and where* β *takes one of the finitely many values such that* $\operatorname{supp}(u - \beta u') \subsetneq \operatorname{supp}(u)$, $\operatorname{supp}(v - \beta v') \subsetneq \operatorname{supp}(v)$, *or* $\operatorname{supp}(w - \beta w') \subsetneq \operatorname{supp}(w)$ *for at least one pair of vectors.*

Again, the third condition in the definition above is necessary to avoid the situation that u' and all occurring v' and w' are zero, which forms a trivial zero reduction.

Proposition 4.2.11 *If the input set, the procedure* normalForm, $f \oplus g$, *and* $s \ominus g$ *are defined as above for the LP case, then Algorithm 4.2.1 terminates and returns a set of vectors containing* $\mathcal{H}_\infty$.

Proof. In the following we show that $G \setminus F$ contains at most $2^{n_1}2^{2^{n_2}2^{2^{n_3}}}$ elements and so the algorithm always terminates.

Define $(v, W_{u,v})$ and $(v', W_{u',v'})$ to be equivalent as for the two-stage case (see proof of Lemma 3.2.13). This defines an equivalence relation with $2^{n_2}2^{2^{n_3}}$ equivalence classes. Define V_u to be equivalent to $V_{u'}$ if for each $(v, W_{u,v}) \in V_u$ there is some $(v', W_{u',v'}) \in V_{u'}$ equivalent to $(v, W_{u,v})$. This defines an equivalence relation with $2^{2^{n_2}2^{2^{n_3}}}$ equivalence classes. Moreover, define (u, V_u) and $(u', V_{u'})$ to be equivalent if $\operatorname{supp}(u) = \operatorname{supp}(u')$ and V_u is equivalent to $V_{u'}$. This defines an equivalence relation with $2^{n_1}2^{2^{n_2}2^{2^{n_3}}}$ equivalence classes.

Moreover, if (u, V_u) and $(u', V_{u'})$ are equivalent it holds that $(u, V_u) \sqsubseteq (u', V_{u'})$ and $(u', V_{u'}) \sqsubseteq (u, V_u)$. Therefore, there are no two equivalent pairs (u, V_u) and $(u', V_{u'})$ in $G \setminus F$, because if (u, V_u) was added to G first then $(u', V_{u'})$ could have been reduced by (u, V_u) before being added to G. This means that $G \setminus F$ contains at most $2^{n_1}2^{2^{n_2}2^{2^{n_3}}}$ different pairs (u, V_u) and the algorithm terminates.

To show correctness, that is $\mathcal{H}_\infty \subseteq G$, we have to prove that $\mathcal{H}_{N_1,N_2} \subseteq G$ for arbitrary $N_1, N_2 \in \mathbb{Z}_+$. Fix N_1, N_2 and start a Graver test set computation (see Section 1.3) with

$$\begin{aligned}\bar{F} \;:=\; &\{(u, v_1, w_{1,1}, \ldots, w_{1,N_2}, \ldots, v_N, w_{N_1,1}, \ldots, w_{N_1,N_2}) : \\ &(u, V_u) \in G, (v_i, W_{u,v_i}) \in V_u, w_{i,j} \in W_{u,v_i}, i = 1, \ldots, N_1, j = 1, \ldots, N_2\}\end{aligned}$$

as input set. $\bar{F}$ generates $\ker_{\mathbb{R}^{d_{N_1,N_2}}}(A_{N_1,N_2})$ over $\mathbb{R}$ for all $N_1, N_2 \in \mathbb{Z}_+$, since, by the assumption on the input set to Algorithm 4.2.1, $F_{N_1,N_2} \subseteq F$.

If all S-vectors $z + \alpha z' \in \text{S-vectors}(z, z')$ of two elements $z, z' \in \bar{F}$ can be written as a linear combination of elements of $\bar{F}$ whose support is contained in $\text{supp}(z + \alpha z')$, then the Graver basis $\mathcal{G}_{N_1,N_2}$ is contained in $\bar{F}$ (see Remark 1.3.5). Since N_1, N_2 were chosen arbitrarily, $\mathcal{G}_{N_1,N_2} \subseteq \bar{F}$ will imply $\mathcal{H}_{N_1,N_2} \subseteq G$ for all $N_1, N_2 \in \mathbb{Z}_+$ and we are done.

Take two arbitrary elements

$$z = (u, v_1, w_{1,1}, \ldots, w_{1,N_2}, \ldots, v_{N_1}, w_{N_1,1}, \ldots, w_{N_1,N_2})$$

and

$$z' = (u', v'_1, w'_{1,1}, \ldots, w'_{1,N_2}, \ldots, v'_{N_1}, w'_{N_1,1}, \ldots, w'_{N_1,N_2})$$

from $\bar{F}$ and any $\alpha \in \mathbb{R}$ such that $z + \alpha z' \in \text{S-vectors}(z, z')$.

In the above algorithm, $(u, V_u) + \alpha(u', V_{u'}) \in (u, V_u) \oplus (u', V_{u'})$ was successively reduced to $(0, \{(0, \{0\})\})$ by elements (u_1, V_{u_1}), …, $(u_k, V_{u_k}) \in G$. From this sequence we will now construct a sequence $z_1, \ldots, z_k$ of vectors in $\bar{F}$ such that $z + \alpha z' = \sum \alpha_i z_i$ and $\text{supp}(z_i) \subseteq \text{supp}(z + \alpha z')$ as desired.

(u_1, V_{u_1}) reduces $(u, V_u) + \alpha(u', V_{u'})$ to the pair $((u, V_u) + \alpha(u', V_{u'})) \ominus \alpha_1(u_1, V_{u_1})$. But $(u_1, V_{u_1}) \sqsubseteq (u, V_u) + \alpha(u', V_{u'})$ implies that $\text{supp}(u_1) \subseteq \text{supp}(u + \alpha u')$ and that there exist $(v_{1,1}, W_{u_1,v_{1,1}})$, …, $(v_{1,N_1}, W_{u_1,v_{1,N_1}}) \in V_{u_1}$ and $w_{1,j,1}, \ldots, w_{1,j,N_2} \in W_{u_1,v_{1,j}}$, $j = 1, \ldots, N_1$, such that

- $\text{supp}(v_{1,i}) \subseteq \text{supp}(v_i + \alpha v'_i)$ for $i = 1, \ldots, N_1$, and
- $\text{supp}(w_{1,j,k}) \subseteq \text{supp}(w_{j,k} + \alpha w'_{j,k})$ for $j = 1, \ldots, N_1$, $k = 1, \ldots, N_2$.

Thus, we have $z_1 := (u_1, v_{1,1}, w_{1,1,1}, \ldots, w_{1,1,N_2}, \ldots, v_{1,N}, w_{1,N_1,1}, \ldots, w_{1,N_1,N_2}) \in \bar{F}$, $\text{supp}(z_1) \subseteq \text{supp}(z + \alpha z')$, and all the building blocks of $z + \alpha z' - \alpha_1 z_1$ lie in the pair $((u, V_u) \oplus (u', V_{u'})) \ominus \alpha_1(u_1, V_{u_1})$.

But $((u, V_u) + \alpha(u', V_{u'})) \ominus \alpha_1(u_1, V_{u_1})$ was further reduced by the pair $(u_2, V_{u_2}) \in G$ to $((u, V_u) + \alpha(u', V_{u'})) \ominus \alpha_1(u_1, V_{u_1}) \ominus \alpha_2(u_2, V_{u_2})$. Thus, we can construct from (u_2, V_{u_2}) a vector $z_2 \in \bar{F}$ with $\text{supp}(z_2) \subseteq \text{supp}(z + \alpha z' - \alpha_1 z_1)$ where again, all building blocks of $z + \alpha z' - \alpha_1 z_1 - \alpha_2 z_2$ lie in $((u, V_u) + \alpha(u', V_{u'})) \ominus \alpha_1(u_1, V_{u_1}) \ominus \alpha_2(u_2, V_{u_2})$.

Repeating this construction, we arrive in the k^{th} step at $z + \alpha z' - \sum_{i=1}^{k-1} \alpha_i z_i$ whose building blocks all lie in $((u, V_u) + \alpha(u', V_{u'})) \ominus \alpha_1(u_1, V_{u_1}) \ominus \ldots \ominus \alpha_{k-1}(u_{k-1}, V_{u_{k-1}})$. The latter can be reduced to $(0, \{(0, \{0\})\})$ by the pair $(u_k, V_{u_k}) \in G$. Therefore, there

exists a vector $z_k \in \bar{F}$ such that $\mathrm{supp}(z_k) \subseteq \mathrm{supp}(z + \alpha z' - \alpha_1 z_1 - \ldots - \alpha_{k-1} z_{k-1})$ and $0 = z + \alpha z' - \sum_{i=1}^{k} \alpha_i z_i$.

Thus, $z + \alpha z' = \sum_{i=1}^{k} \alpha_i z_i$ with $\mathrm{supp}(z_i) \subseteq \mathrm{supp}(z + \alpha z')$ for $i = 1, \ldots, k$, which completes the proof. □

Note that termination and correctness of the above algorithm again imply finiteness of $\mathcal{H}_\infty$. Therefore, the above proves the following theorem.

Theorem 4.2.12 *Given integer matrices A_1, A_2, T_1, T_2, and W_2 of appropriate dimensions, and let $\mathcal{H}_\infty$ be defined for 3-stage stochastic linear programs as in Definition 4.1.2. Then $\mathcal{H}_\infty$ is a finite set.*

Although we have presented the generalization of our decomposition approach to the 3-stage situation only, the above presentation of decomposition, notions, algorithms, and proofs readily extends to linear multi-stage stochastic programs. This gives the following result.

Theorem 4.2.13 *The set $\mathcal{H}_\infty$ is a finite set for all multi-stage linear stochastic programs.*

4.3 Solving the Optimization Problem with the Help of $\mathcal{H}_\infty$

As we have seen in Section 2.6, universal test sets can help to find an initial feasible solution to our optimization problem, and to augment it to optimality. However, the universal test set is again not given explicitly. By Remark 2.6.4, we can concentrate on finding improving directions. In the following, we will see how these directions can be reconstructed from $\mathcal{H}_\infty$ in the 3-stage situation.

4.3.1 IP case

Suppose we are given $\mathcal{H}_\infty$, a cost function vector

$$c = (c_u, c_{v,1}, c_{w,1,1}, \ldots, c_{w,1,N_2}, \ldots, c_{v,N_1}, c_{w,N_1,1}, \ldots, c_{w,N_1,N_2}),$$

and a feasible solution $z_0 = (u, v_1, w_{1,1}, \ldots, w_{1,N_2}, \ldots, v_{N_1}, w_{N_1,1}, \ldots, w_{N_1,N_2})$.

Lemma 4.3.1 *Suppose that there exists no pair $(u', V_{u'}) \in \mathcal{H}_\infty$ such that*

1. $u' \leq u$,

2. *for* $i = 1, \ldots, N_1$, *there exist* $(\bar{v}_i, W_{u',\bar{v}_i}) \in V_{u'}$ *and* $\bar{w}_{i,1}, \ldots, \bar{w}_{i,N_2} \in W_{u',\bar{v}_i}$ *with* $\bar{v}_i \leq v_i$ *and* $\bar{w}_{i,j} \leq w_{i,j}$ *for* $j = 1, \ldots, N_2$,

3. $c^\intercal z' > 0$, *where* $z' = (u', v'_1, w'_{1,1}, \ldots, w'_{1,N_2}, \ldots, v'_{N_1}, w'_{N_1,1}, \ldots, w'_{N_1,N_2})$ *and for* $i = 1, \ldots, N_1$, *the vector* $(v'_i, w'_{i,1}, \ldots, w'_{i,N_2})$, *is an optimal solution of*

$$\max\{c_{v,i}^\intercal \bar{v}_i + \textstyle\sum_{j=1}^{N_2} c_{w,i,j}^\intercal \bar{w}_{i,j} : \begin{array}{rcllcl} (\bar{v}_i, W_{u',\bar{v}_i}) & \in & V_{u'}, & \bar{v}_i & \leq & v_i, \\ \bar{w}_{i,j} & \in & W_{u',\bar{v}_i}, & \bar{w}_{i,j} & \leq & w_{i,j}, \\ & & & & & j = 1, \ldots, N_2\}. \end{array}$$

Then $z_0 = (u, v_1, w_{1,1}, \ldots, w_{1,N_2}, \ldots, v_{N_1}, w_{N_1,1}, \ldots, w_{N_1,N_2})$ *is an optimal optimal solution of the problem* $\min\{c^\intercal z : A_{N_1,N_2} z = b, z \in \mathbb{Z}_+^{d_{N_1,N_2}}\}$.

If there exists such a pair $(u', V_{u'}) \in \mathcal{H}_\infty$ *then* $z_0 - z'$ *is a feasible solution with* $c^\intercal(z_0 - z') < c^\intercal z_0$.

Proof. Suppose that z_0 is not optimal. Then there is some vector

$$z'' = (u'', v''_1, w''_{1,1}, \ldots, w''_{1,N_2}, \ldots, v''_{N_1}, w''_{N_1,1}, \ldots, w''_{N_1,N_2}) \in \mathcal{G}_{N_1,N_2}$$

such that $z_0 - z''$ is feasible and $c^\intercal(z_0 - z'') < c^\intercal z_0$. Feasibility of $z_0 - z''$ implies $z_0 - z'' \geq 0$, hence $z'' \leq z_0$. Therefore, we have $u'' \leq u$, $v''_i \leq v_i$, $i = 1, \ldots, N_1$, and $w''_{j,k} \leq w_{j,k}$, $j = 1, \ldots, N_1$, $k = 1, \ldots, N_2$, the latter implying that for any $i = 1, \ldots, N_1$, $j = 1, \ldots, N_2$, there exist vectors $\bar{v}_i$ and $\bar{w}_{i,j}$ as required in the second condition of Lemma 4.3.1.

Choose $u' = u''$ and let z' as defined in the third condition. Now $c^\intercal(z_0 - z'') < c^\intercal z_0$ implies that $c^\intercal z'' > 0'$. Moreover $c^\intercal z' \geq c^\intercal z'' > 0$. In conclusion, the pair $(u', V_{u'})$ fulfills conditions $1. - 3.$, proving the first claim of the lemma.

With z' according to 3. we obtain $c^\intercal(z_0 - z') < c^\intercal z_0$. Moreover, we have $u' \leq u$, $v'_i \leq v_i$, $i = 1, \ldots, N_1$, and $w'_{j,k} \leq w_{j,k}$, $j = 1, \ldots, N_1$, $k = 1, \ldots, N_2$, together imply $z' \leq z_0$, and $z_0 - z' \geq 0$. Finally, we have $z' \in \ker_{\mathbb{Z}^{d_{N_1,N_2}}}(A_{N_1,N_2})$, and therefore $A_{N_1,N_2}(z_0 - z') = A_{N_1,N_2} z_0 + 0 = b$ which completes the proof. □

Remark. Note that for fixed i the above optimization problem

$$\max\{c_{v,i}^\intercal \bar{v}_i + \textstyle\sum_{j=1}^{N_2} c_{w,i,j}^\intercal \bar{w}_{i,j} : \begin{array}{rcllcl} (\bar{v}_i, W_{u',\bar{v}_i}) & \in & V_{u'}, & \bar{v}_i & \leq & v_i, \\ \bar{w}_{i,j} & \in & W_{u',\bar{v}_i}, & \bar{w}_{i,j} & \leq & w_{i,j}, \\ & & & & & j = 1, \ldots, N_2\} \end{array}$$

can easily be solved by considering those finitely many $\bar{v}_i$ such that $(\bar{v}_i, W_{u',\bar{v}_i}) \in V_{u'}$ one after another. For fixed $\bar{v}_i$ the above optimization problem decomposes and we have to solve

$$\max\{c_{w,i,j}^\intercal \bar{w}_{i,j} : \bar{w}_{i,j} \leq w_{i,j}, \bar{w}_{i,j} \in W_{u',\bar{v}_i}\}$$

for $j = 1, \ldots, N_2$.

Moreover note, that the above procedure to find an improving vector is again linear with respect to the number $N_1 N_2$ of scenarios. □

Theorem 4.3.2 *If the problem* $\min\{c^\intercal z : A_{N_1,N_2} z = b, z \in \mathbb{Z}_+^{d_{N_1,N_2}}\}$ *is solvable, an optimal solution can be computed in finitely many steps by application of the Augmentation Algorithm 0.0.2 together with the above reconstruction procedure to find improving vectors.*

4.3.2 LP case

Suppose we are given $\mathcal{H}_\infty$, a cost function vector

$$c = (c_u, c_{v,1}, c_{w,1,1}, \ldots, c_{w,1,N_2}, \ldots, c_{v,N_1}, c_{w,N_1,1}, \ldots, c_{w,N_1,N_2}),$$

and a feasible solution $z_0 = (u, v_1, w_{1,1}, \ldots, w_{1,N_2}, \ldots, v_{N_1}, w_{N_1,1}, \ldots, w_{N_1,N_2})$.

Lemma 4.3.3 *Suppose that there exists no pair* $(u', V_{u'}) \in \mathcal{H}_\infty$ *such that*

1. $\operatorname{supp}((u')^+) \subseteq \operatorname{supp}(u)$,
2. *for* $i = 1, \ldots, N_1$, *there exist* $(\bar{v}_i, W_{u',\bar{v}_i}) \in V_{u'}$ *and* $\bar{w}_{i,1}, \ldots, \bar{w}_{i,N_2} \in W_{u',\bar{v}_i}$ *with* $\operatorname{supp}(\bar{v}_i^+) \subseteq \operatorname{supp}(v_i)$ *and* $\operatorname{supp}(\bar{w}_{i,j}^+) \subseteq \operatorname{supp}(w_{i,j})$ *for* $j = 1, \ldots, N_2$,
3. $c^\intercal z' > 0$, *where* $z' = (u', v_1', w_{1,1}', \ldots, w_{1,N_2}', \ldots, v_{N_1}', w_{N_1,1}', \ldots, w_{N_1,N_2}')$ *and for* $i = 1, \ldots, N_1$, *the vector* $(v_i', w_{i,1}', \ldots, w_{i,N_2}')$ *is an optimal solution of*

$$\begin{aligned} \max\{c_{v,i}^\intercal \bar{v}_i + \textstyle\sum_{j=1}^{N_2} c_{w,i,j}^\intercal \bar{w}_{i,j} : (\bar{v}_i, W_{u',\bar{v}_i}) &\in V_{u'}, \quad \operatorname{supp}(\bar{v}_i^+) \subseteq \operatorname{supp}(v_i), \\ \bar{w}_{i,j} &\in W_{u',\bar{v}_i}, \operatorname{supp}(\bar{w}_{i,j}^+) \subseteq \operatorname{supp}(w_{i,j}), \\ & \qquad j = 1, \ldots, N_2\}. \end{aligned}$$

Then $z_0 = (u, v_1, w_{1,1}, \ldots, w_{1,N_2}, \ldots, v_{N_1}, w_{N_1,1}, \ldots, w_{N_1,N_2})$ *is an optimal optimal solution of the problem* $\min\{c^\intercal z : A_{N_1,N_2} z = b, z \in \mathbb{R}_+^{d_{N_1,N_2}}\}$.

If there exists such a pair $(u', V_{u'}) \in \mathcal{H}_\infty$ *then there exists* $\alpha \in \mathbb{R}_{>0}$, *such that* $z_0 - \alpha z'$ *is a feasible solution with* $c^\intercal(z_0 - \alpha z') < c^\intercal z_0$.

Proof. Suppose that z_0 is not optimal. Then there is some vector

$$z'' = (u'', v''_1, w''_{1,1}, \ldots, w''_{1,N_2}, \ldots, v''_{N_1}, w''_{N_1,1}, \ldots, w''_{N_1,N_2}) \in \mathcal{G}_{N_1,N_2}$$

and a positive scalar β such that $z_0 - \beta z''$ is feasible and $c^\intercal(z_0 - \beta z'') < c^\intercal z_0$. Feasibility of $z_0 - \beta z''$ implies $z_0 - \beta z'' \geq 0$, hence $\mathrm{supp}((z'')^+) \subseteq \mathrm{supp}(z_0)$. Therefore, we have $\mathrm{supp}((u'')^+) \subseteq \mathrm{supp}(u)$, $\mathrm{supp}((v'')^+_i) \subseteq \mathrm{supp}(v_i)$, $i = 1, \ldots, N_1$, and $\mathrm{supp}((w'')^+_{j,k}) \subseteq \mathrm{supp}(w_{j,k})$, $j = 1, \ldots, N_1$, $k = 1, \ldots, N_2$, the latter implying that for any $i = 1, \ldots, N_1$, $j = 1, \ldots, N_2$, there exist vectors $\bar{v}_i$ and $\bar{w}_{i,j}$ as required in the second condition of Lemma 4.3.3.

Choose $u' = u''$ and let z' be as defined in the third condition above. But now $c^\intercal(z_0 - \beta z'') < c^\intercal z_0$ and $\beta > 0$ imply that $c^\intercal z'' > 0$. Moreover $c^\intercal z' \geq c^\intercal z'' > 0$. In conclusion, the pair $(u', V_{u'})$ fulfills conditions $1. - 3.$, proving the first claim of the lemma.

With z' according to 3., $c^\intercal z' > 0$ implies that $c^\intercal(z_0 - \alpha z') < c^\intercal z_0$ for any $\alpha > 0$. Moreover, we have $\mathrm{supp}((u')^+) \subseteq \mathrm{supp}(u)$, $\mathrm{supp}((v')^+_i) \subseteq \mathrm{supp}(v_i)$, $i = 1, \ldots, N_1$, and $\mathrm{supp}((w')^+_{j,k}) \subseteq \mathrm{supp}(w_{j,k})$, $j = 1, \ldots, N_1$, $k = 1, \ldots, N_2$, together imply that we can indeed choose a strictly positive scalar α such that $z_0 - \alpha z' \geq 0$. Finally, $z' \in \ker_{\mathbb{R}^{d_{N_1,N_2}}}(A_{N_1,N_2})$, and therefore $A_{N_1,N_2}(z_0 - \alpha z') = A_N z_0 - \alpha \cdot 0 = b$ which completes the proof. □

Remark. Note that for fixed i the above optimization problem

$$\begin{array}{lll} \max\{c^\intercal_{v,i}\bar{v}_i + \sum_{j=1}^{N_2} c^\intercal_{w,i,j}\bar{w}_{i,j} : & (\bar{v}_i, W_{u',\bar{v}_i}) \in V_{u'}, & \mathrm{supp}(\bar{v}_i^+) \subseteq \mathrm{supp}(v_i), \\ & \bar{w}_{i,j} \in W_{u',\bar{v}_i}, & \mathrm{supp}(\bar{w}_{i,j}^+) \subseteq \mathrm{supp}(w_{i,j}), \\ & & j = 1, \ldots, N_2\}. \end{array}$$

can easily be solved by considering those finitely many $\bar{v}_i$ such that $(\bar{v}_i, W_{u',\bar{v}_i}) \in V_{u'}$ one after another. For fixed $\bar{v}_i$ the above optimization problem decomposes and we have to solve

$$\max\{c^\intercal_{w,i,j}\bar{w}_{i,j} : \mathrm{supp}(\bar{w}_{i,j}^+) \subseteq \mathrm{supp}(w_{i,j}), \bar{w}_{i,j} \in W_{u',\bar{v}_i}\}$$

for $j = 1, \ldots, N_2$.

Moreover note, that the above procedure to find an improving vector is again linear with respect to the number $N_1 N_2$ of scenarios. □

Theorem 4.3.4 *If the problem* $\min\{c^\intercal z : A_{N_1,N_2} z = b, z \in \mathbb{R}^{d_{N_1,N_2}}_+\}$ *is solvable, an optimal solution can be computed in finitely many steps by application of the Augmentation Algorithm 0.0.2 together with the Augmentation Strategy 2.5.1 and the above reconstruction procedure for finding improving vectors.*

Note, that the Augmentation Strategy 2.5.1 can also be applied at the building block level, since in phase 1 we have to look for an improving vector t to our feasible solution z_0 for which $c^\intercal t > 0$ and $\text{supp}(t) \subseteq \text{supp}(z_0)$ holds.

4.4 Conclusions

In this chapter we have presented the details of our decomposition for 3-stage stochastic programs. Again, we defined a set $\mathcal{H}_\infty$ of building blocks that is independent on the number of scenarios. Moreover, we showed how to compute this set and how to employ it in order to find an improving vector. As in the two-stage situation, both algorithms work entirely on the building block level. The presentation should have made clear how to define the necessary notions for the decomposition and the algorithms for higher multi-stage stochastic programs.

We have shown finiteness of $\mathcal{H}_\infty$ for 3-stage stochastic linear programs. Analogous counting arguments show finiteness of $\mathcal{H}_\infty$ also for general multi-stage stochastic linear programs. Whether $\mathcal{H}_\infty$ is finite for the integer case is already an open question for the 3-stage situation. In order to prove finiteness of $\mathcal{H}_\infty$ for general multi-stage stochastic integer linear programs successive extensions of Maclagan's Theorem, see Chapter 6, would turn out very useful.

In the next chapter we show that our decomposition approach to two- and multi-stage stochastic programming can be generalized to arbitrary problem matrices. This will lead us to the definition of a connection set. These sets turn out to be much smaller than the test set they correspond to. Moreover, improving vectors can be reconstructed from connection sets without knowledge of the full test set. Most surprisingly, however, is the fact that this reconstruction of improving directions via connection sets may be much faster even if the corresponding test set is given!

Chapter 5

Decomposition of Test Sets for Arbitrary Matrices

In the previous two chapters we presented a novel decomposition approach for the solution of two- and multi-stage stochastic programs. In this chapter we extend this decomposition idea to problems with arbitrary integer problem matrix A. We define the notions of building blocks and of connection vectors. As the building block in the last two chapters, also the connection vectors can be computed directly without computing the corresponding test set first. Connection vectors can be employed to find improving vectors to non-optimal solutions. More importantly, however, it may be much faster to reconstruct an improving vector from the connection vectors even if the full test set is given.

Definition 5.0.1 *Let* $A = (A_1|A_2) \in \mathbb{Z}^{l\times k} \times \mathbb{Z}^{l\times(d-k)}$, $b \in \mathbb{R}^l$, *and*

$$T \subseteq \{(u_1, u_2) \in \mathbb{R}^k \times \mathbb{R}^{d-k} : A_1u_1 + A_2u_2 = b\}.$$

Then we call

$$\mathrm{CS}_k(T) := \{A_1u_1 : (u_1, u_2) \in T\}$$

the connection set of T *at* k, *its elements* A_1u_1 *connection vectors, and the vectors* u_1 *and* u_2 *building blocks.*

We will use the above definition only for $b = 0$. For $b = 0$ the above definition introduces connection sets for the different types of test sets. We will see, that there are always finite connection sets for IP, LP, and MIP Graver test sets. Moreover, finite connection sets can be defined for two-stage stochastic linear and integer linear programs. In the stochastic mixed-integer case the situation is a bit more complicated.

In Section 5.3 we will deal with this case and show that there are finite connection sets for a large class of two-stage stochastic mixed-integer linear programs, and we give an algorithm to compute them.

In the following two sections, however, we look at connection sets for IP, LP, and MIP Graver test sets (also for stochastic programming) and we show how these connection sets can be used to find an improving vector, the main ingredient to the Augmentation Algorithm 0.0.2.

5.1 Connection Sets of LP, IP, and MIP Test Sets

Let $\mathbb{X}, \mathbb{Y} \in \{\mathbb{Z}, \mathbb{R}\}$, and suppose we are given the connection set $\mathrm{CS}_k(\mathcal{T})$ of a universal test set $\mathcal{T}$, say the Graver test set, for the family of linear (mixed-integer) problems

$$\min\{c_1^\intercal z_1 + c_2^\intercal z_2 : A_1 z_1 + A_2 z_2 = b,\ z_1 \in \mathbb{X}_+^k, z_2 \in \mathbb{Y}_+^{d-k}\}$$

as $(c_1, c_2) \in \mathbb{R}^d$ and $b \in \mathbb{R}^l$ or only parts of the data vary. We may assume that the connection set $\mathrm{CS}_k(\mathcal{T})$ is finite. While that is clear in the LP and IP situations, it was proven in Section 2.4 that a finite connection set suffices also for mixed-integer programs.

Moreover, suppose that we are given a specific cost function vector $(c_1, c_2) \in \mathbb{R}^d$, a right-hand side $b \in \mathbb{R}^l$, and a feasible solution $z_0 = (z_{0,1}, z_{0,2}) \in \mathbb{X}_+^k \times \mathbb{Y}_+^{d-k}$ to the above problem.

The following theorems allow us to check for a fixed connection vector $s \in \mathrm{CS}_k(\mathcal{T})$ whether there is an improving vector to z_0. Moreover, these lemmas show how to compute an improving vector, if one exists. Since $\mathrm{CS}_k(\mathcal{T})$ is finite, this yields a finite procedure to reconstruct an improving vector to z_0 or to decide that no such vector exists.

Lemma 5.1.1 *Let $A_1, A_2, b, c_1, c_2, \mathbb{X} = \mathbb{Y} = \mathbb{Z}, \mathcal{T}, k, z_0$ be defined as above, and let $s \in \mathrm{CS}_k(\mathcal{T})$. If the vector $(\bar{u}_1, \bar{u}_2)$ that satisfies*

- $\bar{u}_1 \in \arg\max\{c_1^\intercal z_1 : A_1 z_1 = s,\ z_1 \le z_{0,1},\ z_1 \in \mathbb{Z}^k\}$ *and*
- $\bar{u}_2 \in \arg\max\{c_2^\intercal z_2 : A_2 z_2 = -s,\ z_2 \le z_{0,2},\ z_2 \in \mathbb{Z}^{d-k}\}$

does not have a positive cost function value $c_1^\intercal \bar{u}_1 + c_2^\intercal \bar{u}_2$ then there exists no improving vector for z_0 with connection vector s.

Lemma 5.1.2 *Let $A_1, A_2, b, c_1, c_2, \mathbb{X} = \mathbb{Y} = \mathbb{R}, \mathcal{T}, k, z_0$ be defined as above, and let $s \in \mathrm{CS}_k(\mathcal{T})$. If the vector $(\bar{u}_1, \bar{u}_2)$ that satisfies*

- $\bar{u}_1 \in \begin{cases} \arg\max\{c_1^\intercal z_1 : A_1 z_1 = s,\ z_1 \leq z_{0,1},\ z_1 \in \mathbb{R}^k\}, & \text{if } s = 0, \\ \arg\max\{c_1^\intercal z_1 : A_1 z_1 = s,\ \mathrm{supp}(z_1^+) \subseteq \mathrm{supp}(z_{0,1}),\ z_1 \in \mathbb{R}^k\}, & \text{if } s \neq 0, \end{cases}$
  and
- $\bar{u}_2 \in \begin{cases} \arg\max\{c_2^\intercal z_2 : A_2 z_2 = -s,\ z_2 \leq z_{0,2},\ z_2 \in \mathbb{R}^{d-k}\}, & \text{if } s = 0, \\ \arg\max\{c_2^\intercal z_2 : A_2 z_2 = -s,\ \mathrm{supp}(z_2^+) \subseteq \mathrm{supp}(z_{0,2}),\ z_2 \in \mathbb{R}^{d-k}\}, & \text{if } s \neq 0, \end{cases}$

does not have a positive cost function value $c_1^\intercal \bar{u}_1 + c_2^\intercal \bar{u}_2$ then there exists no improving vector for z_0 with connection vector s.

The distinction between the two cases $s = 0$ and $s \neq 0$ is necessary, since with

Lemma 5.1.3 *Let $A_1, A_2, b, c_1, c_2, \mathbb{X} = \mathbb{Z}, \mathbb{Y} = \mathbb{R}, \mathcal{T}, k, z_0$ be defined as above, and let $s \in \mathrm{CS}_k(\mathcal{T})$. If the vector $(\bar{u}_1, \bar{u}_2)$ that satisfies*

- $\bar{u}_1 \in \arg\max\{c_1^\intercal z_1 : A_1 z_1 = s,\ z_1 \leq z_{0,1},\ z_1 \in \mathbb{Z}^k\}$ *and*
- $\bar{u}_2 \in \arg\max\{c_2^\intercal z_2 : A_2 z_2 = -s,\ z_2 \leq z_{0,2},\ z_2 \in \mathbb{R}^{d-k}\}$

does not have a positive cost function value $c_1^\intercal \bar{u}_1 + c_2^\intercal \bar{u}_2$ then there exists no improving vector for z_0 with connection vector s.

Proof of Lemmas 5.1.1, 5.1.2, 5.1.3. If $(\bar{u}_1, \bar{u}_2)$ is defined as above for the IP, LP, and MIP cases, respectively, and if $c_1^\intercal \bar{u}_1 + c_2^\intercal \bar{u}_2 > 0$ holds, then $(\bar{u}_1, \bar{u}_2)$ is an improving direction for z_0. On the other hand, if (u_1, u_2) is an improving direction to z_0 with connection vector s, then the two optimization problems in the IP, LP, and MIP situations, respectively, are non-empty and thus solvable. Moreover, $c_1^\intercal \bar{u}_1 + c_2^\intercal \bar{u}_2 \geq c_1^\intercal u_1 + c_2^\intercal u_2 > 0$. Thus, $(\bar{u}_1, \bar{u}_2)$ is an improving direction for z_0. □

Lemmas 5.1.1, 5.1.1, and 5.1.3 allow us to apply the Augmentation Algorithm 0.0.2. If z_0 is not optimal then there exists an improving vector $(v_1, v_2) \in \mathcal{T}$. Choosing $s = A_1 v_1$ in the (specific) lemma above, $\bar{u}_1$ and $\bar{u}_2$ exist (since v_1 and v_2, respectively, are feasible solutions to the optimization problems that determine $\bar{u}_1$ and $\bar{u}_2$), and $c_1^\intercal u_1 + c_2^\intercal u_2 \geq c_1^\intercal v_1 + c_2^\intercal v_2 > 0$. Thus, if the current feasible solution z_0 is not optimal, there is at least one connection vector $s \in \mathrm{CS}_k(\mathcal{T})$ from which an improving vector can be reconstructed as indicated in the above three lemmas. In other words, if for all $s \in \mathrm{CS}_k(\mathcal{T})$ no improving vector could be found, the solution z_0 is optimal.

The following example demonstrates that, even if the full test set is known, it can be much faster to reconstruct an improving vector from a corresponding connection set than to extract the improving vector from the test set itself.

Example. The IP Graver basis to the integer program

$$\min\{c^\intercal z : (k\ 1\ 1)z = b, z \in \mathbb{Z}^3_+\}$$

equals $\mathcal{G}_{\mathrm{IP}}(A) := \{\pm(0,1,-1), \pm(1,0,-k), \pm(1,-1,-(k-1)), \ldots, \pm(1,-k,0)\}$, where $A = (k\ 1\ 1)$. Thus, we can form the two connection sets

- $\mathrm{CS}_1(\mathcal{G}_{\mathrm{IP}}(A)) = \{0, \pm k\}$ and
- $\mathrm{CS}_2(\mathcal{G}_{\mathrm{IP}}(A)) = \{0, \pm 1, \pm 2, \ldots, \pm k\}$.

This already shows that the size of a connection set may heavily depend on the position at which we split the problem matrix.

Given a solution $(z_{0,1}, z_{0,2}, z_{0,3})$ we have to solve for each $s \in \mathrm{CS}_1(\mathcal{G}_{\mathrm{IP}}(A)) = \{0, \pm k\}$ the two problems

- $\max\{c_1^\intercal z_1 : z_1 = s,\ z_1 \leq z_{0,1},\ z_1 \in \mathbb{Z}\}$ and
- $\max\{c_2^\intercal z_2 + c_3^\intercal z_3 : z_2 + z_3 = -s,\ (z_2, z_3) \leq (z_{0,2}, z_{0,3}),\ z_2, z_3 \in \mathbb{Z}\}$.

The time to find an improving vector in the test set $\mathcal{G}_{\mathrm{IP}}(A)$ increases with growing k. The time to reconstruct an improving vector from the connection set, however, does not (or at most only very mildly) depend on k. This speed-up may be even more drastic for connection sets for stochastic programs as introduced in the following section. □

5.2 Connection Sets in Stochastic Programming

In two-stage stochastic programming we deal with matrices of the form

$$A_N := \begin{pmatrix} T & W & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ T & 0 & \cdots & W \end{pmatrix},$$

where in the following we assume for simplicity of exposition that the matrix A, introduced in (4), page 13, is incorporated into $(T|W)$ as the submatrix $(A|0)$. Then,

the above matrix A_N can be decomposed into

$$A_{N,1} := \begin{pmatrix} T \\ \vdots \\ T \end{pmatrix} \qquad \text{and} \qquad A_{N,2} := \begin{pmatrix} W & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & W \end{pmatrix}.$$

Applying the decomposition ideas and the 3 lemmas from the previous section to the matrices $A_{N,1}$ and $A_{N,2}$, we see that each connection vector has the form $(Tu, \ldots, Tu)^\intercal$ and that the two problems that have to be solved in order to reconstruct an improving vector either simplify or decompose into smaller problems.

As we have seen in Chapter 3, the set $\mathcal{H}_\infty$ constructed there is finite in the IP and LP cases. Thus, there are only finitely many vectors Tu with $u \in \mathcal{H}_\infty$. Therefore, we define the set $\mathrm{CS}_m(\mathcal{H}_\infty) := \{Tu : u \in \mathcal{H}_\infty, u \text{ is a first-stage building block}\}$ to be the connection set of $\mathcal{H}_\infty$. Herein, m denotes again the number of columns of T, that is, the number of first stage variables.

Connection sets for two-stage stochastic mixed-integer programs can be defined in the same way. Unfortunately, they need not be finite. We give an example in Section 5.3. However, we prove finiteness of the connection set for a large subclass of two-stage stochastic mixed-integer programs, and we present an algorithm to compute them.

In the following, assume that we are given a number N of scenarios, a specific cost function $c := (c_u, c_1, \ldots, c_N)$, and a feasible solution $z_0 = (u, v_1, \ldots, v_N)$ to the problem.

In the subsequent lemmas we show again, how an improving vector can be found for a fixed connection vector $s \in \mathrm{CS}_m(\mathcal{H}_\infty)$. For finite $\mathrm{CS}_m(\mathcal{H}_\infty)$, this yields a finite procedure to reconstruct an improving vector to z_0 or to decide that no such vector exists.

Lemma 5.2.1 *Let $z_0 = (u, v_1, \ldots, v_N)$ be a feasible solution, $s \in \mathrm{CS}_m(\mathcal{H}_\infty)$, and suppose that the optimization problems*

- $\bar{u} \in \arg\max\{c_u^\intercal z : Tz = s,\ z \leq u,\ z \in \mathbb{Z}^m\}$,
- $\bar{v}_i \in \arg\max\{c_i^\intercal z : Wz = -s,\ z \leq v_i,\ z \in \mathbb{Z}^n\},\ i = 1, \ldots, N,$

are all solvable. If the vector $(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N)$ does not have a positive cost function value then there exists no improving vector for z_0 with connection vector s.

Lemma 5.2.2 *Let $z_0 = (u, v_1, \ldots, v_N)$ be a feasible solution, $s \in \mathrm{CS}_m(\mathcal{H}_\infty)$, and suppose that the optimization problems*

- $\bar{u} \in \begin{cases} \arg\max\{c_u^\intercal z : Tz = s,\, z \le u,\, z \in \mathbb{R}^m\}, & \text{if } s = 0, \\ \arg\max\{c_u^\intercal z : Tz = s,\, \operatorname{supp}(z^+) \subseteq \operatorname{supp}(u),\, z \in \mathbb{R}^m\}, & \text{if } s \neq 0, \text{ and} \end{cases}$
- $\bar{v}_i \in \begin{cases} \arg\max\{c_i^\intercal z : Wz = -s,\, z \le v_i,\, z \in \mathbb{R}^n\}, & \text{if } s = 0, \\ \arg\max\{c_i^\intercal z : Wz = -s,\, \operatorname{supp}(z^+) \subseteq \operatorname{supp}(v_i),\, z \in \mathbb{R}^n\}, & \text{if } s \neq 0, \end{cases}$
  $i = 1, \ldots, N.$

are all solvable. If the vector $(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N)$ does not have a positive cost function value then there exists no improving vector for z_0 with connection vector s.

Lemma 5.2.3 *Suppose that there is no row in $(T|W)$ that couples a first-stage continuous variable with a second-stage continuous variable, that is, there is no problem constraint that involves both first- and second-stage continuous variables. Let $z_0 = (u, v_1, \ldots, v_N)$ be a feasible solution, $s \in \mathrm{CS}_m(\mathcal{H}_\infty)$, and suppose that the optimization problems*

- $\bar{u} \in \arg\max\{c_u^\intercal z : Tz = s,\, z \le u,\, z \in \mathbb{Z}^m\}$,
- $\bar{v}_i \in \arg\max\{c_i^\intercal z : Wz = -s,\, z \le v_i,\, z \in \mathbb{R}^n\}, i = 1, \ldots, N.$

are all solvable. If the vector $(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N)$ does not have a positive cost function value then there exists no improving vector for z_0 with connection vector s.

Proof of Lemmas 5.2.1, 5.2.2, 5.2.3. If $\bar{z} := (\bar{u}, \bar{v}_1, \ldots, \bar{v}_N)$ is defined as above for the IP, LP, and MIP cases, respectively, and if $c^\intercal \bar{z} > 0$ holds, then $\bar{z}$ is an improving direction for z_0. On the other hand, if $z' := (u', v'_1, \ldots, v'_N)$ is an improving direction to z_0 with connection vector s, then all the optimization problems in the IP, LP, and MIP situations, respectively, are non-empty and thus solvable. Moreover, $c^\intercal \bar{z} \ge c^\intercal z' > 0$. Thus, $\bar{z}$ is an improving direction for z_0. □

5.3 Connection Sets in Stochastic Mixed-Integer Programming

In Section 3.3 we have seen how to compute the set $\mathcal{H}_\infty$ for two-stage stochastic linear and integer linear programs entirely on the building block level. Therefore, we can use these algorithms to compute the connection sets of $\mathcal{H}_\infty$ in these two cases. But, since $\mathcal{H}_\infty$ contains all building blocks of $\mathcal{G}_{\mathrm{LP}}(T|W)$ and of $\mathcal{G}_{\mathrm{IP}}(T|W)$, respectively, we

can use the same two algorithms to compute the connection sets $\mathrm{CS}_k(\mathcal{G}_{\mathrm{LP}}(A_1|A_2))$ and $\mathrm{CS}_k(\mathcal{G}_{\mathrm{IP}}(A_1|A_2))$ by choosing $T = A_1$ and $W = A_2$.

Although the knowledge of $\mathcal{H}_\infty$ already suffices to solve any given specific problem instance to optimality, the reconstruction of an improving vector may be done much faster via the connection set of $\mathcal{H}_\infty$ (see the example on page 94). We refer to already presented algorithms to stress the fact that also connection sets can be computed on a building block level. Clearly, an algorithm to compute the connection set without computing the set $\mathcal{H}_\infty$ first is even more desireable.

Connections sets of $\mathrm{CS}_k(\mathcal{G}_{\mathrm{MIP}}(A_1|A_2))$ can be computed via the algorithm presented in Section 2.4. Therefore, we restrict our attention in this section to the computation of connections sets of $\mathcal{H}_\infty$ for two-stage stochastic mixed-integer linear programs.

5.3.1 The Main Result

As we have seen in Section 2.4, MIP Graver test sets need not be finite in general. Therefore, a similar decomposition approach as presented in Chapter 3 applied to two-stage mixed-integer linear stochastic programs leads to an infinite set $\mathcal{H}_\infty$ of building blocks. However, although $\mathcal{H}_\infty$ may be not finite, the connection set of $\mathcal{H}_\infty$ can be finite. For each of the finitely many connection vectors a number of integer and mixed-integer linear programs have to be solved, see Lemma 5.2.3. Thus, we can employ the Augmentation Algorithm 0.0.2 again to solve two-stage stochastic mixed-integer programs in finitely many augmentation steps.

It remains to determine interesting cases in which $\mathrm{CS}_m(\mathcal{H}_\infty)$ is finite. Our main result in this respect is the following.

Theorem 5.3.1 *Suppose that there is no row in $(T|W)$ that couples a first-stage continuous variable with a second-stage continuous variable. Then $\mathrm{CS}_m(\mathcal{H}_\infty)$ is finite.*

Before we prove this claim by presenting an algorithm that computes a finite superset to $\mathrm{CS}_m(\mathcal{H}_\infty)$, let us show that $\mathrm{CS}_m(\mathcal{H}_\infty)$ need not be finite if there is a row that couples a first-stage continuous variable with a second-stage continuous variable. For this we will use the example from Section 2.4 which showed that MIP Graver test sets need not be finite in general.

Example. Let $T = (1\ 1)$ and $W = (1)$. Moreover, let only the first first-stage variable be integer and let all other variables be continuous. As we have seen in Section 2.4, all vectors $(1, -1 + \beta, -\beta)$ with $\beta \in \mathbb{R}$, $0 < \beta < 1$ lie in $\mathcal{G}_{\mathrm{MIP}}(T|W)$.

Since all corresponding connection vectors of $\mathcal{G}_{\mathrm{MIP}}(T|W)$, $(1+(-1+\beta))=(\beta)$, are also connection vectors in $\mathrm{CS}_2(\mathcal{H}_\infty)$, the set $\mathrm{CS}_2(\mathcal{H}_\infty)$ is not finite. □

5.3.2 Computation of $\mathrm{CS}_m(\mathcal{H}_\infty)$

In the following, we present all ingredients for a completion algorithm to compute $\mathrm{CS}_m(\mathcal{H}_\infty)$. The treatment, the definitions, and the proofs follow the main lines from Section 2.4 about the computation of MIP Graver test sets. However, as we deal with two-stage stochastic programs, we will also use the arguments from Section 3.3 in the subsequent proofs. First, let us assume as in Theorem 5.3.1 that no first-stage and second-stage continuous variables are coupled by a row. Thus, we can write T and W as follows:

$$T=\begin{pmatrix} T_1 & T_2 \\ T_3 & 0 \end{pmatrix} \qquad \text{and} \qquad W=\begin{pmatrix} W_1 & 0 \\ W_3 & W_4 \end{pmatrix},$$

where the columns of T_2 and of W_4 correspond to the first- and second-stage continuous variables, respectively. Moreover, every connection vector can be written as $(s,t)^\intercal$, where s, of dimension l_s, corresponds to the rows of T_1, T_2, and W_1, and where t, of dimension l_t, corresponds to the rows of T_3, W_3, and W_4. For any vector w let the indices z and q indicate its integer and continuous parts, respectively. Therefore, $w=(w_z,w_q)$. Moreover, denote by m_z, m_q, n_z, n_q the numbers of first-stage and second-stage integer and continuous variables, respectively. For better readability, let $\ker(A_1)$ and $\ker(A_N)$ denote the mixed-integer kernels of A_1 and A_N, as sets of appropriate dimensions.

Note that, since $-s=W_1v_z$ and $t=T_3u_z$, every connection vector $(s,t)^\intercal$ has to be integer.

The objects in the completion procedure presented below will be pairs of the form $((u_z,P_{(T_2|-T_2),s-T_1u_z}),V_{u_z,s})$, where

$$V_{u_z,s}:=\{(v_z,P_{(W_4|-W_4),-t-W_3v_z}):W_1v_z=-s,t=T_3u_z\}$$

for certain vectors v_z. Note that with each pair $((u_z,P_{(T_2|-T_2),s-T_1u_z}),V_{u_z,s})$ there is a fixed connection vector $(s,t)=(s,T_3u_z)$. For better readability, we will write $P_{u_z,s}:=P_{(T_2|-T_2),s-T_1u_z}$ and $Q_{v_z,t}:=P_{(W_4|-W_4),-t-W_3v_z}$. (Remember that we defined $P_{A,b}:=\{z:Az=b,z\in\mathbb{R}^d_+\}$ for given dimension d.)

Algorithm 5.3.2 *(Algorithm to compute $\mathcal{H}_\infty$)*

Input: a symmetric generating set F of $\ker(A_1)$ in $((u_z, P_{u_z,s}), V_{u_z,s})$-notation to be specified below

Output: a set G from which a superset of $CS_m(\mathcal{H}_\infty)$ can be extracted

$G := F$

$C := \bigcup_{f,g \in G} \{f \oplus g\}$ *(forming* S-vectors*)*

while $C \neq \emptyset$ *do*

$s :=$ *an element in* C

$C := C \setminus \{s\}$

$f := \text{normalForm}(s, G)$

if $f \neq ((0, P_{0,0}), \{(0, Q_{0,0})\})$ *then*

$C := C \cup \bigcup_{g \in G \cup \{f\}} \{f \oplus g\}$ *(adding* S-vectors*)*

$G := G \cup \{f\}$

return G.

Behind the function normalForm(s, G) there is again the following algorithm.

Algorithm 5.3.3 *(Normal form algorithm)*

Input: a pair s, a set G of pairs

Output: a normal form of s with respect to G

while there is some $g \in G$ such that $g \sqsubseteq s$ do

$s := s \ominus g$

return s

In the following we define the necessary notions for the above completion algorithm.

Definition 5.3.4 *Let $\bar{F}$ be a symmetric integer generating set for the integer lattice $\{(u_z, v_z) \in \mathbb{Z}^{m_z+n_z} : \exists (u_q, v_q) \in \mathbb{R}^{m_q+n_q}$ with $(u_z, u_q, v_z, v_q) \in \ker(A_1)\}$ which contains an integer generating set for the integer lattice $\{(0, v_z) \in \mathbb{Z}^{m_z+n_z} : \exists v_q \in \mathbb{R}^{n_q}$ with $(0, 0, v_z, v_q) \in \ker(A_1)\}$ consisting only of vectors with zero as first-stage component. Define $F := \{(u_z, P_{u_z,s}), V_{u_z,s}) : (u_z, v_z) \in \bar{F} \cup \{0\}$ for some v_z with $W_1 v_z = -s\}$, where $V_{u_z,s} := \{(v_z, Q_{v_z,t}) : (u_z, v_z) \in \bar{F} \cup \{0\}, W_1 v_z = -s, t = T_3 u_z\}$. Let F be the input set to the completion algorithm.*

Analogously to the proof of Lemma 3.3.3 one can show that

$$F_N := \{(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N) : \quad ((\bar{u}_z, P_{\bar{u}_z,s}), V_{\bar{u}_z,s}) \in F, (\bar{u}_q^+, \bar{u}_q^-) \in P_{\bar{u}_z,s}, \\ (\bar{v}_{i,z}, Q_{\bar{v}_{i,z},t}) \in V_{\bar{u}_z,s}, (\bar{v}_{i,q}^+, \bar{v}_{i,q}^-) \in Q_{\bar{v}_{i,z},t}, i = 1, \ldots, N\}$$

finitely generates $\ker(A_N)$ over $\mathbb{Z}$ for every $N \in \mathbb{Z}_+$, that is, every vector from $\ker(A_N)$ can be written as a finite integer linear combination of elements from the (possibly infinite) set F_N.

Definition 5.3.5 *We define*

$$V_{u_z,s} \oplus V_{u'_z,s'} := \{(v_z + v'_z, Q_{v_z+v'_z,t+t'}) : (v_z, Q_{v_z,t}) \in V_{u_z,s}, (v'_z, Q_{v'_z,t'}) \in V_{u'_z,s'}\}$$

and

$$((u_z, P_{u_z,s}), V_{u_z,s}) \oplus ((u'_z, P_{u'_z,s'}), V_{u'_z,s'}) := ((u_z + u'_z, P_{u_z+u'_z,s+s'}), V_{u_z,s} \oplus V_{u'_z,s'}).$$

Definition 5.3.6 *We say that* $((u'_z, P_{u'_z,s'}), V_{u'_z,s'})$ *reduces* $((u_z, P_{u_z,s}), V_{u_z,s})$, *or* $((u'_z, P_{u'_z,s'}), V_{u'_z,s'}) \sqsubseteq ((u_z, P_{u_z,s}), V_{u_z,s})$ *for short, if*

- $u'_z \sqsubseteq u_z$,
- $P_{u_z,s} = P_{u'_z,s'} + P_{u_z-u'_z,s-s'}$,
- *for every* $(v_z, Q_{v_z,t}) \in V_{u_z,s}$ *there exists* $(v'_z, Q_{v'_z,t'}) \in V_{u'_z,s'}$ *with*
  - $v'_z \sqsubseteq v_z$, *and*
  - $Q_{v_z,t} = Q_{v'_z,t'} + Q_{v_z-v'_z,t-t'}$,
- $u'_z \neq 0$ *or there exist* $(v_z, Q_{v_z,t}) \in V_{u_z,s}$ *and* $(v'_z, Q_{v'_z,t'}) \in V_{u'_z,s'}$ *with*
  - $0 \neq v'_z \sqsubseteq v_z$, *and*
  - $Q_{v_z,t} = Q_{v'_z,t'} + Q_{v_z-v'_z,t-t'}$.

The last condition is again to avoid the situation that u'_z and all occurring v'_z are zero, which forms a trivial zero reduction.

Definition 5.3.7 *In case of reducibility,* $((u_z, P_{u_z,s}), V_{u_z,s})$ *is reduced to*

$$((u_z, P_{u_z,s}), V_{u_z,s}) \ominus ((u'_z, P_{u'_z,s'}), V_{u'_z,s'}) := ((u_z - u'_z, P_{u_z-u'_z,s-s'}), V_{u_z,s} \ominus V_{u'_z,s'}).$$

Herein, $V_{u_z,s} \ominus V_{u'_z,s'} := \{(v_z - v'_z, Q_{v_z-v'_z,t-t'}) : (v_z, Q_{v_z,t}) \in V_{u_z,s}, (v'_z, Q_{v'_z,t'}) \in V_{u'_z,s'}\}$, *where the* v'_z *and* $Q_{v'_z,t'}$ *fulfill the conditions from Definition 5.3.6.*

Again, it suffices to take only one difference $(v_z - v'_z, Q_{v_z - v'_z, t-t'})$ for every pair $(v_z, Q_{v_z,t}) \in V_{u_z,s}$ in the definition of $V_{u_z,s} \ominus V_{u'_z,s'}$ above.

Proposition 5.3.8 *If the input set, the procedure* normalForm, $\oplus$, *and* $\ominus$ *are defined as above, then the Completion Algorithm 5.3.2 terminates and computes a set* G *such that* $\mathrm{CS}_m(\mathcal{H}_\infty) \subseteq \{(s,t) :\in G, (v_z, Q_{v_z,t}) \in V_{u_z,s}\}$, *where* $(s,t) = (-W_1 v_z, T_3 u_z)$.

Proof. First let us show termination. Assume that the algorithm does not terminate and that it generates an infinite sequence

$$\{((u_{1,z}, P_{u_{1,z},s_1}), V_{u_{1,z},s_1}), ((u_{2,z}, P_{u_{2,z},s_2}), V_{u_{2,z},s_2}), \ldots\}$$

such that $((u_{i,z}, P_{u_{i,z},s_i}), V_{u_{i,z},s_i}) \not\sqsubseteq ((u_{j,z}, P_{u_{j,z},s_j}), V_{u_{j,z},s_j})$ whenever $i < j$.

To show termination we will associate with each pair $((u_{i,z}, P_{u_{i,z},s_i}), V_{u_{i,z},s_i})$ a pair (U_i, V_{U_i}), where U_i is an integer vector and V_{U_i} is a set of integer vectors of the same dimension. Thus, (U_i, V_{U_i}) is of a structure as used in the computation of $\mathcal{H}_\infty$ for two-stage stochastic integer programs. We will show that $(U_i, V_{U_i}) \not\sqsubseteq (U_j, V_{U_j})$ whenever $i < j$. Application of Lemma 3.2.10 then shows finiteness of the sequence $\{(U_1, V_{U_1}), (U_2, V_{U_2}), \ldots\}$ and thus, the above algorithm has to terminate.

As in the proof of Proposition 2.4.12, we may assume that T_2 and W_4, respectively, have full row rank. Let $B_1, \ldots, B_M$ denote all invertible square submatrices of T_2 of maximal row rank, and let $B'_1, \ldots, B'_{M'}$ denote all invertible square submatrices of W_4 of maximal row rank. Then associate with $((u_{i,z}, P_{u_{i,z},s_i}), V_{u_{i,z},s_i})$ the pair (U_i, V_{U_i}) where

$$U_i := (u_{i,z}, \det(B_1) B_1^{-1}(s_i - T_1 u_{i,z}), \ldots, \det(B_M) B_M^{-1}(s_i - T_1 u_{i,z})) \in \mathbb{Z}^{m_z + M l_s}$$

and

$$V_{U_i} := \{f(v_z) : (v_z, Q_{v_z,t_i}) \in V_{u_{i,z},s_i}\} \subseteq \mathbb{Z}^{n_z + M' l_t}$$

where

$$f(v_z) := (v_z, \det(B'_1)(B'_1)^{-1}(-t_i - W_3 v_z), \ldots, \det(B'_{M'})(B'_{M'})^{-1}(-t_i - W_3 v_z)).$$

By Lemma 2.4.9 we have that $P_{u_{j,z},s_j} \neq P_{u_{i,z},s_i} + P_{u_{j,z}-u_{i,z},s_j - s_i}$ implies $U_i \not\sqsubseteq U_j$, and that $Q_{v_{j,z},t_j} = Q_{v_{i,z},t_i} + Q_{v_{j,z}-v_{i,z},t_j - t_i}$ implies $f(v_{i,z}) \not\sqsubseteq f(v_{j,z})$. Therefore, the relation $((u_{i,z}, P_{u_{i,z},s_i}), V_{u_{i,z},s_i}) \not\sqsubseteq ((u_{j,z}, P_{u_{j,z},s_j}), V_{u_{j,z},s_j})$ implies that we either have $u_{i,z} \not\sqsubseteq u_{j,z}$, $U_i \not\sqsubseteq U_j$, or that there is some pair $(v_{j,z}, Q_{v_{j,z},t_j}) \in V_{u_{j,z},s_j}$ such that there is no pair $(v_{i,z}, Q_{v_{i,z},t_i}) \in V_{u_{i,z},s_i}$ with $v_{i,z} \not\sqsubseteq v_{j,z}$ or $f(v_{i,z}) \not\sqsubseteq f(v_{j,z})$. Altogether, $((u_{i,z}, P_{u_{i,z},s_i}), V_{u_{i,z},s_i}) \not\sqsubseteq ((u_{j,z}, P_{u_{j,z},s_j}), V_{u_{j,z},s_j})$ implies that $(U_i, V_{U_i}) \not\sqsubseteq (U_j, V_{U_j})$.

Application of Lemma 3.2.10 to the sequence $\{(U_1, V_{U_1}), (U_2, V_{U_2}), \ldots\}$ shows that this sequence has to be finite and thus, the above algorithm has to terminate.

To show correctness, we have to prove that G contains all connection vectors of $\mathcal{G}_N$ for arbitrary $N \in \mathbb{Z}_+$. Thus, it suffices for fixed N to prove that

$$\begin{aligned}\bar{G} := \{(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N) : \quad & ((\bar{u}_z, P_{\bar{u}_z,\bar{s}}), V_{\bar{u}_z,\bar{s}}) \in G, (\bar{u}_q^+, \bar{u}_q^-) \in P_{\bar{u}_z,\bar{s}}, \\ & (\bar{v}_{i,z}, Q_{\bar{v}_{i,z},\bar{t}}) \in V_{\bar{u}_z,\bar{s}}, (\bar{v}_{i,q}^+, \bar{v}_{i,q}^-) \in Q_{\bar{v}_{i,z},\bar{t}}, i = 1, \ldots, N\}\end{aligned}$$

has the positive sum property with respect to $\ker(A_N)$. To this end, we choose an arbitrary vector $(u, v_1, \ldots, v_N) \in \ker(A_N)$ and construct a finite representation

$$(u, v_1, \ldots, v_N) = \sum_{k,((\bar{u}_z,P_{\bar{u}_z,\bar{s}}),V_{\bar{u}_z,\bar{s}})\in G} \alpha_{\bar{u},\bar{s},k}(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N),$$

where $\alpha_{\bar{u},\bar{s},k} \in \mathbb{Z}_+$, $((\bar{u}_z, P_{\bar{u}_z,\bar{s}}), V_{\bar{u}_z,\bar{s}}) \in G$, $(\bar{u}_q^+, \bar{u}_q^-) \in P_{\bar{u}_z,\bar{s}}$, $(\bar{v}_{i,z}, Q_{\bar{v}_{i,z},\bar{t}}) \in V_{\bar{u}_z,\bar{s}}$, $(\bar{v}_{i,q}^+, \bar{v}_{i,q}^-) \in Q_{\bar{v}_{i,z},\bar{t}}$, $i = 1, \ldots, N$, and $(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N) \sqsubseteq (u, v_1, \ldots, v_N)$ whenever $\alpha_{\bar{u},\bar{s},k} > 0$. We need the additional subscript k since several vectors constructed from the same pair $((\bar{u}_z, P_{\bar{u}_z,\bar{s}}), V_{\bar{u}_z,\bar{s}}) \in G$ may appear in the above sum.

Since G contains F, and thus $\bar{G} \supseteq F_N$ for all $N \in \mathbb{Z}_+$, at least an integer linear combination

$$(u, v_1, \ldots, v_N) = \sum_{k,((\bar{u}_z,P_{\bar{u}_z,\bar{s}}),V_{\bar{u}_z,\bar{s}})\in G} \alpha_{\bar{u},\bar{s},k}(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N)$$

with $\alpha_{\bar{u},\bar{s},k} \in \mathbb{Z}_+$, $((\bar{u}_z, P_{\bar{u}_z,\bar{s}}), V_{\bar{u}_z,\bar{s}}) \in G$, $(\bar{u}_q^+, \bar{u}_q^-) \in P_{\bar{u}_z,\bar{s}}$, $(\bar{v}_{i,z}, Q_{\bar{v}_{i,z},\bar{t}}) \in V_{\bar{u}_z,\bar{s}}$, and $(\bar{v}_{i,q}^+, \bar{v}_{i,q}^-) \in Q_{\bar{v}_{i,z},\bar{t}}$, $i = 1, \ldots, N$, is possible.

For each such integer linear combination consider the value $\sum \|\alpha_{\bar{u},\bar{s},k}(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N)\|_1$, which is always non-negative. Thus, this sum is bounded from below by some non-negative infimum. Let us assume for the moment that there is a linear integer combination $\sum \alpha_{\bar{u},\bar{s},k}(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N)$ that attains this infimum.

If $\sum \|\alpha_{\bar{u},\bar{s},k}(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N)\|_1 = \|(u, v_1, \ldots, v_N)\|_1$ then all vectors $(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N)$ with $\alpha_{\bar{u},\bar{s},k} > 0$ lie in the same orthant of $\mathbb{R}^{d_N}$ as the vector $(u, v_1, \ldots, v_N)$. Therefore, $(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N) \sqsubseteq (u, v_1, \ldots, v_N)$ whenever $\alpha_{\bar{u},\bar{s},k} > 0$. But since $(u, v_1, \ldots, v_N)$ was chosen arbitrarily, this implies that $\bar{G}$ has the positive sum property with respect to $\ker(A_N)$. The claim then follows.

Assume on the contrary that it holds $\sum \|\alpha_{\bar{u},\bar{s},k}(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N)\|_1 > \|(u, v_1, \ldots, v_N)\|_1$ for such a minimal combination. Therefore, there have to exist $(\bar{u}_1, \bar{v}_{1,1}, \ldots, \bar{v}_{1,N})$ and $(\bar{u}_2, \bar{v}_{2,1}, \ldots, \bar{v}_{2,N})$ with $\alpha_{\bar{u}_1,\bar{s}_1,k_1} > 0$ and $\alpha_{\bar{u}_2,\bar{s}_2,k_2} > 0$, and a component p such that

$$(\bar{u}_1, \bar{v}_{1,1}, \ldots, \bar{v}_{1,N})^{(p)}(\bar{u}_2, \bar{v}_{2,1}, \ldots, \bar{v}_{2,N})^{(p)} < 0.$$

Thus,

$$\|(\bar{u}_1+\bar{u}_2, \bar{v}_{1,1}+\bar{v}_{2,1}, \ldots, \bar{v}_{1,N}+\bar{v}_{2,N})\|_1 < \|(\bar{u}_1, \bar{v}_{1,1}, \ldots, \bar{v}_{1,N})\|_1 + \|(\bar{u}_2, \bar{v}_{2,1}, \ldots, \bar{v}_{2,N})\|_1.$$

During the run of the algorithm, $((\bar{u}_1+\bar{u}_2, P_{\bar{u}_1+\bar{u}_2,\bar{s}_1+\bar{s}_2}), V_{\bar{u}_1,\bar{s}_1} \oplus V_{\bar{u}_2,\bar{s}_2})$ was reduced to $((0,P_{0,0}),\{(0,Q_{0,0})\})$ by elements $((u_1,P_{u_1,s_1}),V_{u_1,s_1})$, …, $((u_r,P_{u_r,s_r}),V_{u_r,s_r}) \in G$. Analogously to the construction in the proof of Proposition 3.3.7, this reduction to $((0,P_{0,0}),\{(0,Q_{0,0})\})$ yields a representation

$$(\bar{u}_1+\bar{u}_2, \bar{v}_{1,1}+\bar{v}_{2,1}, \ldots, \bar{v}_{1,N}+\bar{v}_{2,N}) = \sum_{i=1}^{r} (u_i, v_{i,1}, \ldots, v_{i,N})$$

where $u_{i,z} \sqsubseteq \bar{u}_{1,z}+\bar{u}_{2,z}$, $v_{i,j,z} \sqsubseteq \bar{v}_{1,j,z}+\bar{v}_{2,j,z}$,

$$\begin{aligned} P_{\bar{u}_1+\bar{u}_2,\bar{s}_1+\bar{s}_2} &= P_{0,0}+P_{u_1,s_1}+\ldots+P_{u_r,s_r}, \\ Q_{\bar{v}_{1,j}+\bar{v}_{2,k},\bar{t}_1+\bar{t}_2} &= Q_{0,0}+Q_{v_{1,1,z},t_1}+\ldots+Q_{v_{1,r,z},t_r}, \end{aligned}$$

$(u^+_{i,q}, u^-_{i,q}) \in P_{u_{i,z},s_i}$, $(v^+_{i,j,q}, v^-_{i,j,q}) \in Q_{v_{1,i,z},t_i}$, $i = 1,\ldots r$, $j = 1,\ldots,N$.

Moreover

$$\begin{aligned} P_{\bar{u}_1+\bar{u}_2,\bar{s}_1+\bar{s}_2} &= P_{0,0}+P_{u_1,s_1}+\ldots+P_{u_r,s_r}, \\ Q_{\bar{v}_{1,j}+\bar{v}_{2,k},\bar{t}_1+\bar{t}_2} &= Q_{0,0}+Q_{v_{1,1,z},t_1}+\ldots+Q_{v_{1,r,z},t_r}, \end{aligned}$$

imply that we can choose the continuous parts of the u_i and $v_{i,j}$ in such a way that

$$\begin{aligned} (\bar{u}_{1,q}+\bar{u}_{2,q})^+ &= \sum_{i=0}^{r} u^+_{i,q}, \\ (\bar{u}_{1,q}+\bar{u}_{2,q})^- &= \sum_{i=0}^{r} u^-_{i,q}, \\ (\bar{v}_{1,j,q}+\bar{v}_{2,j,q})^+ &= \sum_{i=0}^{r} v^+_{i,j,q}, \\ (\bar{v}_{1,j,q}+\bar{v}_{2,j,q})^- &= \sum_{i=0}^{r} v^-_{i,j,q}, \end{aligned}$$

for $j = 1,\ldots,N$. Thus, $u_{i,q} \sqsubseteq \bar{u}_{1,q}+\bar{u}_{2,q}$ and $v_{i,j,q} \sqsubseteq \bar{v}_{1,j,q}+\bar{v}_{2,j,q}$, $i = 0,\ldots r$, $j = 1,\ldots,N$, that is,

$$(u_i, v_{i,1}, \ldots, v_{i,N}) \sqsubseteq (\bar{u}_1+\bar{u}_2, \bar{v}_{1,1}+\bar{v}_{2,1}, \ldots, \bar{v}_{1,N}+\bar{v}_{2,N}),$$

for $i = 0, \ldots r$. Therefore,

$$\begin{aligned}\sum_{i=0}^{r} \|(u_i, v_{i,1}, \ldots, v_{i,N})\|_1 &= \|(\bar{u}_1 + \bar{u}_2, \bar{v}_{1,1} + \bar{v}_{2,1}, \ldots, \bar{v}_{1,N} + \bar{v}_{2,N})\|_1 \\ &< \|(\bar{u}_1, \bar{v}_{1,1}, \ldots, \bar{v}_{1,N})\|_1 + \|(\bar{u}_2, \bar{v}_{2,1}, \ldots, \bar{v}_{2,N})\|_1.\end{aligned}$$

Now rewrite

$$(u, v_1, \ldots, v_N) = \sum_{k,((\bar{u}_z, P_{\bar{u}_z,\bar{s}}), V_{\bar{u}_z,\bar{s}}) \in G} \alpha_{\bar{u},\bar{s},k}(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N)$$

as

$$\begin{aligned}(u, v_1, \ldots, v_N) = &\sum_{\substack{(\bar{u},\bar{s},k) \neq (\bar{u}_1,\bar{s}_1,k_1),\\ (\bar{u},\bar{s},k) \neq (\bar{u}_2,\bar{s}_2,k_2)}} \alpha_{\bar{u},\bar{s},k}(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N) + \\ &(\alpha_{\bar{u}_1,\bar{s}_1,k_1} - 1)(\bar{u}_1, \bar{v}_{1,1}, \ldots, \bar{v}_{1,N}) + \\ &(\alpha_{\bar{u}_2,\bar{s}_2,k_2} - 1)(\bar{u}_2, \bar{v}_{2,1}, \ldots, \bar{v}_{2,N}) + \\ &\sum_{i=0}^{r} (u_i, v_{i,1}, \ldots, v_{i,N}),\end{aligned}$$

which is an integer linear combination that contradicts the minimality assumption on the integer linear combination

$$\sum_{k,((\bar{u}_z, P_{\bar{u}_z,\bar{s}}), V_{\bar{u}_z,\bar{s}}) \in G} \alpha_{\bar{u},\bar{s},k}(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N).$$

Thus, it remains to show that the infimum of $\sum \alpha_{\bar{u},\bar{s},k} \|(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N)\|_1$ is indeed attained by some linear integer combination. Using an analogous argument as in the proof of Proposition 2.4.12, this can be seen as follows.

First, since there is at least one integer linear combination

$$(u, v_1, \ldots, v_N) = \sum_{k,((\bar{u}_z, P_{\bar{u}_z,\bar{s}}), V_{\bar{u}_z,\bar{s}}) \in G} \alpha_{\bar{u},\bar{s},k}(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N),$$

we have an upper bound K for the infimum. Thus, there are at most finitely many choices for the non-negative integers $\alpha_{\bar{u},\bar{s},k}$ and the (computed) building blocks $\bar{u}_z$, $\bar{v}_{j,z}$, $j = 1, \ldots, N$, such that $\sum \alpha_{\bar{u},\bar{s},k} \|(\bar{u}_z, \bar{v}_{1,z}, \ldots, \bar{v}_{N,z})\|_1 \leq K$. It remains to show that for fixed integers $\alpha_{\bar{u},\bar{s},k}$ and for fixed building blocks $\bar{u}_z$, $\bar{v}_{j,z}$, $j = 1, \ldots, N$, the infimum of $\sum \alpha_{\bar{u},\bar{s},k} \|(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N)\|_1$ is indeed attained for some choice of the

vectors $\bar{u}_q$, $\bar{v}_{j,q}$, $j = 1, \ldots, N$. Then, also the global infimum as the least value of the finitely many minimal values of $\sum \alpha_{\bar{u},\bar{s},k} \|(\bar{u}, \bar{v}_1, \ldots, \bar{v}_N)\|_1$, where $\alpha_{\bar{u},\bar{s},k}$ and $\bar{u}_z$, $\bar{v}_{j,z}$, $j = 1, \ldots, N$, are kept fixed, is attained by some combination.

But as in the proof of Proposition 2.4.12, fixing the integers $\bar{u}, \bar{s}, k$ and the building blocks $\bar{u}_z$, $\bar{v}_{j,z}$, $j = 1, \ldots, N$, leads to a linear minimization problem with non-empty feasible region, whose objective is bounded from below by 0. Thus, the minimum is attained for some choice of the vectors $\bar{u}_q$, $\bar{v}_{j,q}$, $j = 1, \ldots, N$. This concludes the proof. □

Termination and correctness of the above algorithm imply finiteness of $\mathrm{CS}_m(\mathcal{H}_\infty)$ if there is no row of $(T|W)$ that couples a first-stage continuous variable with a second-stage continuous variable. Thus, Theorem 5.3.1 is finally proved.

Note that in case of $T_1 = 0$, $T_2 = 0$, $W_1 = 0$, and $W_3 = 0$, we have a two-stage stochastic mixed-integer program with pure integer first-stage and pure continuous second-stage. The above algorithm then coincides with the algorithm to compute MIP Graver test sets as presented in Section 2.4. Moreover, in case of $T_2 = 0$ and $W_4 = 0$, the above algorithm simplifies to the algorithm to compute the set $\mathcal{H}_\infty$ for two-stage stochastic integer programs.

5.4 Conclusions

In this chapter we have introduced the new concepts of connection vectors and of connection sets. The knowledge of a connection set is already enough to reduce the problem of finding an improving direction for a given non-optimal feasible solution to the solution of finitely many smaller optimization problems. These smaller problems can be considered easy to solve compared to the full problem at hand. Moreover, there are only two classes of subproblems and the problems in each class differ only in their right-hand sides (and in their cost function vectors), which makes further application of test sets for their solution advisable. In this respect, we may even think of further decomposing the smaller matrices A_1 and A_2, and T and W, respectively, into smaller parts. This corresponds to a decomposition of building blocks into still smaller building blocks that are related via certain connection vectors.

When computing connection sets, we compute left-hand (or first-stage) and right-hand side (or second-stage) building blocks as well, although we do not need them for the construction of improving vectors. Moreover, there may be too many building blocks compared to the number of connection vectors. In order to speed up computa-

tions of connection sets, fast algorithmic solutions to the problems presented in the next chapter are needed.

Chapter 6

Some Open Problems

6.1 Algorithmic Improvements

In Chapters 3, 4, and 5 we have presented algorithms to compute $\mathcal{H}_\infty$ and to compute connection sets. All algorithms included the computation of left-hand side (or first-stage) and of right-hand side (or second-stage) building blocks. However, as we have seen in Chapter 5, these building blocks are not needed for the construction of improving vectors. Since there may be far more building blocks than connection vectors, we should aim at finding an algorithm that computes the connection set without computing the building blocks. Fast algorithmic solutions to the following two problems would speed up the computation of connection sets tremendously. A fast algorithmic solution to Problem 6.1.1 which is the same as Problem 2.4.6 on page 41, is also needed for the computation of $\mathcal{G}_{\mathbb{Z}}(A)$, or equivalently, of connection sets of MIP Graver test sets.

Problem 6.1.1 *Let $A \in \mathbb{Z}^{l\times d}$. For any $b \in \mathbb{R}^l$ define $P_{A,b} := \{z : Az = b, z \in \mathbb{R}^d_+\}$. For given $b_1, b_2 \in \mathbb{R}^l$ decide, whether $P_{A,b_1+b_2} = P_{A,b_1} + P_{A,b_2}$, where $P_{A,b_1} + P_{A,b_2}$ denotes the Minkowski sum of P_{A,b_1} and P_{A,b_2}.*

Problem 6.1.2 *Let $A \in \mathbb{Z}^{l\times d}$. For any $b \in \mathbb{R}^l$ define $P^I_{A,b} := \{z : Az = b, z \in \mathbb{Z}^d_+\}$. For given $b_1, b_2 \in \mathbb{R}^l$ decide, whether $P^I_{A,b_1+b_2} = P^I_{A,b_1} + P^I_{A,b_2}$, where $P^I_{A,b_1} + P^I_{A,b_2}$ again denotes the Minkowski sum of P^I_{A,b_1} and P^I_{A,b_2}.*

6.2 Extension of Maclagan's Theorem

In two- and multi-stage stochastic integer linear programming we have constructed a set $\mathcal{H}_\infty$ of building blocks that has very nice properties. Moreover, we gave an algorithm that, upon termination, returns this set. Unfortunately, in the integer situation, we can prove termination of this algorithm, and thus finiteness of $\mathcal{H}_\infty$, only for two-stage stochastic programs. For the multi-stage integer situation a successive generalization of Maclagan's Theorem would be very useful.

Definition 6.2.1 *Let $\mathcal{P}(S)$ denote the power set of a set S, and let $\leq_1$ denote the partial order $\leq$ on $S_1 := \mathbb{Z}_+^d$. Then S_{k+1} and $\leq_{k+1}$ are inductively defined as follows. Let $S_{k+1} = \mathcal{P}(S_k)$ and for $S, T \in S_{k+1}$ define $S \leq_{k+1} T$ if and only if for each $t \in T$ there is some $s \in S$ with $s \leq_k t$.*

With this definition, the Gordan-Dickson Lemma and Maclagan's Theorem can be restated as follows.

Lemma 6.2.2 *(Gordan-Dickson)*

Let $\{p_1, p_2, \ldots\}$ be a sequence of elements in S_1 such that $p_i \not\leq_1 p_j$ whenever $i < j$. Then this sequence is finite.

Lemma 6.2.3 *(Maclagan)*

Let $\{p_1, p_2, \ldots\}$ be a sequence of elements in S_2 such that $p_i \not\leq_2 p_j$ whenever $i < j$. Then this sequence is finite.

The following conjecture is a generalization of Maclagan's Theorem.

Conjecture 6.2.4 *For fixed $k \in \mathbb{Z}$, $k > 2$, let $\{p_1, p_2, \ldots\}$ be a sequence of elements in S_k such that $p_i \not\leq_k p_j$ whenever $i < j$. Then this sequence is finite.*

If this conjecture were true, then we could prove also in the multi-stage integer situation that $\mathcal{H}_\infty$ is finite and that the algorithm to compute this set terminates. Note that already a decision for the next case, $k = 3$, would be interesting. Clearly, an inductive argument for the general case would be best.

Chapter 7

Implementational Details

Some of the algorithms presented in the previous chapters have been implemented as a part of this thesis work. We called the program MLP in order to emphasize that MLP computes sets of minimal lattice points. MLP is written in the programming language C and is available from `http://www.testsets.de`.

7.1 The Program MLP

For given integer matrices A, T, and W, the program MLP allows the user to compute the following sets:

- the (truncated) IP Graver basis $\mathcal{G}_{\mathrm{IP}}(A)$,
- all $\sqsubseteq$-minimal elements in $\Lambda \setminus \{0\}$ for some integer lattice $\Lambda \subseteq \mathbb{Z}^d$,
- the minimal Hilbert basis of the pointed rational cone $\ker_{\mathbb{R}^d}(A) \cap \mathbb{R}^d_+$,
- the set $\mathcal{H}_\infty$ for the IP case for given T and W, where $(A|0)$ is assumed to be included into $(T|W)$.

7.1.1 Invocation

The program MLP is invoked as follows:

```
mlp option ... option FILENAME
```

Herein, the input file specified by

`FILENAME`

contains the matrices A, T, and W, respectively, together with specifications for the problem sizes. The used options specify the set to be computed.

7.1.2 Options

The following options are implemented:

- "gra" Computes the IP Graver basis $\mathcal{G}_{\mathrm{IP}}(A)$.
- "hil" Computes the unique minimal Hilbert basis of $\ker_{\mathbb{R}^d}(A) \cap \mathbb{R}^d_+$.
- "sip" Computes the set $\mathcal{H}_\infty$ for two-stage stochastic integer programs.
- "tru" Switches to truncated Graver/Hilbert basis computation.
- "gen" Input file contains a list of generators of $\ker_{\mathbb{Z}^d}(A)$.
- "mat" Input is a matrix.
- "tra" Input matrix is $A^\intercal$ instead of the problem matrix A.
- "per" Read the input and, depending on the form of the input, permute either
  - the components of the generators, or
  - the columns of the input matrix, or
  - the rows of the input matrix, which is given in transposed form.

  The output, however, will be given in the original ordering!
- "sla" This option allows a much faster Graver basis computation if either
  - the input matrix is of the form $(I|A)$, or
  - the set of generators is of the form $\{(e_1, -Ae_1), ..., (e_d, -Ae_d)\}$.

  (Each condition has to be true only up to sign of the unit vectors. The order of the generators is not important for the second condition, whereas it is crucial for the first condition that the the rows (columns) are ordered as specified. In that case, MLP will automatically produce a generating set as specified in the second condition.)

One may use longer names for all options, since only 3 characters are significant. If no option is chosen, then the default setting

```
mlp gra mat FILENAME
```

is invoked, that is, the IP Graver basis of the matrix given in FILENAME is computed. Note that all $\sqsubseteq$-minimal elements in $\Lambda \setminus \{0\}$ for some integer lattice $\Lambda \subseteq \mathbb{Z}^d$, are computed if

```
mlp gra gen FILENAME
```

is invoked. To this end, the input file has to contain a generating set of Λ over $\mathbb{Z}$.

When the option "sla", applicable only in special cases, is used, then a different implementation of the IP Graver basis algorithm is used, where the special structure of the generating set allows a tremendous speed-up, since much fewer S-vectors have to be considered. Moreover, the algorithm only checks reducibility and does not perform any reduction step, which leads to another speed-up. If an S-vector is reducible with respect to the current basis, then it can be guaranteed that it will reduce to 0 with respect to the current basis. Thus, the reduction need not be executed, since the result, 0, is already known beforehand. In addition to these two improvements, the set of those S-vectors that still have to be considered for reduction, need not be stored in the computer memory. This allows us to solve bigger problems within the same given amount of RAM. Termination and correctness of this algorithm is proved in Subsection 7.3.1.

The option "per" allows the user to permute the columns of the problem matrix or the components of the generators in order to make the application of the option "sla" possible.

7.1.3 Input File

The input file has to contain the following data:

- an INTEGER which denotes the number of (first-stage) variables
- only if $\mathcal{H}_\infty$ shall be computed: an INTEGER which denotes the number of second-stage variables
- an INTEGER which denotes either

  - the number of vectors in the generating set or
  - the number of rows in the matrix (=number of columns if matrix is transposed)

- (number of variables) INTEGERS which define a permutation of $\{1, \ldots, d\}$ if option "per" is chosen
- (number of variables) INTEGERS which defines an upper bounds vector if option "tru" is chosen
- either
  - a matrix A or $(T|W)$ with (number of rows) * (number of variables) INTEGERS, or
  - a generating set with (number of vectors) * (number of variables) INTEGERS

Note that in order to compute the unique minimal Hilbert basis of $\ker_{\mathbb{R}^d}(A) \cap \mathbb{R}^d_+$, the input file has to contain either the matrix A (use option "mat") or the vectors (e_1, Ae_1), ..., (e_d, Ae_d) (use option "gen").

7.1.4 Output Files

Depending on the specific set that is computed, the following output files are generated:

- "FILENAME.gra" contains the Graver basis $\mathcal{G}_{\mathrm{IP}}(A)$.
- "FILENAME.gra2" contains the Graver basis $\mathcal{G}_{\mathrm{IP}}(A)$ without brackets together with the numbers of variables and of vectors.
- "FILENAME.hil" contains the unique minimal Hilbert basis of $\ker_{\mathbb{R}^d}(A) \cap \mathbb{R}^d_+$.
- "FILENAME.bin" contains the Graver basis $\mathcal{G}_{\mathrm{IP}}(A)$ transformed into binomials of the form $x^{v^+} - x^{v^-}$ in the variables $x1, x2, \ldots, xd$.
- "FILENAME.sip" contains the set $\mathcal{H}_\infty$.
- "FILENAME.log" contains a copy of the standard output (screen).

7.2 Data Structure

The basic objects in the program MLP are sets of integer vectors of given dimension d. Each d-dimensional vector v is stored as an array of integers of length $d+4$, where the 4 additional integer entries are used to store the characteristic vectors of v^+, v^-, and the norms $\|v^+\|_1$, $\|v^-\|_1$. This additional data allows a quicker decision that a vector v is not reducible by another vector w, since $\|w^+\|_1 > \|v^+\|_1$ and $\|w^-\|_1 > \|v^-\|_1$ immediately imply $w \not\sqsubseteq v$.

Sets of vectors are internally stored both as a list of vectors and in a ternary tree structure. Each leaf of this ternary tree corresponds to exactly one of the 3^d possible sign patterns from $\{-, 0, +\}^d$ and contains those vectors from the set that have this particular sign pattern. Clearly, unused leaves or branches of the tree are fathomed, that is, the tree is constructed only as far as needed in order to save memory. This tree structure allows MLP to find reducing vectors faster, since many vectors in the current basis G are not considered as their sign pattern is not compatible with the vector that has to be reduced. Moreover, this tree structure allows a quicker construction of S-vectors if parts of the vectors have to lie in the same orthant as it is the case in the computation of truncated Graver bases (Lemma 2.2.2) or when the option "sla" is used.

7.3 Algorithmic Improvements

7.3.1 The Option "sla"

The option "sla" can only be used if a lattice generating set F of the special form $\{(e_1, -Ae_1), ..., (e_d, -Ae_d)\}$ for some integer matrix A is given. Clearly, such a generating set always exists for the lattice $\ker_{\mathbb{Z}^{d+l}}(A|E)$. Before we introduce and prove the algorithm behind the option "sla", let us consider the following example from [2] (example "altmann" on [22]).

Example. The IP Graver basis of

$$\begin{pmatrix} 1 & 0 & 0 & 2 & 0 & 0 & 2 & 1 & 1 & 3 & 2 & 2 & 2 & 3 & 3 & 3 \\ 0 & 1 & 0 & 0 & 2 & 0 & 1 & 2 & 1 & 2 & 3 & 2 & 3 & 2 & 3 & 3 \\ 0 & 0 & 1 & 0 & 0 & 2 & 1 & 1 & 2 & 2 & 2 & 3 & 3 & 3 & 2 & 3 \end{pmatrix}$$

consists of 73459 vectors and their negatives. If one computes this set via the invocation

```
mlp gra mat altmann
```

that is, without the option "sla", it takes 1297891 seconds, or 360.53 hours, whereas the invocation

```
mlp gra mat sla altmann
```

that uses the option "sla" via the algorithm presented below takes only 12787 seconds, or 3.55 hours. Times are given in CPU seconds on a SUN Enterprise 450, 300 MHz Ultra-SPARC. □

In the following we will present the algorithm behind the option "sla". To this end, let $\mathcal{G}_i := \{(v, -Av) \in \mathcal{G}_{\mathrm{IP}}(A|E) : \|v\|_1 = i\}$.

Clearly, $\mathcal{G}_0 = \{0\}$ and $\mathcal{G}_1 = F \cup -F$. The idea of the proposed algorithm is to compute $\mathcal{G}_2$, $\mathcal{G}_3, \ldots$ inductively one after another. This process corresponds to a selection strategy by which the next S-vector is chosen from the set C in the completion algorithm to compute IP Graver bases, see Section 1.3. This particular selection strategy, valid only if the given generating set of the lattice is of the above form, leads to a tremendous speed up, since fewer S-vectors have to be considered, no S-vector has to be reduced but only checked for reducibility, and the list C of S-vectors that are still to be considered need not be stored in memory. In addition to that, no vector is generated in the final output that does not belong to the desired Graver basis which may happen in the general Graver basis algorithm from Section 1.3.

Suppose we have computed $\mathcal{G}_1, \ldots, \mathcal{G}_k$, for some $k \geq 1$. Then the following observation allows us to avoid all the reduction steps.

Lemma 7.3.1 *The set $\mathcal{G}_{\leq k} := \mathcal{G}_1 \cup \ldots \cup \mathcal{G}_k$ has the positive sum property with respect to the set $\{(v, -Av) \in \ker_{\mathbb{Z}^{d+l}}(A|E) : \|v\|_1 \leq k\}$.*

Proof. Let $(z, -Az) \in \{(v, -Av) \in \ker_{\mathbb{Z}^{d+l}}(A|E) : \|v\|_1 \leq k\} \subseteq \ker_{\mathbb{Z}^{d+l}}(A|E)$. By the positive sum property of $\mathcal{G}_{\mathrm{IP}}(A|E)$ with respect to $\ker_{\mathbb{Z}^{d+l}}(A|E)$ we can construct a finite linear representation $z = \sum \alpha_i(g_i, -Ag_i)$ with $\alpha_i \in \mathbb{Z}_+$, $(g_i, -Ag_i) \in \mathcal{G}_{\mathrm{IP}}(A|E)$, and $(g_i, -Ag_i) \sqsubseteq (z, -Az)$ for all i. But these conditions imply $\|g_i\|_1 \leq \|z\|_1 \leq k$ for all i. Thus, $(g_i, -Ag_i) \in \mathcal{G}_{\leq k}$ as claimed. □

Corollary 7.3.2 *Let $(z, -Az) \in \ker_{\mathbb{Z}^{d+l}}(A|E)$ and $\|z\|_1 = k + 1$. Then there are precisely two possible situations:*

1. *There exists a vector $(v, -Av) \in \mathcal{G}_{\leq k}$ with $(v, -Av) \sqsubseteq (z, -Az)$. Then we know already that* $\mathrm{normalForm}((z, -Az), \mathcal{G}_{\leq k}) = 0$.

2. *There does not exist a vector* $(v, -Av) \in \mathcal{G}_{\leq k}$ *with* $(v, -Av) \sqsubseteq (z, -Az)$. *Then* $(z, -Az) \in \mathcal{G}_{k+1}$.

Proof. First, suppose that there exists $(v, -Av) \in \mathcal{G}_{\leq k}$ with $(v, -Av) \sqsubseteq (z, -Az)$. Since $\|z - v\|_1 \leq k$, we know by the positive sum property of $\mathcal{G}_{\leq k}$ that there are finitely many (not necessarily different) elements $(g_i, -Ag_i) \in \mathcal{G}_{\leq k}$ such that $(z-v, -A(z-v)) = \sum(g_i, -Ag_i)$ and $(g_i, -Ag_i) \sqsubseteq (z-v, -A(z-v)) \sqsubseteq (z, -Az)$ for all i. Choosing $(v, -Av)$ together with these $(g_i, -Ag_i)$ in the algorithm normalForm we obtain $\text{normalForm}((z, -Az), \mathcal{G}_{\leq k}) = 0$.

On the other hand, if there does not exist $(v, -Av) \in \mathcal{G}_{\leq k}$ with $(v, -Av) \sqsubseteq (z, -Az)$, then $(z, -Az) \in \mathcal{G}_{k+1}$, that is, the vector $(z, -Az)$ cannot be written as a sum $(v_1, -Av_1) + (v_2, -Av_2)$ with $(v_i, -Av_i) \in \ker_{\mathbb{Z}^{d+l}}(A|E) \setminus \{0\}$, $(v_i, -Av_i) \sqsubseteq (z, -Az)$, $i = 1, 2$. To see this, assume on the contrary that such vectors $(v_1, -Av_1)$ and $(v_2, -Av_2)$ exist. Therefore, we know that $\|v_i\|_1 \leq k$, $i = 1, 2$, since $(0, 0)$ is the only vector in $\ker_{\mathbb{Z}^{d+l}}(A|E)$ starting with d zero components. Therefore, by the positive sum property of $\mathcal{G}_{\leq k}$ with respect to $\{(v, -Av) \in \ker_{\mathbb{Z}^{d+l}}(A|E) : \|v\|_1 \leq k\}$, $(v_1, -Av_1)$ and $(v_2, -Av_2)$ can both be written as positive linear integer combinations of elements from $\mathcal{G}_{\leq k}$ that all lie in the same orthants as $(v_1, -Av_1)$ and $(v_2, -Av_2)$, respectively. Put together, both combinations allow a positive linear integer representation of $(z, -Az)$ by elements from $\mathcal{G}_{\leq k}$ that all lie in the same orthant as $(z, -Az)$. But all the summands $(v, -Av) \in \mathcal{G}_{\leq k}$ in this representation of $(z, -Az)$ fulfill $(v, -Av) \sqsubseteq (z, -Az)$ in contradiction to our initial assumption, that no such vector $(v, -Av)$ exists. Thus, $(z, -Az) \in \mathcal{G}_{k+1}$. □

Next we define the set of necessary S-vectors, that have to be checked whether they are reducible with respect to $\mathcal{G}_{\leq k}$, as follows:

$$\text{S-vectors}((v, -Av), (w, -Aw)) := \begin{cases} \{(v + w, -A(v + w))\} & \text{if v and w lie in the} \\ & \text{same orthant of } \mathbb{R}^d \\ & \text{and } \|v + w\|_1 = k + 1, \\ \emptyset & \text{otherwise.} \end{cases}$$

Now we can employ the following algorithm to find $\mathcal{G}_{k+1}$.

Algorithm 7.3.3 *(Algorithm to find $\mathcal{G}_{k+1}$)*

Input: $\mathcal{G}_{\leq k}$

Output: $\mathcal{G}_{k+1}$

$G := \emptyset$

$C := \bigcup_{f,g \in \mathcal{G}_{\leq k}} \text{S-vectors}(f,g)$

for each $s \in C$ *do*

  if normalForm$(s, \mathcal{G}_{\leq k}) \neq 0$ *then*

    $G := G \cup \{s, -s\}$

return G.

Note that the algorithm returns precisely the set $\mathcal{G}_{k+1}$ instead of only a possibly larger superset as it may happen with the general Graver basis algorithm presented in Section 1.3. Moreover, Corollary 7.3.2 allows a simple and fast algorithmic test whether normalForm$(s, \mathcal{G}_{\leq k}) \neq 0$.

Lemma 7.3.4 *With the above definitions of input set,* normalForm*, and* S-vectors*, Algorithm 7.3.3 terminates and returns $\mathcal{G}_{k+1}$.*

Proof. First, the algorithm terminates since only a finite number of vectors is checked for reducibility. By Corollary 7.3.2 every vector in G must belong to $\mathcal{G}_{k+1}$. Therefore, it remains to prove that every vector of $\mathcal{G}_{k+1}$ is indeed generated by the algorithm.

By G denote the set that is returned by Algorithm 7.3.3 and let $(z, -Az) \in \mathcal{G}_{k+1}$. Since $F \cup -F = \mathcal{G}_1 \subseteq \mathcal{G}_{\leq k}$, we can write $(z, -Az)$ as a positive linear integer combination $(z, -Az) = \sum \alpha_i (v_i, -Av_i)$ for some $\alpha_i \in \mathbb{Z}_{>0}$ and vectors $(v_i, -Av_i) \in \mathcal{G}_{\leq k} \cup G$ with $v_i \sqsubseteq z$ for all i. From the set of all such positive linear integer combinations $\sum \alpha_i (v_i, -Av_i)$ choose one such that $\sum \alpha_i \|Av_i\|_1$ is minimal. Note that we have $\sum \alpha_i \|Av_i\|_1 \geq \|Az\|_1$ with equality if and only if $Av_i \sqsubseteq Az$ for all i.

If $\sum \alpha_i \|Av_i\|_1 = \|Az\|_1$, the we get $(v_i, -Av_i) \sqsubseteq (z, -Az)$ for all i. Together with $(z, -Az) = \sum \alpha_i (v_i, -Av_i)$, $\alpha_i \in \mathbb{Z}_{>0}$, and $(z, -Az) \in \mathcal{G}_{k+1}$ this representation must be trivial, that is, $(z, -Az) = 1 \cdot (z, -Az) \in \mathcal{G}_{\leq k} \cup G$, and consequently, $(z, -Az) \in G$. Hence, we will assume on the contrary that $\sum \alpha_i \|Av_i\|_1 > \|Az\|_1$ holds.

Thus, there must exist $(v_{i_1}, -Av_{i_1}), (v_{i_2}, -Av_{i_2})$ such that $(Av_{i_1})^{(m_0)} (Av_{i_2})^{(m_0)} < 0$ for some component m_0. Now there are two possible cases.

If $\|v_{i_1} + v_{i_2}\|_1 = k + 1$, then we must have $(v_{i_1}, -Av_{i_1}) + (v_{i_2}, -Av_{i_2}) = (z, -Az)$, since $(0, 0)$ is the only vector $(v, -Av) \in \ker_{\mathbb{Z}^{l+d}}(A|E)$ with $v = 0$. Moreover, since

$(v_{i_1}, -Av_{i_1}) + (v_{i_2}, -Av_{i_2}) \in$ S-vectors$((v_{i_1}, -Av_{i_1}), (v_{i_2}, -Av_{i_2}))$, the vector $(z, -Az)$ was considered through the run of Algorithm 7.3.3 and thus, it was added to G, since $(z, -Az) \in \mathcal{G}_{k+1}$ cannot be reduced by $\mathcal{G}_{\leq k}$.

If, on the contrary, $\|v_{i_1} + v_{i_2}\|_1 \leq k$, then there is a positive linear integer combination $(v_{i_1}, -Av_{i_1}) + (v_{i_2}, -Av_{i_2}) = \sum \beta_j (v'_j, -Av'_j)$ for some positive integers β_j and some $(v'_j, -Av'_j) \in \mathcal{G}_{\leq k}$, by the positive sum property of $\mathcal{G}_{\leq k}$ with respect to the set $\{(v, -Av) \in \ker_{\mathbb{Z}^{d+l}}(A|E) : \|v\|_1 \leq k\}$ (Lemma 7.3.1).

Moreover, $\beta_j(v'_j, -Av'_j) \sqsubseteq (v_{i_1}, -Av_{i_1}) + (v_{i_2}, -Av_{i_2})$ for all j, implying that $v'_j \sqsubseteq z$ and that for each component m

$$\begin{aligned} \sum \beta_j \left|(Av'_j)^{(m)}\right| &= \left|\sum \beta_j (Av'_j)^{(m)}\right| \\ &= \left|(A(v_{i_1} + v_{i_2}))^{(m)}\right| \\ &\leq \left|(Av_{i_1})^{(m)}\right| + \left|(Av_{i_2})^{(m)}\right|, \end{aligned}$$

holds, where the last inequality is strict for $m = m_0$ by construction. Summation over m now gives

$$\sum \beta_j \|Av'_j\|_1 = \|A(v_{i_1} + v_{i_2})\|_1 < \|Av_{i_1}\|_1 + \|Av_{i_2}\|_1.$$

But then $(z, -Az) = \sum \beta_i (v'_i, -Av'_i) + (\alpha_{i_1} - 1)(v_{i_1}, -Av_{i_1}) + (\alpha_{i_2} - 1)(v_{i_2}, -Av_{i_2}) + \sum_{i \neq i_1, i_2} \alpha_i (v_i, -Av_i)$ contradicts the minimality of $\sum \alpha_i \|Av_i\|_1$. We conclude that already $\sum \alpha_i \|Av_i\|_1 = \|Az\|_1$ and the claim follows. □

It remains to find some k such that $\mathcal{G}_{\text{IP}}(A|E) = \mathcal{G}_{\leq k}$.

Lemma 7.3.5 *Let $k \in \mathbb{Z}_{>0}$ satisfy $\mathcal{G}_{k+1} = \ldots = \mathcal{G}_{2k} = \emptyset$. Then $\mathcal{G}_{IP}(A|E) = \mathcal{G}_{\leq k}$.*

Proof. Since $F \subseteq \mathcal{G}_{\leq k}$, $\mathcal{G}_{\leq k}$ generates $\ker_{\mathbb{Z}^{l+d}}(A|E)$ over $\mathbb{Z}$. Moreover, $\mathcal{G}_{\leq k}$ has the positive sum property with respect to the set $\{(v, -Av) \in \ker_{\mathbb{Z}^{d+l}}(A|E) : \|v\|_1 \leq k\}$, Lemma 7.3.1. Since $\mathcal{G}_{k+1} = \ldots = \mathcal{G}_{2k} = \emptyset$, $\mathcal{G}_{\leq k}$ must have the positive sum property with respect to the set $\{(v, -Av) \in \ker_{\mathbb{Z}^{d+l}}(A|E) : \|v\|_1 \leq 2k\}$.

This implies that choosing $\mathcal{G}_{\leq k}$ as the input set to the completion algorithm that computes IP Graver bases, all S-vectors $(v+w, -A(v+w))$ satisfy $\|v+w\|_1 \leq 2k$ and thus, by the positive sum property of $\mathcal{G}_{\leq k}$, reduce to 0 in the algorithm normalForm. Therefore, $\mathcal{G}_{\leq k}$ is returned by the algorithm and hence, it contains $\mathcal{G}_{\text{IP}}(A|E)$. Since each element in $\mathcal{G}_{\leq k}$ belongs to $\mathcal{G}_{\text{IP}}(A|E)$, the claim is proved. □

The input set to the algorithm that computes all $\sqsubseteq$-minimal solutions in the set $\{z \in \mathbb{Z}^d : Az = 0, l \leq z \leq u\} \setminus \{0\}$, where $l \leq 0 \leq u$, presented in Section

2.2, was of the form $\{\pm(e_1, -Ae_1), ..., \pm(e_d, -Ae_d)\}$. Therefore, the above algorithmic improvement also applies to this computation. It has been implemented in MLP, as well.

7.3.2 Computing the Minkowski Sum of Two Sets of Vectors

During the computation of $\mathcal{H}_\infty$ we often have to compute the Minkowski sums $V_u + V_{u'} = \{v + v' : v \in V_u, v' \in V_{u'}\}$ for two given sets V_u, $V_{u'}$ of vectors. The sets V_u and $V_{u'}$ are stored as lists and thus, when we form all possible combinations $v + v' : v \in V_u, v' \in V_{u'}$, many of these sums will usually occur several times, although they are needed only once in the definition of $V_u + V_{u'}$. Clearly, these multiple occurrences of sums should be avoided, and if possible, with a small number of comparisons of vectors.

To avoid these multiple occurrences of sums, MLP employs the following strategy. First, the vectors in both lists V_u and $V_{u'}$ are ordered with respect to increasing lexicographic ordering. The idea is to construct the vectors in $V_u + V_{u'}$ with respect to increasing lexicographic ordering, as well, starting with the smallest element. Let $V_u := \{v_1, \ldots, v_p\}$ and $V_{u'} := \{v'_1, \ldots, v'_q\}$ denote the ordered lists, and let $C := \{1, \ldots, p\} \times \{1, \ldots, q\}$ be the set of all those ordered pairs (i, j) of indices such that the sum $v_i + v'_j$ has not been considered so far.

Note that if $(i_1, j_1) \leq (i_2, j_2)$, then $v_{i_1} + v'_{j_1}$ is lexicographically smaller than (or at most equal to) $v_{i_2} + v'_{j_2}$. Thus, the index pair (i, j) of the next lexicographically smallest sum $v_i + v'_j$ that is to be added to $V_u \oplus V_{u'}$ must be minimal in C with respect to $\leq$ on $\mathbb{Z}^2_+$.

Thus, the lexicographically smallest element in $V_u + V_{u'}$ is $v_1 + v'_1$. The second smallest element is either $v_1 + v'_2$ or $v_2 + v'_1$, since $(1, 2)$ and $(2, 1)$ are the only two minimal pairs in $\{1, \ldots, p\} \times \{1, \ldots, q\} \setminus \{(1, 1)\}$. Therefore, we can employ the following algorithm in order to find $V_u + V_{u'}$.

Algorithm 7.3.6 *(Algorithm to find $V_u + V_{u'}$)*

Input: V_u, $V_{u'}$ *lexicographically ordered, smallest element first*

Output: $V_u + V_{u'}$ *lexicographically ordered, smallest element first*

$G := \emptyset$

$C := \{1, \ldots, p\} \times \{1, \ldots, q\}$

$t := 0$

while $C \neq \emptyset$ *do*

$M := \{v_i + v'_j : (i,j)$ *is a* $\leq$*-minimal pair in* $C\}$

Choose $s = v_i + v'_j$*, a lexicographically minimal vector in* M*.*

$C := C \setminus \{(i,j)\}$

if $s \neq t$ *then*

$G := G \cup \{s\}$

$t := s$

return G*.*

Lemma 7.3.7 *Algorithm 7.3.6 terminates and returns* $V_u + V_{u'}$ *lexicographically ordered, smallest element first.*

Proof. Termination of the algorithm is clear, since each of the pq sums is considered exactly once. Moreover, it is clear that the returned set G contains $V_u + V_{u'}$, since only those vectors s are not added to G that already occur in G as its currently last element t. Thus, it remains to show that G is lexicographically ordered, smallest element first.

When a sum $v_{i_0} + v'_{j_0}$ was chosen in the run of Algorithm 7.3.6, $v_{i_0} + v'_{j_0}$ was the lexicographically smallest element in the set $\{v_i + v'_j : (i,j) \in C\}$. To see this, remember that a lexicographically smallest element in the set $\{v_i + v'_j : (i,j) \in C\}$ must have a pair of indices that is $\leq$-minimal in C. Thus, each lexicographically smallest element in the set $\{v_i + v'_j : (i,j) \in C\}$ must belong to M. But $v_{i_0} + v'_{j_0}$ was a lexicographically smallest element in M and thus, $v_{i_0} + v'_{j_0}$ is lexicographically smaller than (or at most equal to) all elements in the set $\{v_i + v'_j : (i,j) \in C\}$. But this implies that G is lexicographically ordered, smallest element first. □

Note that since G is ordered, Algorithm 7.3.6 returns a list G that does not contain an element twice, as desired.

7.3.3 Properties and Structure of H_∞

Given a pair $(u, V_u) \in H_\infty$ in the two-stage stochastic integer case. What can we say about its structure?

Lemma 7.3.8 *(Structure of* V_0*)*

Let $(0, V_0) \in \mathcal{H}_\infty$*. Then* $\mathcal{G}_{IP}(W) \subseteq V_0 \subseteq \mathcal{G}_{IP}(W) \cup \{0\}$*.*

Proof. For arbitrary $N \in \mathbb{Z}_+$, any Graver basis element $z = (0, v_1, \ldots, v_N) \in \mathcal{G}_N$ contains exactly one non-zero building block v_i. If, on the contrary, v_i and v_j were both non-zero, then we could construct a non-zero vector $z' \in \ker_{\mathbb{Z}^{d_N}}(A_N)$ with $z' \sqsubseteq z$ and $z' \neq z$ by replacing v_i by 0 in z. This contradicts the $\sqsubseteq$-minimality of the Graver basis element z. Since there is only one non-zero building block v_i in z, this building block has to be $\sqsubseteq$-minimal in $\ker_{\mathbb{Z}^d}(W) \setminus \{0\}$, that is, v_i belongs to the Graver basis of W. □

Thus, we can immediately construct V_0 from the IP Graver basis of W.

Lemma 7.3.9 *(Structure of V_u)*

For any $(u, V_u) \in \mathcal{H}_\infty$, $u \neq 0$, the set V_u coincides with the set of all $\sqsubseteq$-minimal solutions v of $Wv = -Tu$, where a solution v is called $\sqsubseteq$-minimal if no other solution v' of $Wv = -Tu$ with $v' \sqsubseteq v$ exists.

Proof. For arbitrary N, any Graver basis element $z = (u, v_1, \ldots, v_N)$ can contain only $\sqsubseteq$-minimal solutions v of $Wv = -Tu$ as building blocks $v_i, i = 1, \ldots, N$. If v_j were not such a $\sqsubseteq$-minimal solution, then there exists some $v_j' \neq v_j$ with $Wv_j' = -Tu$ and $v_j' \sqsubseteq v_j$. Thus, replacing in z the building block v_j by v_j' we obtain a non-zero vector $z' \in \ker_{\mathbb{Z}^{d_N}}(A_N)$ satisfying $z' \sqsubseteq z$ and $z' \neq z$, which contradicts the $\sqsubseteq$-minimality of the Graver basis element z. The set $M(u)$ of all $\sqsubseteq$-minimal solutions of $Wv = -Tu$, however, is finite. To see this, apply the Gordan-Dickson-Lemma to the set $\{(v^+, v^-) : v \in M(u)\}$.

On the other hand, choosing N sufficiently large and forming $z = (u, v_1, \ldots, v_N)$ such that each of the finitely many $\sqsubseteq$-minimal solution of $Wv = -Tu$ appears as some building block v_i, then z has to belong to $\mathcal{G}_N$. Thus, $\mathcal{H}_\infty$ indeed contains all these $\sqsubseteq$-minimal solutions. The claim now follows. □

Thus, we can, for given u, immediately compute the final set V_u as it should appear in $\mathcal{H}_\infty$. Replacing the set V_u by the set of all $\sqsubseteq$-minimal solutions of $Wv = -Tu$ before adding (u, V_u) to G in the completion algorithm to compute $\mathcal{H}_\infty$ for two-stage stochastic integer programs, Section 3.3, tremendously speeds up the computation. No further pair (u, V_u) with some other V_u will be added to G.

7.3.4 Minimal Integer Solutions to $Az = b$

To compute all $\sqsubseteq$-minimal integer solutions of $Az = b$ we will employ the Completion Algorithm 1.3.1. To this end, we have to specify the input set, the reduction needed in the normalForm algorithm and the set of S-vectors.

As input set we choose $\mathcal{G}_{\mathrm{IP}}(A) \cup \{z_0\}$, where we may choose any integer solution z_0 of $Az = b$. Moreover, if $g \sqsubseteq s$ and $g \notin \mathcal{G}_{\mathrm{IP}}(A)$ then we say that $s \in \mathbb{Z}^d$ reduces by $g \in \mathbb{Z}^d$ to 0. Finally, we define

$$\text{S-vectors}(v, w) := \begin{cases} \{v + w\} & \text{if } v \in G \text{ and } w \in \mathcal{G}_{\mathrm{IP}}(A), \\ \emptyset & \text{otherwise.} \end{cases}$$

Lemma 7.3.10 *With the above definitions of input set,* normalForm*, and* S-vectors*, the Algorithm 1.3.1 terminates and returns a set containing all $\sqsubseteq$-minimal integer solutions of $Az = b$.*

The set of all minimal integer solutions to $Az = b$ is the set of all those elements z in G which are irreducible with respect to $G \setminus \{z\}$.

Proof. Termination follows by application of Gordan-Dickson Lemma to the sequence $\{(g^+, g^-) : g \in G \setminus \mathcal{G}_{\mathrm{IP}}(A)\}$.

To see that every minimal integer solution z_1 to $Az = b$ is contained in the returned set G note that $z_1 = z + \sum \alpha_i g_i$ for some $z \in G \setminus \mathcal{G}_{\mathrm{IP}}(A)$, some positive integers α_i, and some vectors $g_i \in G(A)$ with $g_i \sqsubseteq z_1 - z$, by the positive sum property of $\mathcal{G}_{\mathrm{IP}}(A)$. Now choose $z \in G \setminus \mathcal{G}_{\mathrm{IP}}(A)$ such that $\sum \alpha_i \|g_i\|_1$ is minimal. By the same rewriting technique as in the correctness proof of the integer version of Algorithm 1.3.1 one can prove that this sum must be zero, implying that $z_1 = z \in G \setminus \mathcal{G}_{IP}(A)$. □

Notation

$\mathbb{Z}$	integer numbers
$\mathbb{Q}$	rational numbers
$\mathbb{R}$	real numbers
$\mathbb{X}_+$	$\{x \in \mathbb{X} : x \geq 0\}$
$\mathbb{X}_{>0}$	$\{x \in \mathbb{X} : x > 0\}$
$\ker_{\mathbb{X}}(A)$	$\{x \in \mathbb{X} : Ax = 0\}$
$(\mathrm{P})_{c,b}$	$\min\{c^\intercal z : Az = b,\, z \in \mathbb{X}_+\}$
$(\mathrm{IP})_{c,b}$	$(\mathrm{P})_{c,b}$ where $\mathbb{X} = \mathbb{Z}^d$
$(\mathrm{LP})_{c,b}$	$(\mathrm{P})_{c,b}$ where $\mathbb{X} = \mathbb{R}^d$
$(\mathrm{MOD})_{c,b,\bar{b}}$	$\min\{c^\intercal z : Az = b, \bar{A}z \equiv \bar{b} \pmod{p}, z \in \mathbb{Z}^d_+\}$
$v^{(i)}$	i^{th} component of v
$\|v\|_1$	$l_1 - \text{norm } \sum_{i=1}^d \|v^{(i)}\|$
$\mathrm{supp}(v)$	support $\{i : v^{(i)} \neq 0\}$
$(v^+)^{(i)}$	$\max(v^{(i)}, 0)$
$(v^-)^{(i)}$	$\max(-v^{(i)}, 0)$
x^v	$x_1^{v^{(1)}} \cdot \ldots \cdot x_d^{v^{(d)}}$
$e_1, \ldots, e_d$	unit vectors in $\mathbb{R}^d$
$v \leq w,$ where $v, w \in \mathbb{R}^d_+$	$v^{(i)} \leq w^{(i)}$ for $i = 1, \ldots, d$
$v \sqsubseteq w,$ where $v, w \in \mathbb{R}^d$	$(v^+, v^-) \leq (w^+, w^-)$
$P_{A,b}$	$\{z : Az = b, z \in \mathbb{R}^d_+\}$
$P^I_{A,b}$	$\{z : Az = b, z \in \mathbb{Z}^d_+\}$

Conclusions

In this thesis we dealt with the solution of (mixed-integer) linear programs $(\mathrm{P})_{c,b}$ via a simple augmentation algorithm that relies on an oracle that provides improving directions to all non-optimal feasible solutions. Graver test sets provide such an oracle for arbitrary choices of the cost function c and of the right-hand side b. They depend only on the problem matrix and contain improving directions to all non-optimal feasible solutions to $(\mathrm{P})_{c,b}$. This enormous amount of stored information, however, leads to huge test sets already for small problems with less that 100 variables, say.

From a practical point of view, however, we are usually not interested in a test set. Instead we want to find improving directions to non-optimal solutions or we seek a proof that the current solution is optimal. Test sets help with this problem but they are too big. Therefore, we have constructed sets of building blocks and sets of connection vectors which are usually much smaller than the test set they correspond to. Nonetheless, an improving vector can still be efficiently constructed from the vectors in these smaller sets. Thus, larger problems can be solved via the Augmentation Algorithm 0.0.2, if the improving directions are provided via building blocks or via connection sets.

Connection sets and building blocks encode the full test set much more efficiently than just by listing its vectors. Since less information is computed off-line, before the specific problem data are known, more computational effort is moved to the online part of the solution of the optimization problem, when c, b, and a feasible solution z_0 are finally given.

It is our belief that more compact representations of the information stored in a test set will further drastically improve the performance of the Augmentation Algorithm 0.0.2, where the improving directions are reconstructed from these compact representations. To this end, one might think of further decomposition of building blocks into even smaller building blocks, other types of decomposition, or exploitation of the (still fairly unknown) structure of test sets.

In this thesis we have presented a novel decomposition approach in two-stage stochas-

tic (mixed-integer) linear programming, that to the best of our knowledge does not have a counter-part in the existing stochastic programming literature. For one part, this decomposition led to an interesting finiteness result (finiteness of $\mathcal{H}_\infty$). Moreover, using this approach, we could solve large, although academic, examples much faster than other existing (specialized) computer codes. This supports the above hope/belief that test set methods will eventually solve problems of practical sizes and interest. Most probably, however, they will display their full power when successfully combined with the existing algorithmic ideas from mixed-integer linear programming.

In Chapters 1 and 2 we laid the necessary notational and algorithmic basis of the later decomposition approach. We have presented the positive sum property inherent to Graver bases, a simple property that already implies the universal test set property. More importantly, however, this property pointed our attention to the notion of a completion procedure. This algorithmic pattern was then employed throughout this thesis not only to compute LP, IP, and MIP Graver test sets but also to compute building blocks and connection sets.

Computation of $\mathcal{H}_\infty$ and of connection sets would strongly benefit from fast algorithmic solutions to Problems 6.1.1 and 6.1.2, Chapter 6. From a theoretical and structural point of view, however, we think that the challenging combinatorial Problem 6.2.4 is even more interesting, since it would imply finiteness of $\mathcal{H}_\infty$ also for multi-stage stochastic integer linear programs.

Some of the algorithms presented in this thesis for the IP case were implemented in the computer program MLP. Computational experience showed that the presented algorithms work fine for problem matrices arising in combinatorial algebra. In integer programming, however, many improvements still have to be done to make a primal solution of practical problems via test set methods possible. We think that the decomposition approach presented in this thesis, in particular that for two-stage stochastic programs, already indicates that this practical applicability of test set methods may be achieved in future. In this respect, a combination of test set ideas with other algorithmic ideas from (mixed-) integer linear programming should be established and exploited.

Bibliography

[1] S. Ahmed, M. Tawarmalani, and N. V. Sahinides. A finite branch and bound algorithm for two-stage stochastic integer programs. Preprint, University of Illinois at Urbana-Champaign, 2000, downloadable from `http://archimedes.scs.uiuc.edu/group/publications.html`.

[2] K. Altmann. The chain property for the associated primes of A-graded ideals. Electronic preprint, Humboldt University Berlin, 2000, downloadable from `http://front.math.ucdavis.edu/math.AG/0004142`.

[3] J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer, New York, 1997.

[4] J. R. Birge and R. J.-B. Wets. Sublinear upper bounds for stochastic programs with recourse. *Mathematical Programming* **43** (1989), 131-149.

[5] A. M. Bigatti, R. LaScala, and L. Robbiano. Computing toric ideals. *Journal of Symbolic Computation* **27** (1999), 351-365.

[6] B. Buchberger. Gröbner bases: an algorithmic method in polynomial ideal theory. In *Recent Trends in Multidimensional Systems Theory*, N. K. Bose, ed., D. Reidel Publishing Company, Dordrecht, 1985, 184-232.

[7] B. Buchberger. History and basic features of the critical-pair/completion procedure. *Journal of Symbolic Computation*, **2** (1987), 3-38.

[8] C. C. Carøe. Decomposition in stochastic integer programming. PhD thesis, University of Copenhagen, 1998.

[9] C. C. Carøe and R. Schultz. A two-stage stochastic program for unit commitment under uncertainty in a hydro-thermal system. Preprint SC 98-11, Konrad-Zuse-Zentrum für Informationstechnik Berlin, 1998, downloadable as SC 98-11 from `http://www.zib.de/bib/pub/pw/index.en.html`.

[10] C. C. Carøe and R. Schultz. Dual decomposition in stochastic integer programming. *Operations Research Letters* **24** (1999), 37-45.

[11] C. C. Carøe and J. Tind. A cutting plane approach to mixed 0-1 stochastic integer programs. *European Journal of Operational Research* **101** (1997), 306-316.

[12] C. C. Carøe and J. Tind. L-shaped decomposition of two-stage stochastic programs with integer recourse. *Mathematical Programming* **83** (1998), 451-464.

[13] P. Conti and C. Traverso. Buchberger algorithm and integer programming. In *Proceedings AAECC-9, (New Orleans)*, LNCS **539**, Springer-Verlag, 1991, 130-139.

[14] W. Cook, A. M. H. Gerards, A. Schrijver, and É. Tardos. Sensitivity theorems in integer linear programming. *Mathematical Programming* **34** (1986), 251-264.

[15] G. Cornuéjols, R. Urbaniak, R. Weismantel, and L. Wolsey. Decomposition of integer programs and of generating sets. LNCS **1284**, Springer-Verlag, 1997, 92-103.

[16] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties, Algorithms.* Springer-Verlag, 1992.

[17] A. D. Foroudi and J. E. Graver. On the foundations of mixed-integer programming. *Unpublished manuscript*, Department of Mathematics, Syracuse University, 1989.

[18] F. R. Giles and W. R. Pulleyblank. Total dual integrality and integer polyhedra. *Linear Algebra and its Applications* **25** (1979), 191-196.

[19] J. E. Graver. On the foundation of linear and integer programming I. *Mathematical Programming* **9** (1975), 207-226.

[20] U. U. Haus, M. Köppe, and R. Weismantel. The integral basis method for integer programming. *Mathematical Methods of Operations Research*, **53** (2001).

[21] U. U. Haus, M. Köppe, and R. Weismantel. A primal all-integer algorithm based on irreducible solutions. Preprint, Otto-von-Guericke-Universität Magdeburg, 2001, downloadable from `http://www.math.uni-magdeburg.de/preprints/01.html`.

[22] R. Hemmecke. Homepage on test sets. URL=`http://www.testsets.de`.

[23] M. Henk, M. Köppe, and R. Weismantel. Integral decomposition of polyhedra and some applications in mixed integer programming. Preprint, Otto-von-Guericke-Universität Magdeburg, 2000, downloadable from `http://www.math.uni-magdeburg.de/preprints/01.html`.

[24] J. L. Higle and S. Sen. *Stochastic Decomposition: A Statistical Method for Large Scale Stochastic Linear Programming.* Kluwer, Dordrecht, 1996.

[25] S. Hosten and B. Sturmfels. GRIN: An implementation of Gröbner bases for integer programming. In *Integer programming and combinatorial optimization*, E. Balas and J. Clausen, eds., LNCS **920**, Springer-Verlag, 1995, 267-276.

[26] P. Kall and S. W. Wallace. *Stochastic Programming.* Wiley, Chichester, 1994.

[27] W. K. Klein Haneveld and M. H. van der Vlerk. Stochastic integer programming: General models and algorithms. *Annals of Operations Research* **85** (1999), 39-58.

[28] M. Köppe. Erzeugende Mengen für gemischt-ganzzahlige Programme. Diploma thesis, Otto-von-Guericke-Universität Magdeburg, 1999, downloadable from `http://www.math.uni-magdeburg.de/~mkoeppe/`.

[29] A. N. Letchford and A. Lodi Primal cutting plane algorithms revisited. In preparation.

[30] Q. Li, Y. Guo, T. Ida, and J. Darlington. The minimised geometric Buchberger algorithm: an optimal algebraic algorithm for integer programming. In *Proceedings ISAAC-97*, ACM, 1997, 331-338.

[31] A. Løkketangen and D. L. Woodruff. Progressive hedging and tabu search applied to mixed integer (0,1) multi-stage stochastic programming. *Journal of Heuristics* **2** (1996), 111-128.

[32] D. Maclagan. Antichains of monomial ideals are finite. Electronic preprint, University of California at Berkeley, 1999, downloadable from `http://front.math.ucdavis.edu/math.CO/9909168`.

[33] D. Maclagan. Structures on sets of monomial ideals. PhD thesis, University of California at Berkeley, 2000.

[34] M. P. Nowak. Stochastic Lagrangian relaxation in power scheduling of a hydro-thermal system under uncertainty. PhD thesis, Humboldt University Berlin, 1999.

[35] M. P. Nowak and W. Römisch. Stochastic Lagrangian relaxation applied to power scheduling in a hydro-thermal system under uncertainty. Preprint 98-36, Deutsche Forschungsgemeinschaft, Schwerpunktprogramm "Echtzeit-Optimierung großer Systeme", 1998.

[36] J. G. Oxley. *Matroid Theory.* Oxford University Press, 1992.

[37] L. Pottier. Euclide's algorithm in dimension n. Research report, ISSAC **96**, ACM Press, 1996.

[38] A. Prékopa. *Stochastic Programming.* Kluwer, Dordrecht, 1995.

[39] R. T. Rockafellar and R. J.-B. Wets. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research* **16** (1991), 119-147.

[40] W. Römisch and R. Schultz: Multistage stochastic integer programs: an introduction. Preprint SM-DU-496, University of Duisburg, 2001, downloadable from `http://www.uni-duisburg.de/FB11/disma/`.

[41] A. Ruszczyński. Some advances in decomposition methods for stochastic linear programming. *Annals of Operations Research* **85** (1999), 153-172.

[42] A. Schrijver. *Theory of Linear and Integer Programming.* Wiley, Chichester, 1986.

[43] H. E. Scarf. Production sets with indivisibilities, Part I: Generalities. *Econometrica* **49** (1981), 1-32.

[44] H. E. Scarf. Neighborhoodsystems for production sets with indivisibilities. *Econometrica* **54** (1986), 507-532.

[45] H. E. Scarf. Test sets for integer programs. In *Mathematical Programming*, T. M. Liebling and D. de Werra, eds., vol. **79** of B, Mathematical Programming Society, 1997, 355-368.

[46] R. Schultz. On structure and stability in stochastic programs with random technology matrix and complete integer recourse. *Mathematical Programming* **70** (1995), 73-89.

[47] R. Schultz, L. Stougie, and M. H. van der Vlerk. Solving stochastic programs with integer recourse by enumeration: A framework using Gröbner basis reductions. *Mathematical Programming* **83** (1998), 229-252.

[48] B. Sturmfels. *Gröbner Bases and Convex Polytopes.* American Mathematical Society, Providence, Rhode Island, 1995.

[49] B. Sturmfels and R. R. Thomas. Variation of cost functions in integer programming. *Mathematical Programming* **77** (1997), 357-387.

[50] B. Sturmfels, R. Weismantel, and G. M. Ziegler. Gröbner bases of lattices, corner polyhedra, and integer programming. *Beiträge zur Algebra und Geometrie/Contributions to Algebra and Geometry* **36** (1995), 282-298.

[51] S. Takriti, J. R. Birge, and E. Long, A stochastic model for the unit commitment problem. *IEEE Transactions on Power Systems* **11** (1996), 1497-1508.

[52] R. R. Thomas. A geometric Buchberger algorithm for integer programming. *Mathematics of Operations Research* **20** (1995), 864-884.

[53] R. R. Thomas and R. Weismantel. Truncated Gröbner bases for integer programming. *Applicable Algebra in Engineering, Communication and Computing* **8** (1997), 241-257.

[54] R. Urbaniak. Decomposition of generating sets of integer programs. Dissertation, Technische Universität Berlin, 1998.

[55] R. Urbaniak, R. Weismantel, and G. M. Ziegler. A variant of the Buchberger algorithm for integer programming. *SIAM J. Discrete Mathematics* **10** (1997), 96-108.

[56] J. G. van der Corput. Über Systeme von linear-homogenen Gleichungen und Ungleichungen. *Proceedings Koninklijke Akademie van Wetenschappen te Amsterdam* **34** (1931), 368-371.

[57] R. Weismantel. Test sets of integer programs. *Mathematical Methods of Operations Research* **47** (1998), 1-37.

[58] H. P. Williams. Fourier-Motzkin elimination extension to integer programming problems. *Journal of Combinatorial Theory (A)* **21** (1976), 118-123.

[59] H. P. Williams. A characterization of all feasible solutions to an integer program. *Discrete Applied Mathematics* **5** (1983), 147-155.

[60] H. P. Williams. A duality theorem for linear congruences. *Discrete Applied Mathematics* **7** (1984), 93-103.

Erklärung

Hiermit versichere ich, Raymond Hemmecke, daß ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde weder im Inland noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt.

Duisburg, den 22. 5. 2001 Raymond Hemmecke