

Decomposition of Test Sets in Stochastic Integer Programming

Raymond Hemmecke

Department of Mathematics, Gerhard-Mercator University Duisburg
Lotharstr. 65, D-47048 Duisburg, Germany
hemmecke@math.uni-duisburg.de

Rüdiger Schultz

Department of Mathematics, Gerhard-Mercator University Duisburg
Lotharstr. 65, D-47048 Duisburg, Germany
schultz@math.uni-duisburg.de

October 26, 2000

Abstract

Graver test sets for linear two-stage stochastic integer programs are studied. It is shown that test sets can be decomposed into finitely many building blocks whose number is independent on the number of scenarios of the stochastic program. A finite algorithm to compute the building blocks directly, without prior knowledge of test set vectors, is presented. Once computed, building blocks can be employed to solve the stochastic program by a simple augmentation scheme, again without explicit knowledge of test set vectors. Preliminary computational experience is reported.

Key Words. test sets, stochastic integer programming, decomposition methods

AMS subject classifications. 90C15, 90C10, 13P10

1 Introduction

The two-stage mixed-integer linear stochastic program is the optimization problem

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

where

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

and

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

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 $h^T x$ 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 [2, 18, 25].

Under mild assumptions, all ingredients in the above model are well defined ([29]). 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 ([2, 18, 25, 29]), 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) ξ^1, \dots, ξ^N and probabilities π^1, \dots, π^N . Then (1)-(3) is equivalent to the mixed-integer linear program

$$\min\{h^T x + \sum_{\nu=1}^N \pi^\nu q^T y^\nu : Ax = a, x \in X, Tx + Wy^\nu = \xi^\nu, y^\nu \in Y, \nu = 1, \dots, N\}. \quad (4)$$

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 ([2, 17, 18, 25, 27]). 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 (cf. [19] for a recent survey). So far, two main lines of research have been pursued in decomposition of stochastic 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.

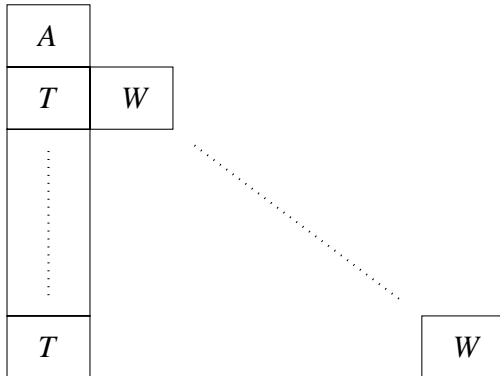


Figure 1: Constraints matrix structure of (4)

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, \dots, y^N) can be carried out separately for each y^ν . The major difficulty with this approach 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 ([5, 8, 9]), to solving the master problem by enumeration and bounding while handling the second-stage problems by methods exploiting problem similarities ([30]), 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 the scenarios $\xi^\nu, \nu = 1, \dots, N$ in (4), respectively. Decomposition then is achieved by Lagrangian relaxation of the non-anticipativity constraints. In [20] the framework of progressive hedging ([26]) is adopted where Augmented Lagrangians lead to decomposition into quadratic mixed-integer programs which are tackled by tabu search. In [6, 7] 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 ([23, 24, 33]).

A common feature of all the algorithms discussed above is decomposition of the problem itself. In the present paper we confine ourselves to pure-integer models and adopt an entirely different viewpoint. Instead of decomposition of the problem (4), we study decomposition of an intimately related object, its Graver test set ([13]). Test sets are finite collections of vectors that enable the solution of integer linear programs by simple augmentation procedures. Decomposition of the Graver test set of (4) will finally 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 on that number.

This paper is organized as follows. In Section 2 we collect prerequisites on existence and computation of test sets in integer programming. Section 3 contains our main results. We introduce building blocks that can be employed to construct test set vectors for two-stage stochastic integer programs. Existence of a finite set \mathcal{H}_∞ is proved, that contains all the building blocks of test set vectors of (4) for arbitrary objective function and right-hand side vectors, and, most importantly, for arbitrary number N of scenarios. Then we develop a finite algorithm based on a completion procedure from symbolic computation that computes \mathcal{H}_∞ . This algorithm entirely works at the building block level. No explicit information on test set vectors is needed. The set \mathcal{H}_∞ is employed by a finite augmentation procedure for solving (4) that, again, is entirely working at the building block level. The section is completed by a discussion of stochastic programs with simple recourse. In Section 4 we report on initial computational experience. Finally, we have a conclusions section.

2 Test Sets

Before starting our analysis of stochastic programs, we collect some necessary prerequisites on test sets. By \mathbb{Q} and \mathbb{Z} we denote the sets of rationals and integers, respectively. For given $A \in \mathbb{Q}^{l \times d}$ consider the family of optimization problems

$$(IP)_{c,b} : \quad \min\{c^\top z : Az = b, z \in \mathbb{Z}_+^d\}$$

as $c \in \mathbb{R}^d$ and $b \in \mathbb{R}^l$ vary. The subsequent exposition on IP Graver test sets follows the main lines of [14].

Definition 2.1 (Test set)

A set $\mathcal{T}_c \subseteq \mathbb{Z}^d$ is called a test set for the family of problems $(IP)_{c,b}$ as $b \in \mathbb{R}^l$ varies if

1. $c^\top t > 0$ for all $t \in \mathcal{T}_c$, and
2. for every $b \in \mathbb{R}^l$ and for every non-optimal feasible solution $z_0 \in \mathbb{Z}_+^d$ to $Az = b$, there exists a vector $t \in \mathcal{T}_c$ such that $z_0 - t$ is feasible. Such a vector is called an improving vector or an improving direction.

A set \mathcal{T} is called a universal test set for the family of problems $(IP)_{c,b}$ as $b \in \mathbb{R}^l$ and as $c \in \mathbb{R}^d$ vary if it contains a test set \mathcal{T}_c for every $c \in \mathbb{R}^d$.

Once a finite test set \mathcal{T}_c is computed or given, we may apply the following augmentation algorithm in order to solve the optimization problem $(IP)_{c,b}$.

Algorithm 2.2 (*Augmentation Algorithm*)Input: a feasible solution z_0 to $(IP)_{c,b}$, a test set \mathcal{T}_c for $(IP)_{c,b}$ Output: an optimal point z_{\min} of $(IP)_{c,b}$ while there is $t \in \mathcal{T}_c$ with $c^\top t > 0$ such that $z_0 - t$ is feasible do

$$z_0 := z_0 - t$$

return z_0

Under the assumption that the optimization problem is solvable this augmentation process always terminates with an optimal solution to $(IP)_{c,b}$ ([34, 14]). Thus, if we were given a finite universal test set for $(IP)_{c,b}$ then for any given right-hand-side b and for any given cost function vector c an optimal solution to $(IP)_{c,b}$ could be easily found as long as an initial feasible solution is available.

It is, however, already a hard problem to find an initial feasible solution to $(IP)_{c,b}$. If we dropped the condition $z \geq 0$ then an integer solution could be found in polynomial time ([28]). Although there are algorithmic alternatives, universal test sets can be used to transform an integer solution to $Az = b$ into a feasible solution to $(IP)_{c,b}$, that is, an integer solution to $Az = b$ whose components are non-negative.

Algorithm 2.3 (*Algorithm to Find a Feasible Solution*)Input: a solution $z_1 \in \mathbb{Z}^d$ to $Az = b$, a universal test set \mathcal{T} for $(IP)_{c,b}$ Output: a feasible solution to $(IP)_{c,b}$ or “FAIL” if no such solution existswhile there is some $g \in \mathcal{T}$ such that $g \leq z_1^+$ and $\|(z_1 - g)^-\|_1 < \|z_1^-\|_1$ do

$$z_1 := z_1 - g$$

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

Here, z^+ denotes the vector whose components $(z^+)^{(i)}$ are given by $(z^+)^{(i)} := \max(0, z^{(i)})$. Accordingly, we use the notation z^- for the vector whose components $(z^-)^{(i)}$ are given by $(z^-)^{(i)} := \max(0, -z^{(i)})$.

Algorithm 2.3 always terminates and returns a feasible solution to $(IP)_{c,b}$ or “FAIL” if no such solution exists ([14]). Thus, if the problem $(IP)_{c,b}$ is bounded with respect to c , we can solve $(IP)_{c,b}$ by means of a finite universal test set as follows.

Algorithm 2.4 (*Algorithm to Find an Optimum of $(IP)_{c,b}$*)Input: $(IP)_{c,b}$, a finite universal test set \mathcal{T} for $(IP)_{c,b}$ Output: an optimum solution to $(IP)_{c,b}$ or “FAIL” if problem is not solvable

$$z_1 := \text{solution to } Az = b, z \in \mathbb{Z}^d \tag{[28]}$$

if no such solution exists then return “FAIL”

$$z_0 := \text{feasible solution } (z_1, \mathcal{T}) \tag{Algorithm 2.3}$$

if no such solution exists then return “FAIL”

$$z_{\min} := \text{optimal solution } (z_0, c, \mathcal{T}) \tag{Algorithm 2.2}$$

return z_{\min}

Thus, finite universal test sets can be used to find an initial feasible solution of $(IP)_{c,b}$ for any given right-hand-side $b \in \mathbb{R}^l$ and to augment this solution to optimality for any given cost function $c \in \mathbb{R}^d$. This enormous amount of information stored in a universal test set leads to huge test sets already for small problems with about 100 variables, say.

Naturally, all the vectors in the integer kernel $\ker(A) := \{v \in \mathbb{Z}^d : Av = 0\}$ of A form an infinite universal test set. However, for all rational matrices A there always exist finite universal test sets as well. In the following we present a particular universal test set, the IP Graver test set, and show how to compute it. IP Graver test sets, together with their LP counterparts, were already introduced by Graver [13] in 1975. Further material on Graver test sets both for LP and IP can be found in [13, 14, 31, 32, 36]. For a general survey on test sets we refer to [36]. We will confine ourselves to IP Graver test sets. Then, the following will turn out useful.

Definition 2.5 (*Hilbert basis*)

Let C be a polyhedral cone with rational generators. A finite set $H = \{h_1, \dots, h_t\} \subseteq C \cap \mathbb{Z}^d$ is 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 $\lambda_1, \dots, \lambda_t$.

The name Hilbert basis was introduced by Giles and Pulleyblank [12] in the context of totally dual integral systems. Note that every pointed, rational cone has a unique Hilbert basis that is minimal with respect to inclusion ([28, 35]). Let \mathbb{O}_j be the j^{th} orthant of \mathbb{Z}^d and $H_j(A)$ be the unique minimal Hilbert basis of the pointed rational cone $\{v \in \mathbb{R}^d : Av = 0\} \cap \mathbb{O}_j$.

Lemma 2.6 $\mathcal{G}(A) := \bigcup H_j(A) \setminus \{0\}$ is a universal test set, called the IP Graver test set or IP Graver basis, for the family of problems $(IP)_{c,b}$ as $b \in \mathbb{R}^l$ and $c \in \mathbb{R}^d$ vary.

Now let $u \sqsubseteq v$ iff $u^+ \leq v^+$ and $u^- \leq v^-$. Then the elements of $\mathcal{G}(A)$ are minimal in $\ker(A) \setminus \{0\}$ with respect to the partial ordering \sqsubseteq on \mathbb{Z}^d . IP Graver test sets exist, are finite, and can be computed using the algorithmic pattern of a completion procedure ([13, 14, 4]).

Algorithm 2.7 (*Algorithm to Compute IP Graver Test Sets*)

Input: $F = \bigcup_{f \in F(A)} \{f, -f\}$, where $F(A)$ is a set of vectors generating $\ker(A)$ over \mathbb{Z}

Output: a set G which contains the IP Graver test set $\mathcal{G}(A)$

$G := F$

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

while $C \neq \emptyset$ do

$s :=$ an element in C

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

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

if $f \neq 0$ then

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

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

return G .

Behind the function $\text{normalForm}(s, G)$ there is the following algorithm.

Algorithm 2.8 (*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 $g \sqsubseteq s$ do

$s := s - g$

return s

In the IP-case we aim at computing \sqsubseteq -minimal vectors of $\ker(A)$. This motivates to say that s can be reduced by g to $s - g$ if $g \sqsubseteq s$. In this case s , g , and $s - g$ all belong to the same orthant.

Lemma 2.9 *With the above specifications the Graver test set algorithm always terminates and computes a set containing the IP Graver test set.*

The Graver test set itself is the set of all elements $z \in G$ which are irreducible with respect to $G \setminus \{z\}$.

The Graver test set algorithm works correctly for arbitrary choices of the element $s \in C$ in the Graver test set algorithm and of the element g in the normal form algorithm. For proofs of Lemmas 2.6 and 2.9 see for example [14].

3 Decomposition of Graver Test Sets for Two-Stage Stochastic Integer Programs

3.1 Building Blocks of Graver Test Sets

As we have seen above, given solvability, the optimization problem $(IP)_{c,b}$ 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) 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 prohibitive.

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 will show that these building blocks can be computed without computing the Graver test set of (4) itself and that we can reconstruct an improving vector to a given non-optimal feasible solution to (4), scenario by scenario, using building blocks only. Incorporating this into Algorithm 2.4, we will find an optimal solution with comparably small effort, once the building blocks have been computed.

Let us remark that this decomposition idea applies to the LP-case as well, where a counterpart to the IP Graver test set exists ([13]). Here we will deal with the IP-case only. The LP-case will be treated in [15].

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 $c = (c_0, c_1, \dots, c_N)^\top := (h, \pi_1 q, \dots, \pi_N q)^\top$ and the right-hand-side $b = (a, \xi^1, \dots, \xi^N)^\top$, where the subscript N corresponds to the number of scenarios, i.e., the number of T 's and W 's used. Problem (4) then may be written as $\min\{c^\top z : A_N z = b, z \in \mathbb{Z}_+^d\}$ with $d = m + Nn$ and m, n as in (1)-(3). We assume all entries in A, T , and W to be rational.

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

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

Lemma 3.1 $(u, v_1, \dots, v_N) \in \ker(A_N)$ if and only if $(u, v_1), \dots, (u, v_N) \in \ker(A_1)$.

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

This guarantees that by permuting the v_i we do not leave $\ker(A_N)$. Moreover, a \sqsubseteq -minimal element of $\ker(A_N)$ will always be transformed into a \sqsubseteq -minimal element of $\ker(A_N)$. 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 3.2 (*Building blocks*)

Let $z = (u, v_1, \dots, v_N) \in \ker(A_N)$ and call the vectors u, v_1, \dots, 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}_∞ denote the set $\bigcup_{N=1}^\infty \mathcal{H}_N$.

The set \mathcal{H}_∞ 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}_∞ 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 $(u, v) \in \ker(A_1)$. 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}_∞ .

The set \mathcal{H}_∞ 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}_∞ , computation of \mathcal{H}_∞ , and reconstruction of improving vectors using \mathcal{H}_∞ .

3.2 Finiteness of \mathcal{H}_∞

As a main result of our paper we will prove finiteness of \mathcal{H}_∞ . To this end, we have to use some further algebraic machinery. For an introduction into basic notions we refer to [11]. In order to prove the subsequent Theorem 3.13 we will use the following recent theorem on monomial ideals.

Theorem 3.3 (*Maclagan, [21, 22]*)

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 $I \subseteq J$.

Remark. This theorem contains as a special case the Gordan-Dickson-Lemma (see for example [11]), which was used in [14] to show finiteness of the algorithm to compute IP Graver test sets. \square

We now add two useful corollaries (cf. [21]) and, for convenience, include their short proofs.

Corollary 3.4 *Let \mathcal{I} be an infinite collection of 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} . \square

Corollary 3.5 *Let (I_1, I_2, \dots) 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 $\{I_j : j = 1, \dots\}$. Then M is finite by Corollary 3.4. Thus, there is some $k \in \mathbb{Z}_+$ such that $M \subseteq \{I_1, \dots, I_k\}$. Therefore, for all $j > k$ there is some $i \in \{1, \dots, k\}$ satisfying $I_j \subseteq I_i$, in contradiction to the assumption that no such pair i, j exists. \square

We will use this last corollary to obtain a similar result on sequences of pairs which will be employed to prove termination of the subsequent algorithm. To this end, we define the following notions.

Definition 3.6 *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 $0 \neq v' \sqsubseteq v$.

Definition 3.7 We associate with (u, V_u) , $u \neq 0$, the monomial ideal $I(u, V_u) \in Q[x_1, \dots, x_{2m+2n}]$ 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 $I(0, V_0) \in Q[x_1, \dots, x_{2n}]$ generated by all the monomials $x^{(v^+, v^-)}$ with $v \neq 0$ and $v \in V_0$.

Lemma 3.8 Let $(u, V_u), (u', V_{u'})$ with $u, u' \neq 0$. 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 both $I(u, V_u)$ 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. [11]). The latter implies that $u' \sqsubseteq u$ and that for every $v \in V_u$ there exists $v' \in V_{u'}$ with $v' \sqsubseteq v$. In other words, we have $(u', V_{u'}) \sqsubseteq (u, V_u)$, and the proof is complete. \square

Lemma 3.9 $(0, V_0') \not\sqsubseteq (0, V_0)$ 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 $v' \in V_0'$ with $v' \neq 0$ and $v' \sqsubseteq v$. But this means that $v \notin I(0, V_0')$ and, therefore, we have $I(0, V_0) \not\subseteq I(0, V_0')$. \square

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

Lemma 3.10 Let $((u_1, V_{u_1}), (u_2, V_{u_2}), \dots)$ be a sequence of pairs such that $u_i \neq 0$ for all $i = 1, 2, \dots$, 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 $(I(u_1, V_{u_1}), I(u_2, V_{u_2}), \dots)$ of monomial ideals. By Lemma 3.8, it fulfils $I(u_j, V_{u_j}) \not\subseteq I(u_i, V_{u_i})$ whenever $i < j$. Thus, by Corollary 3.5, this sequence is finite implying that the sequence $((u_1, V_{u_1}), (u_2, V_{u_2}), \dots)$ is finite, as well. \square

Lemma 3.11 Let $((0, V_1), (0, V_2), \dots)$ be a sequence of pairs such that $(0, V_i) \not\sqsubseteq (0, V_j)$ whenever $i < j$. Then this sequence is finite.

Proof. Consider the sequence $(I(0, V_1), I(0, V_2), \dots)$ of monomial ideals. By Lemma 3.9, it fulfils $I(0, V_j) \not\subseteq I(0, V_i)$ whenever $i < j$. Thus, by Corollary 3.5, this sequence is finite implying that the sequence $((0, V_1), (0, V_2), \dots)$ is finite, as well. \square

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

Lemma 3.12 Let $((u_1, V_{u_1}), (u_2, V_{u_2}), \dots)$ be a sequence of pairs 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 $((u_1, V_{u_1}), (u_2, V_{u_2}), \dots)$ is not finite. Consider the subsequence where all u_i are non-zero and the subsequence where all u_i are zero. 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. \square

This lemma will imply finiteness of the subsequent algorithm to compute \mathcal{H}_∞ . Below we will prove correctness of this algorithm. Then, termination and correctness together imply finiteness of \mathcal{H}_∞ .

Theorem 3.13 Given rational matrices A, T , and W of appropriate dimensions, and let \mathcal{H}_∞ be defined as above. Then \mathcal{H}_∞ is a finite set.

3.3 Computation of \mathcal{H}_∞

Using the finiteness of \mathcal{H}_∞ , 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 computation from Algorithm 2.7, but to work with pairs (u, V_u) instead and to define the two main ingredients, normalForm and S-vectors, appropriately. In what follows, the objects f, g , and s all are pairs of the form (u, V_u) .

Algorithm 3.14 (Algorithm to compute \mathcal{H}_∞)

Input: a generating set F of $\ker(A_1)$ in (u, V_u) -notation to be specified below

Output: a set G which contains \mathcal{H}_∞

$G := F$

$C := \bigcup_{f, g \in G} \{f \oplus g\}$ (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

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

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

return G .

Behind the function $\text{normalForm}(s, G)$ there is the following algorithm.

Algorithm 3.15 (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 where every vector is decomposed into its building blocks.

Lemma 3.16 Let F be a generating set for $\ker(A_1)$ over \mathbb{Z} which contains a generating set for $\{(0, v) : Wv = 0\} \subseteq \ker(A_1)$ consisting only of vectors with zero first-stage component. Moreover, let F_N be the set of all those vectors in $\ker(A_N)$ whose building blocks are also building blocks of vectors in $F \cup \{0\}$. Then, for any N , the vectors F_N generate $\ker(A_N)$ over \mathbb{Z} .

Proof. Let $z = (u, v_1, \dots, v_N) \in \ker(A_N)$. We have to show that z can be written as a linear integer combination of elements from F_N .

Since $(u, v_1) \in \ker(A_1)$ 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 $(u, v_1, \dots, v_1) = \sum_i \alpha_i (u_i, v_{i,1}, \dots, v_{i,1})$ with $(u_i, v_{i,1}, \dots, v_{i,1}) \in F_N$. Moreover, for $j = 2, \dots, N$ we have $W(v_j - v_1) = 0$, that is, $(0, v_j - v_1) \in \ker(A_1)$, since $Tu + Wv_1 = Tu + Wv_j$. Since F contains a generating set for $\{(0, v) : Wv = 0\} \subseteq \ker(A_1)$ consisting only of vectors with zero first-stage component, there are 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 $j = 2, \dots, N$.

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

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

as desired. □

This lemma suggests the following input set.

Definition 3.17 We define the building blocks of all vectors in $F \cup \{0\}$ in (u, V_u) -notation to be the input set to the above algorithm. Herein, F is a generating set for $\ker(A_1)$ over \mathbb{Z} which contains a generating set for $\{(0, v) : Wv = 0\} \subseteq \ker(A_1)$ consisting only of vectors with zero first-stage component.

Definition 3.18 Let

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

where

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

Moreover, let

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

Remark. In $(u, V_u) \ominus (u', V_{u'}) := (u - u', \{v - v' : v \in V_u, v' \in V_{u'}, v' \sqsubseteq v\})$ it suffices to collect only one difference $v - v'$ for every $v \in V_u$. It will be elaborated in the proof of the subsequent proposition that Algorithm 3.14 still terminates and works correctly if we defined $(u, V_u) \ominus (u', V_{u'})$ this way. \square

Proposition 3.19 If the input set, the procedure normalForm, and $f \oplus g, s \ominus g$ are defined as above then Algorithm 3.14 terminates and satisfies its specifications.

Proof. In the course of the algorithm, a sequence of pairs in $G \setminus F$ is generated that satisfies the conditions of Lemma 3.12. Therefore, Algorithm 3.14 terminates.

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 (Algorithm 2.7) with $\bar{F} := \{(u, v_1, \dots, v_N) : (u, V_u) \in G, v_i \in V_u\}$ as input set. \bar{F} generates $\ker(A_N)$ over \mathbb{Z} for all $N \in \mathbb{Z}_+$, since $F_N \subseteq \bar{F}$ by the assumption on the input set to Algorithm 3.14.

We will now show that all sums $z + z'$ of two elements $z, z' \in \bar{F}$ reduce to 0 with respect to \bar{F} . In this case, Algorithm 2.7 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 $z = (u, v_1, \dots, v_N)$ and $z' = (u', v'_1, \dots, v'_N)$ from \bar{F} , and consider the vector $z + z' = (u + u', v_1 + v'_1, \dots, v_N + v'_N)$.

In the above algorithm, $(u, V_u) \oplus (u', V_{u'})$ was reduced to zero by elements $(u_1, V_{u_1}), \dots, (u_k, V_{u_k}) \in G$. From this sequence we can construct a sequence z_1, \dots, 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 $v_{1,1}, \dots, v_{1,N} \in V_{u_1}$ such that $v_{1,i} \sqsubseteq v_i + v'_i$ for $i = 1, \dots, N$. Therefore, $z_1 := (u_1, v_{1,1}, \dots, v_{1,N}) \sqsubseteq z + z'$ 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 by $(u_2, V_{u_2}), \dots, (u_k, V_{u_k}) \in G$. Therefore, we can construct from (u_2, V_{u_2}) a vector $z_2 \in \bar{F}$ with $z_2 \sqsubseteq z + z' - z_1$. Therefore, $z + z' - z_1$ can be further reduced to $z + z' - z_1 - z_2$.

Repeating this construction, we arrive in the k^{th} step at $z + z' - z_1 - \dots - z_{k-1}$ whose building blocks all lie in $((u, V_u) \oplus (u', V_{u'})) \ominus (u_1, V_{u_1}) \ominus \dots \ominus (u_{k-1}, V_{u_{k-1}})$. 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 $z_k \sqsubseteq z + z' - z_1 - \dots - z_{k-1}$ and $0 = z + z' - z_1 - \dots - z_k$. \square

Note that termination and correctness of the above algorithm imply finiteness of \mathcal{H}_∞ . Therefore, the above proves Theorem 3.13.

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

As we have seen in Section 2, universal test sets help to find an initial feasible solution to our optimization problem, and to augment it to optimality. For our optimization problem $\min\{c^\top z : A_N z = n, z \in \mathbb{Z}_+^{m+Nn}\}$, however, the universal test set is not given explicitly. Only its set of building blocks \mathcal{H}_∞ is available. In the following, we will see how the vectors needed for the particular algorithms can be reconstructed from \mathcal{H}_∞ .

3.4.1 Feasible Initial Solutions

Suppose we are given an integer solution $z_1 = (u, v_1, \dots, v_N)$ to $A_N z = b$. If $z_1 \geq 0$ then we have already found an initial feasible solution $z_0 := z_1$, and we can start our augmentation algorithm. If z_1 is not yet feasible we can apply the simple algorithm presented in Section 2 which either produces a feasible solution z_0 or decides that no such exists.

The main ingredient of this algorithm is to find a vector z' in the universal test set which satisfies $\|(z_1 - z')^-\|_1 < \|z_1^-\|_1$. In the following we will show how to construct such a vector z' from \mathcal{H}_∞ .

Define the auxiliary cost function c' by

$$(c')^{(i)} := \begin{cases} 0 & \text{if } z_1^{(i)} \geq 0 \\ -1 & \text{if } z_1^{(i)} < 0 \end{cases}, \text{ for } i = 1, \dots, m + Nn.$$

Lemma 3.20 *Suppose $z_1 = (u, v_1, \dots, v_N) \not\geq 0$ and there exists no pair $(u', V_{u'}) \in \mathcal{H}_\infty$ such that*

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

Then there exists no feasible solution of $A_N z = b, z \in \mathbb{Z}_+^{m+Nn}$.

If there exists such a pair $(u', V_{u'}) \in \mathcal{H}_\infty$ then $\|(z_1 - z')^-\|_1 < \|z_1^-\|_1$.

Proof. Suppose that there exists a feasible solution to $A_N z = b, z \geq 0$. Thus, as was pointed out in Section 2, there has to exist a vector $z'' = (u'', v''_1, \dots, v''_N) \in \mathcal{G}_N$ such that $z'' \leq (z')^+$ and $\|(z_1 - z'')^-\|_1 < \|z_1^-\|_1$. This implies that $(c')^\top z'' > 0$.

Choose $u' = u''$ and consider the pair $(u', V_{u'}) \in \mathcal{H}_\infty$. By construction, the pair $(u', V_{u'})$ satisfies the first two conditions of our lemma, since we can choose $v'_i = v''_i \in V_{u'}$. Since $(c')^\top z' \geq (c')^\top z'' > 0$ the pair $(u', V_{u'})$ also satisfies the third condition.

It remains to show that $\|(z_1 - z')^-\|_1 < \|z_1^-\|_1$. But $z' \leq (z_1)^+$ implies $(z_1 - z')^{(i)} \geq 0$ whenever $z_1^{(i)} \geq 0$. Therefore, $(c')^\top z' > 0$ implies that $\|(z_1 - z')^-\|_1 < \|z_1^-\|_1$. \square

3.4.2 Reconstruction of Improving Vectors

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

Lemma 3.21 *Suppose there exists no pair $(u', V_{u'}) \in \mathcal{H}_\infty$ with the properties*

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

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

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

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

Now $c^\top(z_0 - z'') < c^\top z_0$ implies that $c^\top z'' > 0$. Moreover, $c^\top z' \geq c^\top z'' > 0$. In conclusion, the pair $(u'', V_{u''})$ fulfils conditions 1. – 3. proving the first claim of the lemma.

With $z' = (u', v'_1, \dots, v'_N)$ according to 3. we obtain $c^\top(z_0 - z') < c^\top z_0$. Moreover $v'_i \leq v_i$, $i = 1, \dots, N$, and $u' \leq u$ together imply $z' \leq z_0$, and $z_0 - z' \geq 0$. Finally, $(u', v'_1, \dots, v'_N) \in \ker(A_N)$, and therefore $A_N(z_0 - z') = A_N z_0 + 0 = b$ which completes the proof. \square

Theorem 3.22 *Under the assumption that the optimization problem $\min\{c^\top z : A_N z = b, z \in Z_+^{m+N \cdot n}\}$ is solvable, an optimal solution can be computed in finitely many steps by application of Algorithm 2.4 together with the above reconstruction procedures.*

The reconstruction procedures in the above lemma yield an improving vector in linear time with respect to the number N of scenarios. In accordance with that, we observed in our preliminary test runs (cf. Section 4) 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}_∞ has been computed.

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 ([2, 3, 18, 25]) and two-stage integer linear programs ([19]). In the following we will 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. In order to show this we have to state another property of Graver bases. A similar lemma for Graver test sets of knapsack problems can be found in [10].

Lemma 3.23 *Given a matrix $B = \begin{pmatrix} A & a & -a \end{pmatrix}$, with two columns which differ only in their respective signs. Then the Graver basis of B can be constructed from the Graver basis of $B' = \begin{pmatrix} A & a \end{pmatrix}$ in the following way:*

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

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

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

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 since $0 \leq z'$, the second can be seen as follows. If $z' \leq v$ then $0 \geq -z' + z' = 0 \geq w$ by our above assumption on w . If, on the contrary, $z' \geq v$ we have $0 \geq -z' + v \geq -(v - w) + v = w$. This yields $(u', \min(z', v), -z' + \min(z', v)) \sqsubseteq (u, v, w)$. Moreover, $(u', \min(z', v), -z' + \min(z', v)) \in \ker(B)$ and $u' \neq 0$, so it remains to prove that $(u', \min(z', v), -z' + \min(z', v)) \neq (u, v, w)$ which then implies that (u, v, w) is not \sqsubseteq -minimal in $\ker(B)$ and therefore $(u, v, w) \notin \mathcal{G}(B)$.

Suppose that $(u', \min(z', v), -z' + \min(z', v)) = (u, v, w)$, which 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 above. Thus $(u, v, w) \notin \mathcal{G}(B)$.

Therefore, it remains to prove that every vector (u, v, w) with $u \neq 0$, $v \geq 0$, $w \leq 0$, and $(u, v - w) \in \mathcal{G}(B')$ belongs to $\mathcal{G}(B)$. Clearly, $(u, v, w) \in \ker(B)$. Suppose there were a vector $(u', v', w') \in \ker(B)$ with $u' \neq 0$, $(u', v', w') \sqsubseteq (u, v, w)$, and $(u, v, w) \neq (u', v', w')$. Then $(u', v' - w') \in \ker(B')$ and $(u', v' - w') \sqsubseteq (u, v - w)$. If $(u, v - w) \neq (u', v' - w')$ this contradicts $(u, v - w) \in \mathcal{G}(B')$.

So suppose $(u, v - w) = (u', v' - w')$, $(u', v', w') \sqsubseteq (u, v, w)$, and $(u, v, w) \neq (u', v', w')$. But then $0 \leq v' \leq v$ and $0 \geq w' \geq w$ 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 $v - w = v' - w'$. \square

Corollary 3.24 *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 Graver basis of B can be constructed from the Graver basis of $B' = \begin{pmatrix} A & D \end{pmatrix}$ in the following way:*

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

Herein, v , w , and 1 are vectors whose dimensions equal the number of columns of D . Moreover, we use $vw \leq 0$ in the sense that $v^{(i)}w^{(i)} \leq 0$ for every component i .

Proof. The proof is analogous to the one above, since we can consider the pairs of columns in D and $-D$ independently. \square

Lemma 3.25 *For two-stage stochastic programs with simple recourse \mathcal{H}_∞ coincides with \mathcal{H}_1 .*

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

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

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

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

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

Remark. To reconstruct \mathcal{H}_∞ from \mathcal{H}'_∞ we replace every pair $(u', V_{u'}) \in \mathcal{H}'_\infty$ where $u' \neq 0$ by the pair (u, V_u) given by $u = u'$ and $V_u := \{(v, w) : vw \leq 0, v - w \in V_{u'}\}$. 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. \square

4 Computations

Algorithm 3.14 as well as the initialization and augmentation procedures from Subsection 3.4 have been implemented. The current version of that implementation can be obtained from [16]. 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 3.14. 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

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

Here, the random vector $\xi \in \mathbb{R}^s$ is given by the scenarios ξ^1, \dots, ξ^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}_∞ altogether consisting of 1438 building blocks arranged into 25 pairs (u, V_u) . $\text{Aug}(\mathcal{H}_\infty)$ 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 $y_4^\nu = \xi_2^\nu$, $\nu = 1, \dots, N$ to optimality.

Example	(ξ_1, ξ_2) -grid	(ξ_3, ξ_4) -grid	scenarios	variables	optimum	$\text{Aug}(\mathcal{H}_\infty)$	CPLEX	dualdec
1	5×5	3×3	225	902	(100, 150)	1.52	0.63	> 1800
2	5×5	21×21	11025	44102	(100, 100)	66.37	696.10	–
3	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}_∞ has been computed. Existing methods in stochastic integer programming are much more sensitive on N . The dual decomposition algorithm from [7] 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 dualdec in Examples 2 and 3. On the other hand, dualdec 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, cf. Example 3 in the above table.

5 Conclusions

We presented a novel approach for the solution of linear two-stage stochastic programs based on test set methods. To this end, we constructed a finite object, \mathcal{H}_∞ , which depends only on the matrices A , T , and W , and which allows an algorithmic solution to (4) 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}_∞ , so far, has no counterpart in the existing stochastic programming literature.

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

Finally, let us remark that the presented decomposition approach can be generalized to the multi-stage extension of (4). Again, one can define a set \mathcal{H}_∞ of building blocks not depending on the number of scenarios, one can give an algorithm which, in case of termination, returns \mathcal{H}_∞ , and these building blocks can again be employed to reconstruct improving vectors to a given feasible solution. In this general setting, however, the proof of finiteness of \mathcal{H}_∞ is still part of our ongoing research.

Acknowledgement. We want to thank Rekha Thomas (University of Washington, Seattle) for pointing our attention to Maclagan's theorem ([21, 22]) which finally proved finiteness of \mathcal{H}_∞ . In addition we are grateful to Jesus de Loera (University of California, Davis) for valuable comments on an earlier version of this paper.

References

- [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] J. R. Birge and F. Louveaux. Introduction to Stochastic Programming. Springer, New York, 1997.
- [3] J. R. Birge and R. J.-B. Wets. Sublinear upper bounds for stochastic programs with recourse. Mathematical Programming, 43:131–149, 1989.

- [4] B. Buchberger. History and basic features of the critical-pair/completion procedure. *Journal of Symbolic Computation*, 2:3–38, 1987.
- [5] C. C. Carøe. Decomposition in stochastic integer programming. PhD thesis, University of Copenhagen, 1998.
- [6] 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>.
- [7] C. C. Carøe and R. Schultz. Dual decomposition in stochastic integer programming. *Operations Research Letters* 24 (1999), 37–45.
- [8] 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.
- [9] C. C. Carøe and J. Tind. L-shaped decomposition of two-stage stochastic programs with integer recourse. *Mathematical Programming* 83 (1998), 451–464.
- [10] G. Cornuejols, R. Urbaniak, R. Weismantel, and L. Wolsey. Decomposition of integer programs and of generating sets. LNCS 1284, Springer-Verlag, 1997, pp. 92–103.
- [11] D. Cox, J. Little, and D. O’Shea. *Ideals, Varieties, Algorithms*. Springer-Verlag, 1992.
- [12] F. R. Giles and W. R. Pulleyblank. Total dual integrality and integer polyhedra. *Linear Algebra and its Applications*, 25:191–196, 1979.
- [13] J. E. Graver. On the foundation of linear and integer programming I. *Mathematical Programming*, 9:207–226, 1975.
- [14] R. Hemmecke. On the positive sum property of Graver test sets. Preprint SM-DU-468, University of Duisburg, 2000, downloadable via <http://www.uni-duisburg.de/FB11/disma/ramon/articles/preprint2.ps>.
- [15] R. Hemmecke. Decomposition of test sets in linear stochastic programming. in preparation.
- [16] R. Hemmecke. Homepage on test sets. URL=<http://www.testsets.de>.
- [17] J. L. Hight and S. Sen. *Stochastic Decomposition: A Statistical Method for Large Scale Stochastic Linear Programming*. Kluwer, Dordrecht, 1996.
- [18] P. Kall and S. W. Wallace. *Stochastic Programming*. Wiley, Chichester, 1994.
- [19] 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.
- [20] 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.
- [21] 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>.
- [22] D. Maclagan. Structures on sets of monomial ideals. PhD thesis, University of California at Berkeley, 2000 (submitted).
- [23] M. P. Nowak. Stochastic Lagrangian relaxation in power scheduling of a hydro-thermal system under uncertainty. PhD thesis, Humboldt University Berlin, 1999 (submitted).
- [24] 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.
- [25] A. Prékopa. *Stochastic Programming*. Kluwer, Dordrecht, 1995.

- [26] R. T. Rockafellar and R. J-B. Wets. Scenarios and policy aggregation in optimization under uncertainty. *Mathematics of Operations Research* 16 (1991), 119–147.
- [27] A. Ruszczyński. Some advances in decomposition methods for stochastic linear programming. *Annals of Operations Research* 85 (1999), 153–172.
- [28] A. Schrijver. *Theory of Linear and Integer Programming*. Wiley, Chichester, 1986.
- [29] R. Schultz. On structure and stability in stochastic programs with random technology matrix and complete integer recourse. *Mathematical Programming* 70 (1995), 73–89.
- [30] 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.
- [31] B. Sturmfels. *Gröbner Bases and Convex Polytopes*. American Mathematics Society, Providence, Rhode Island, 1995.
- [32] B. Sturmfels and R. R. Thomas. Variation of cost functions in integer programming. *Mathematical Programming*, 77:357–387, 1997.
- [33] 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.
- [34] R. R. Thomas. A geometric Buchberger algorithm for integer programming. *Mathematics of Operations Research*, 20:864–884, 1995.
- [35] J. G. van der Corput. Über Systeme von linear-homogenen Gleichungen und Ungleichungen. *Proceedings Koninklijke Akademie van Wetenschappen te Amsterdam*, 34:368–371, 1931.
- [36] R. Weismantel. Test sets of integer programs. *Mathematical Methods of Operations Research*, 47:1–37, 1998.