Solving stochastic programs with integer recourse by enumeration: a framework using Gröbner basis reductions

Rüdiger Schultz
Mathematical Institute, University of Leipzig
Augustusplatz 10/11, D-04109 Leipzig, Germany

Leen Stougie
Department of Mathematics and Computing Science
Eindhoven University of Technology
P.O. Box 513, NL-5600 MB Eindhoven, The Netherlands

Maarten H. van der Vlerk
Department of Econometrics, University of Groningen
P.O. Box 800, NL-9700 AV Groningen, The Netherlands

May 15, 1995

Abstract

In this paper we present a framework for solving stochastic programs with complete integer recourse and discretely distributed right-hand side vector, using Gröbner basis methods from computational algebra to solve the numerous second-stage integer programs. Using structural properties of the expected integer recourse function, we prove that under mild conditions an optimal solution is contained in a finite set. Furthermore, we present a basic scheme to enumerate this set and suggest improvements to reduce the number of function evaluations needed.

1 Introduction

In this paper we are concerned with two-stage stochastic integer programs of the type

$$\min\{cx + Q(x) \ : \ x \in C\} \tag{1}$$

where

$$Q(x) = E_\xi v(Tx - \xi) \tag{2}$$

and

$$v(s) = \min\{\tilde{q}y : Wy \geq s, \ y \in \mathbb{Z}_+^m\}. \tag{3}$$

Here c is an n-dimensional vector, $C = \{x \in \mathbb{R}_+^n \ : \ Ax \geq b\}$ is a non-empty polyhedron with A a $q \times n$ matrix and b a q-dimensional vector, T is a $p \times n$-matrix, $\tilde{q}$ is an m-dimensional vector, and W a $p \times m$-matrix. All matrices/vectors have real elements except for W, which is a rational matrix. ξ is a random vector in $\mathbb{R}^p$

and E_ξ denotes the expectation with respect to ξ. For the moment, we assume that both Q and v are well-defined; conditions ensuring that will be given later on.

The stochastic program (1) is designed to finding optimal first-stage decisions x in an optimization problem under uncertainty where the first-stage decisions have to be made before knowing the outcome of the random vector ξ, and where second-stage decisions y serve to compensate possible infeasibilities after having fixed x and observed ξ. For an introduction to two-stage stochastic programming we refer to [14, 21].

In contrast to the many algorithms for stochastic programs with continuous second-stage variables (see e.g. [11, 14, 21]), only few methods have been developed that deal with integer second-stage problems. They usually are restrictive, either in a theoretical sense, or in practical applicability. There are two main difficulties in solving stochastic integer programming problems. The first one is that, in order to compute one function value $Q(x)$, one has to solve many different (but similar) integer programs, which are in general $\mathcal{NP}$-hard.

The other difficulty is that, due to the integer requirements on the second-stage variables y, the value function v in (3) is in general only lower semicontinuous (see [2]) instead of convex for continuous variables y. This destroys the convexity of Q met in the continuous case and solving (1) amounts to minimizing a non-convex and possibly discontinuous objective. The latter is a fairly recent field of research with promising first results [10].

In this paper, we present a framework for solving (1) where the second-stage integer programs are handled via Gröbner basis methods from computational algebra. Employing traditional stochastic programming methodology, the algebraic techniques are embedded into an algorithmic framework that reduces solving (1) to inspecting finite sets of candidate points.

The latter is the main contribution of this paper. Studying the structure of the stochastic integer problems we show that at least some of the optimal first-stage decisions belong to an explicitly given countable set. Under some mild additional assumptions this set can be further restricted to a finite one. To this end we apply certain level sets, which are constructed by solving the continuous relaxation of (1) (i.e., the stochastic program where the integer requirements in the second stage are dropped).

Applying Gröbner bases to solve integer linear programs was first proposed in [7] (see also [31]). It yields additional information at a possibly high computational cost, so that usually it is an inefficient method to solve the problem for a fixed right-hand side. However, the additional information turns out to be highly beneficial when an integer linear program needs to be solved many times, where each time only the right-hand side parameters are different. Therefore, this method does seem particularly useful for solving stochastic programs with integer recourse. Computing the Gröbner basis related to the second-stage integer linear program is in general very time consuming, but has to be done only once. The reason is that the Gröbner basis does not depend on information corresponding to the varying right-hand side, i.e., the realization of ξ and the choice of x.

The existence of the Gröbner basis approach justifies the assumption that it is computationally feasible to evaluate the expected value function Q many times. This assumption is essential for the practicability of our algorithm. However, our algorithm is independent of the actual method used to perform these calculations.

If ξ follows a discrete distribution, problem (1)-(3) is representable as a large-scale mixed-integer linear program with dual block angular structure. If, contrary to our assumptions, all variables are continuous then the equivalent problem is a structured LP, which can be solved by decomposition algorithms related to classical Benders' decomposition, such as the L-shaped method [35], regularized decomposition [23], or stochastic decomposition [12]. In this case the above mentioned convexity of v is crucial. On the other hand, for mixed-integer dual block angular LP's equivalent to (1)-(3) no practicable decomposition methods are known. This paper makes a step forward in the search for such methods: for the complicated lower semicontinuous master problem we work out an enumeration scheme, and for the numerous but similar slave problems we employ an efficient procedure (reductions using a Gröbner basis). Such a decomposition idea, although simple, considerably widens the ability to solve stochastic programs with integer recourse. Indeed, in Section 7 we report on experiments with our method and with the CPLEX 4.0 Mixed-Integer Optimizer (applied to the large-scale block angular MILP version of (1) - (3)). It appears that problems with a second stage of moderate size can be solved by our method, while CPLEX ended up with gaps of about 25% after 50.000 nodes of branching.

The main body of research on two-stage stochastic integer programming has been devoted to structural properties of the second-stage expected value function Q. In [25, 26] complete recourse models are treated, whereas in [16, 19] the focus is on simple integer recourse. For the latter model approximate solution methods based on convex approximations are presented in [15, 34].

First proposals of methods to solve stochastic integer programs are described in [17, 18]. The former approaches the problem by dynamic programming, and can be used only for problems of relatively small size. The approach in the latter paper is based on the assumption that the first stage variables are also integral, thereby obtaining countability of the set of feasible solutions. Moreover, it is implicitly assumed that computation of the second-stage integer problems provides no hardship. An L-shaped method for linear stochastic programs with integer recourse is proposed in [5]. The authors show finiteness of the method but admit that no practicable method for the lower semicontinuous master problem is known. The branch and bound method with stochastic bound estimation, which is developed in [24], can be applied to stochastic programs with integer recourse too. Naturally, this method converges in a stochastic sense (with probability one). Therefore, for a given problem it has to be run several times to produce a reliable solution.

Applicability of Gröbner bases for solving stochastic programming problems has been noticed independently by Tayur et al. In [30] an algorithm for optimization problems under probabilistic constraints is proposed and applied to a real-life scheduling problem. Moreover, Tayur outlined a gradient type algorithm for integer recourse models with continuously distributed parameters in [29]; like in this paper, Gröbner techniques are used as a computational tool. For an overview of the literature on stochastic integer programming we refer to [27, 28].

In Section 2 we describe briefly how a Gröbner basis can be used to evaluate the objective function of a stochastic integer program.

In Section 3 we show that, if the random vector ξ is discretely distributed as we will assume, an optimal solution of (1) is contained in a certain countable set. Under some mild conditions this set of points can even be reduced to a finite set, as exposed in Section 4.

For an effective method the above set of points is to be enumerated completely. In Section 5 we propose a basic enumeration scheme that does this job.

Having presented all ingredients, they are put together in Section 6 which contains a short description of the basic algorithm. This is followed by two improvements, both directed at reducing the number of points to be evaluated .

In Section 7 we illustrate computational possibilities of our algorithm on a few examples of moderate size.

Finally, conclusions and directions for future research on this topic are presented in Section 8.

2 Function evaluations using Gröbner bases

Solving the integer programs behind the function values of the expected recourse function Q is the key problem in stochastic programming with integer recourse. Here, we have to solve

$$\min\{\tilde{q}y : Wy \geq s,\ y \in \mathbb{Z}_+^m\}$$

for arbitrary right-hand sides $s \in \mathbb{Z}^p$ (we assume that W is an integer matrix, hence if $s \notin \mathbb{Z}^p$ then it can be replaced by its componentwise integer round up $\lceil s \rceil$). Without loss of generality we can assume that, after introducing slack variables, the problem (with properly adjusted $\tilde{q}$, W and y) is in equality form

$$\text{(P)} \qquad \min\{\tilde{q}y : Wy = s,\ y \in \mathbb{Z}_+^{\bar{m}}\},$$

with $\bar{m} = m + p$.

For problems of this type, recently a solution technique has been developed based on Gröbner basis methods from computational algebra [7].

For a brief exposition of the method we introduce the problem

$$\text{(IP)} \qquad \min\{c\mathtt{y} \ : \ A\mathtt{y} = b, \mathtt{y} \in \mathbb{Z}_+^n\},$$

where A is an integral $d \times n$-matrix and c, b are integral vectors of dimensions n and d, respectively. To keep the exposition simple we assume here that all entries in c, A and b are non-negative.

Buchberger Algorithm for Integer Linear Programs

STEP 1: Let k be any field and fix the polynomial ring $k[\mathtt{x}_1, \ldots, \mathtt{x}_d, \mathtt{y}_1, \ldots, \mathtt{y}_n]$, written also as $k[\mathtt{x}, \mathtt{y}]$.

STEP 2: Specify a monomial order $\prec$ in $k[\mathtt{x}, \mathtt{y}]$ that is compatible with c and guarantees $\mathtt{x} > \mathtt{y}$.

STEP 3: Form the ideal $I = \langle \mathtt{x}^{a^1} - \mathtt{y}_1, \ldots, \mathtt{x}^{a^n} - \mathtt{y}_n \rangle$., where a^i is the i-th column of A, and $\mathtt{x}^{a^i}$ denotes the monomial $\mathtt{x}_1^{a_{1i}} \cdot \mathtt{x}_2^{a_{2i}} \cdot \ldots \cdot \mathtt{x}_d^{a_{di}}$.

STEP 4: Compute the (reduced) Gröbner basis G of I with respect to $\prec$ using Buchberger's algorithm.

STEP 5: Divide $\mathtt{x}^b$ by G, resulting in the remainder $r_G(\mathtt{x}^b)$.

STEP 6: If $r_G(\mathtt{x}^b) \in k[\mathtt{y}]$ then the remainder is a monomial whose exponent vector is an optimal solution to (IP), otherwise (IP) has no feasible solution.

An introduction covering the algebraic concepts in this exposition can be found in [8], which also contains a description of Buchberger's algorithm to determine a Gröbner basis (originally presented in [3]). For the background and details of the specific application to solving integer linear programs, as well as for the extension to the case in which negative entries are allowed, we refer to [7, 13, 31].

In our integer recourse setting, the interesting feature of this method is that the right-hand side only enters in STEP 5, i.e., after computing the Gröbner basis. Solving the integer linear program with any given right-hand side then amounts to a single generalized division. In other words, after computing the (reduced) Gröbner basis corresponding to (P) only once, each evaluation of the value function

$$v(s) = \min\{\tilde{q}y : Wy \geq s,\ y \in \mathbb{Z}_+^m\}$$

is cheap!

The computation of Gröbner bases is however of exponential complexity and solving large instances of integer programs using this approach is far from today's possibilities. Nevertheless, for problems of moderate size Gröbner bases can be found (see [13]). They contain the essential information to efficiently organize the repeated solution of (P) for varying right-hand side. To our knowledge, no other method can supply comparable information.

To prepare computational experiments on which we will report in Section 7, we computed the reduced Gröbner basis corresponding to the following knapsack problem:

$$\begin{aligned}\max\{16y_1 + 19y_2 + 23y_3 + 28y_4 \ :& \\ & 2y_1 + 3y_2 + 4y_3 + 5y_4 \leq s_1, \\ & 6y_1 + \ \ y_2 + 3y_3 + 2y_4 \leq s_2,\ y_l \in \{0,1\},\ l = 1,\ldots,4\}\end{aligned}$$

Algebraic calculations were performed with the general purpose computer algebra package CoCoA (Release 3.0b) [4]. For computation times, this created some overhead since no advantage is taken from specific features that are inherent to our application in integer programming, in particular that the ideal I is generated by binomials. On the other hand, there is active research directed to exploiting these features (see [13, 31, 32, 33]). Any progress along this line can be used directly to speed up or widen applicability of our algorithm for stochastic integer programs.

With CoCoA the reduced Gröbner basis related to the above knapsack problem was found in 0.34 seconds (on a SPARCstation 4 with 110 MHz microSPARC II CPU). The basis consists of 55 elements. Using this basis to solve the knapsack problem for all relevant right-hand sides, i.e., for all 182 integer 2-vectors $s \in [1, 14] \times [0, 12]$, took 4.19 seconds of CPU time.

3 A countable number of function evaluations

In this section we show that the set of points in which evaluation of the objective function is required is countable in case right-hand sides ξ follow a discrete distribution. We first state general assumptions, also known from continuous recourse modelling, that assure that our model is well defined, followed by a condition that allows application of the Gröbner basis method for evaluations of the function Q. Then we present resulting properties of the function Q and the model (1), that lead to a countable number of operations required by our method.

3.1 Assumptions

We assume that

(i) For any $s \in \mathbb{R}^p$ there exists a $y \in \mathbb{Z}^m_+$ such that $Wy \geq s$.

(ii) There exists a $u \in \mathbb{R}^p_+$ such that $W'u \leq \tilde{q}$.

(iii) The random vector ξ has finite first moment.

Assumption (i) says that, for any possible value of $Tx - \xi$, there exists a feasible second-stage (or recourse) decision y. Following the continuous-recourse terminology, we say that (1) has complete integer recourse. By assumption (ii), the dual to the continuous relaxation of (3) has a feasible point. Therefore, (i) and (ii) together imply that $v(Tx - \xi) \in \mathbb{R}$, for all $x \in \mathbb{R}^n$ and all $\xi \in \mathbb{R}^p$ (Proposition I.6.7. in [20]). Moreover, there exist constants $a_1, a_2 \in \mathbb{R}$ such that for all $s_1, s_2 \in \mathbb{R}^p$

$$|v(s_1) - v(s_2)| \leq a_1||s_1 - s_2|| + a_2$$

(Theorem 8.1, [1]; Theorem 2.1, [2]). Therefore, assumptions (i) - (iii) imply that $Q(x) \in \mathbb{R}$ for all $x \in \mathbb{R}^n$, and (1) is well-defined.

The model (1) is a special case of the mixed-integer recourse model studied in [26]. As a consequence of Proposition 3.1 in [26] we obtain

Lemma 3.1 *Assume (i) - (iii). Then Q is a real-valued lower semicontinuous function on $\mathbb{R}^n$, i.e. $\liminf_{x \to x_o} Q(x) \geq Q(x_o)$ for all $x_o \in \mathbb{R}^n$.*

We assume throughout the paper:

(iv) The random vector ξ follows a discrete distribution with finite support Ξ, say $\Xi = \{\xi^1, \xi^2, \dots \xi^r\}$ and $p^i = \Pr(\xi = \xi^i)$.

Due to this assumption the expected value function Q (and hence the objective function of (1)) is discontinuous, as we will see in the following subsection.

Thus, our results below do not apply to models with continuously distributed ξ. However, it is shown in [26] that, under mild assumptions, local optimal values and sets of local optimal solutions to (1) behave stable if the distribution of ξ is perturbed with respect to the topology of weak convergence of probability measures (Proposition 4.1, [26]). Therefore, it is possible to resort to discrete distributions of ξ when solving (1). If ξ has a continuous distribution then, according to the mentioned stability result, solutions to (1) can be approximated with any given accuracy if a discrete distribution is taken for ξ that is sufficiently close to the original one in the topology of weak convergence.

To be able to apply the Gröbner basis algorithm for function evaluations we assume:

(v) All elements of W are integers.

Actually, it is sufficient if W is rational. Integrality is then obtained by scaling.

Moreover, assumption (v) serves to facilitate specification of the sets where the function Q is constant, as discussed in the following subsection.

3.2 Countability

To establish countability of the number of function evaluations required under the above assumptions, we first analyze the structure of the expected value function. In the following, $\lceil \cdot \rceil$ and $\lfloor \cdot \rfloor$ denote (componentwise) integer round up and round down, respectively.

For all non-negative integer vectors y, $Wy \geq t$ implies $Wy \geq \lceil t \rceil$. Therefore, the second-stage value function v is constant on subsets

$$\{s \in \mathbb{R}^p : \lceil s \rceil = k\} = \{s : k - (1, \dots, 1)' < s \leq k\} \qquad \forall k \in \mathbb{Z}^p,$$

and the function Q is constant on intersections of such subsets. For every $\bar{x} \in \mathbb{R}^n$, the function Q is constant on

$$\begin{aligned} C(\bar{x}) &= \bigcap_{i=1}^{r} \left\{x : \lceil Tx - \xi^i \rceil = \lceil T\bar{x} - \xi^i \rceil \right\} \\ &= \bigcap_{i=1}^{r} \bigcap_{j=1}^{p} \left\{x : \lceil T_j x - \xi^i_j \rceil = \lceil T_j \bar{x} - \xi^i_j \rceil \right\} \\ &= \bigcap_{i=1}^{r} \bigcap_{j=1}^{p} \left\{x : \lceil T_j \bar{x} - \xi^i_j \rceil + \xi^i_j - 1 < T_j x \leq \lceil T_j \bar{x} - \xi^i_j \rceil + \xi^i_j \right\}. \qquad (4) \end{aligned}$$

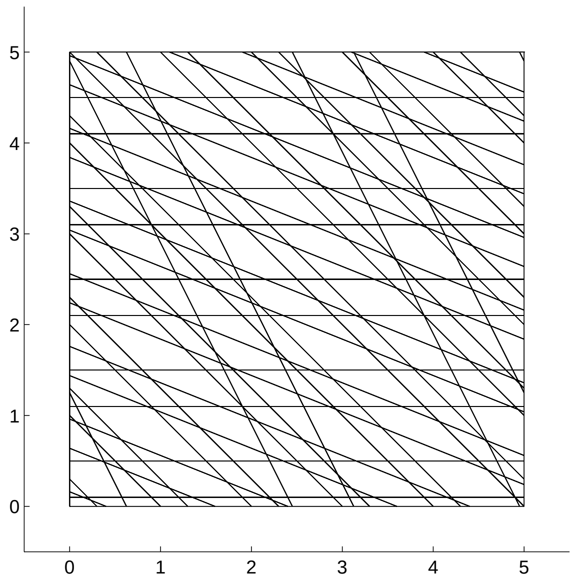

Figure 1: Example of the partition of $[0,5] \times [0,5]$ *in sets* $C(\cdot)$ *if* T *and* Ξ *are given by (5).*

Here T_j is the jth row of the matrix T and ξ^i_j is the jth component of the ith vector in the support of ξ.

From (4) we see that every set $C(\cdot)$ is obtained by intersecting rp sets of the form $\{x : k^i_j + \langle \xi^i_j \rangle - 1 < T_j x \leq k^i_j + \langle \xi^i_j \rangle\}$, where $k^i_j \in \mathbb{Z}$ and $\langle s \rangle = s - \lfloor s \rfloor$ denotes the fractional part of $s \in \mathbb{R}$. Using this structure, in principle we can construct a partition of the feasible set $C = \{x \in \mathbb{R}^n_+ : Ax \geq b\}$ in sets where the expected value function Q is constant. See Figure 1 for an example with

$$T = \begin{pmatrix} .4 & .2 \\ 1 & 1 \\ .5 & 1.25 \\ 0 & 1 \end{pmatrix} \qquad \Xi = \left\{ \begin{pmatrix} .25 \\ .3 \\ .2 \\ .1 \end{pmatrix}, \begin{pmatrix} .98 \\ 0 \\ .8 \\ .5 \end{pmatrix} \right\}. \tag{5}$$

Notice that, since each of the constituting sets is the intersection of an open and a closed half-space, in general the sets $C(\cdot)$ are neither open nor closed.

Below we will show how the fact that Q is constant on every set $C(\cdot) \cap C$ can be used to locate an optimal solution of (1), at least if such sets have vertices. The following condition guarantees this. By 0^+C we denote the recession cone of the convex polyhedron C, i.e., the set of all directions $w \in \mathbb{R}^n$ such that $x + tw \in C$ for some $x \in C$ and all $t \geq 0$. If

$$0^+C \cap \{x : Tx = 0\} = \{0\} \tag{6}$$

then each of the sets $C(\cdot) \cap C$ is bounded and has vertices.

In the next section we will show that, under a mild additional assumption, we can restrict the search for optimal solutions to a bounded subset of C. In that case existence of vertices is no longer a question and the condition (6) can be dropped.

In the following theorem we show that the countable set of all vertices of the sets $C(\cdot) \cap C$ contains an optimal solution of (1).

Definition 3.1 The countable set V, given by

$$V = \{x \in \mathbb{R}^n : x \text{ is a vertex of } C(x) \cap C\},$$

is called the *set of candidates*; an element of V is called a *candidate point.*

Example 3.1 Consider an integer recourse model with feasible region $C = \{x \in \mathbb{R}^2 : 0 \le x_j \le 5,\ j = 1,2\}$, technology matrix $T = \binom{1\,0}{0\,1}$ and random right-hand side vector ξ with support $\{5,\ 5.5, \dots,\ 15\} \times \{5,\ 5.5, \dots,\ 15\}$. Then, for $\bar{x} \in \mathbb{R}^2$,

$$C(\bar{x}) \quad = \quad \bigcap_{i=1}^{121} \left\{ x : \begin{array}{l} k_1^i(\bar{x}) + \langle \xi_1^i \rangle - 1 < x_1 \le k_1^i(\bar{x}) + \langle \xi_1^i \rangle \\ k_2^i(\bar{x}) + \langle \xi_2^i \rangle - 1 < x_2 \le k_2^i(\bar{x}) + \langle \xi_2^i \rangle \end{array} \right\},$$

where $k_j^i(\bar{x}) = \lceil \bar{x}_j - \xi_j^i \rceil + \lfloor \xi_j^i \rfloor$ is an integer for all i and j. Since $\langle \xi_j^i \rangle \in \{0, 1/2\}$, $j = 1, 2$, we see that x is a vertex of some set $C(\cdot)$ if and only if $x = (k_1/2, k_2/2)$, $k_1, k_2 \in \mathbb{Z}$. Hence in this case the set of candidates is

$$V = \left\{ x \in \mathbb{R}^2 : \begin{array}{l} x_j = k_j/2,\ k_j \in \mathbb{Z} \\ 0 \le x_j \le 5 \end{array},\ j = 1, 2 \right\},$$

which has cardinality 121. $\triangleleft$

Theorem 3.1 *Let V, the set of candidates, be non-empty. If* $\operatorname{argmin}\{cx + Q(x) : x \in C\} \neq \emptyset$ *then*

$$V \cap \operatorname{argmin}\{cx + Q(x) : x \in C\} \neq \emptyset.$$

Proof. Let $\bar{x} \in \operatorname{argmin}\{cx + Q(x) : x \in C\}$. For all $x \in C(\bar{x}) \cap C$ we have $Q(x) = Q(\bar{x})$. Consider minimizing the linear function $cx + Q(\bar{x})$ on the closure of $C(\bar{x}) \cap C$, denoted by $\operatorname{cl}(C(\bar{x}) \cap C)$. Since the minimum over this polyhedral set is attained, it is attained in one of its vertices, say $\hat{x}$. If $\hat{x} \in C(\bar{x}) \cap C$ we are done, since in that case $Q(\hat{x}) = Q(\bar{x})$ and $c\hat{x} \le c\bar{x}$, implying that the vertex $\hat{x}$ is an optimal solution. Otherwise, if $\hat{x} \in \operatorname{cl}(C(\bar{x}) \cap C) \setminus C(\bar{x}) \cap C$ consider the set $C(\hat{x}) \cap C$ which trivially contains $\hat{x}$ as a vertex. It holds

$$Q(\hat{x}) \le \lim_{x \to \hat{x}} Q(x) = Q(\bar{x}) \qquad \forall x \in C(\bar{x}) \cap C,$$

where the inequality is valid by the lower semicontinuity of Q. Since also $c\hat{x} \le c\bar{x}$, it follows that $\hat{x} \in V$ is an optimal solution. $\square$

Thus, in order to find an optimal solution of (1), it is sufficient to consider only elements of the countable set V. In the next section we present conditions that allow to define a finite subset of V that still contains all optimal candidate points. Notice that finiteness of V itself would be implied directly by boundedness of C.

4 Finiteness using level sets

The purpose of this section is to show how the continuous relaxation of (1) can be used to reduce the set of candidates defined in the previous section to a finite set. The continuous relaxation is obtained by dropping the integrality conditions on the second-stage variables in (3):

$$\min\{cx + Q_R(x)\ :\ x \in C\} \tag{7}$$

where

$$Q_R(x) = E_\xi\ v_R(Tx - \xi)$$

and

$$v_R(s) = \min\{\tilde{q}y : Wy \ge s,\ y \in \mathbb{R}_+^m\}.$$

By (i) - (iii), the problem (7) is well defined. Its optimal value is obviously a lower bound to the optimal value of (1).

The following result is the basic tool that allows the use of the continuous relaxation to restrict the set of candidate points.

Lemma 4.1 *Let X be a non-empty set, and f and $\bar{f}$ real functions on X such that $\bar{f}(x) \leq f(x)$ for all $x \in X$. Then, for all $\bar{x} \in X$,*

$$\operatorname*{argmin}_{x \in X} f(x) \subset \{x \in X : \bar{f}(x) \leq f(\bar{x})\}.$$

Moreover, the difference between these sets is smaller according as $\bar{f}$ is a better approximation of f and $f(\bar{x})$ is a better approximation of $\inf_{x\in X} f(x)$. In particular, if $\bar{f}(\bar{x}) = f(\bar{x})$ and $\bar{x} \in \operatorname{argmin}_{x\in X} \bar{f}(x)$ then $\bar{x} \in \operatorname{argmin}_{x\in X} f(x)$.

Proof. For any $\bar{x} \in X$

$$\begin{aligned} \operatorname*{argmin}_{x \in X} f(x) &= \bigcap_{y \in X} \{x \in X : f(x) \leq f(y)\} \\ &\subset \bigcap_{y \in X} \{x \in X : \bar{f}(x) \leq f(y)\} \\ &\subset \{x \in X : \bar{f}(x) \leq f(\bar{x})\}, \end{aligned}$$

where the tightness of each inclusion clearly depends on the indicated properties of f and $\bar{x}$, respectively.

The last claim follows directly from the assumptions. We have $f(\bar{x}) = \bar{f}(\bar{x}) \leq \bar{f}(x) \leq f(x)$ for all $x \in X$, which precisely means that $\bar{x} \in \operatorname{argmin}_{x\in X} f(x)$. □

Since Q_R is a lower bound for Q on $\mathbb{R}^n$, this lemma implies that for any feasible $\bar{x}$ the corresponding level set of the objective of the continuous relaxation, given by

$$L(c\bar{x} + Q(\bar{x})) = \{x \in C : cx + Q_R(x) \geq c\bar{x} + Q(\bar{x})\},$$

contains all minimizers of the integer recourse problem (1). Moreover, each time a feasible point with a lower objective value is found, the level set can be reduced, thus decreasing the number of points that have to be enumerated even further.

It is clear now that if there is a bounded level set then an optimal solution of (1) is contained in the finite intersection of this level set and the set of candidates. To arrive at conditions under which this is the case, we first review a well-known dual representation of the function Q_R, to be used in the subsequent discussion of a (partial) description of the level sets $L(\cdot)$. In the next section, where we discuss how to actually enumerate the set of candidates, these level sets also play an important role.

By linear programming duality, we obtain

$$v_R(s) = \max\{su : W'u \leq \tilde{q}, u \in \mathbb{R}^p_+\}.$$

Assumptions (i) - (ii) together imply that the set $M_D = \{u \in \mathbb{R}^p_+ : W'u \leq \tilde{q}\}$ is a non-empty compact polyhedron. Denoting its vertices by $d_l,\ l \in N_o$, where N_o is a finite index set, we obtain

$$v_R(s) = \max_{l \in N_o} d_l s.$$

Hence

$$Q_R(x) = \sum_{i=1}^{r} p^i v_R(Tx - \xi^i) = \sum_{i=1}^{r} p^i \max_{l \in N_o} d_l(Tx - \xi^i),$$

from which it is easy to see the well-known fact that Q_R is a piecewise linear convex function on $\mathbb{R}^n$.

As explained above, approximations of lower level sets of the objective function $cx + Q_R(x)$ are used in our algorithm. These sets are constructed as follows.

Let $N \subset N_o$ denote the index set of a subset of the vertices of M_D, and $\bar{\xi} = \sum_{i=1}^r p^i \xi^i$. Then the function

$$\bar{Q}_R(x) = \max_{l \in N} d_l(Tx - \bar{\xi})$$

is a lower bound for Q_R. Indeed, by the convexity of Q_R and Jensen's inequality, we have for all $x \in \mathbb{R}^n$

$$\begin{aligned} Q_R(x) &\geq v_R(Tx - \bar{\xi}) \\ &= \max_{l \in N_o} d_l(Tx - \bar{\xi}) \\ &\geq \max_{l \in N} d_l(Tx - \bar{\xi}) \\ &= \bar{Q}_R(x). \end{aligned}$$

Thus, a collection of (outer) approximations of the lower level set

$$L(\alpha) = \{x \in C : cx + Q_R(x) \leq \alpha\} \tag{8}$$

is given by

$$\bar{L}_N(\alpha) = \{x \in C : cx + d_l(Tx - \bar{\xi}) \leq \alpha,\ l \in N\}, \quad N \subset N_o. \tag{9}$$

Obviously $\bar{L}_M(\alpha) \subset \bar{L}_N(\alpha)$ if $M \supset N$.

For problems of very moderate size, the complete list of vertices of M_D can be obtained via stochastic programming pre-processing techniques as in [14, 36], or by general vertex enumeration methods (cf. [6] for a comfortable implementation). In general, however, one has to live with a partial list of vertices. Algorithms for (non-integer) stochastic programs like the regularized decomposition method [23] yield such a list in the course of computation. For this reason the first step of our algorithm will be to solve the continuous relaxation of (1), so that we can assume that at least a partial list of vertices of M_D is available.

It is clear that to obtain finiteness of our method we need to restrict the enumeration to a finite number of candidate points. Due to Lemma 4.1 finiteness is guaranteed if, for some $N \subset N_o$ and $\alpha \in \mathbb{R}$, the (approximate) level set $\bar{L}_N(\alpha)$ is bounded and non-empty. By Corollary 8.7.1 in [22] it then follows that $\bar{L}_N(\alpha)$ is bounded for every α, and by construction the same is true for $\bar{L}_M(\alpha)$, $M \supset N$. In particular, it holds that if the solution set of the continuous relaxation (which is a level set, say $\bar{L}_{N_o}(\alpha^*)$) is bounded and non-empty, then $\bar{L}_{N_o}(\alpha)$ is bounded and non-empty for all $\alpha \geq \alpha^*$. Using (9) we see that boundedness of this set is implied by

$$\{w \in 0^+C \ : \ (c + d_l T)w \leq 0,\ l \in N_o\} = \{0\}.$$

However, in the description of this condition some constraints may be redundant. In that case we only need a subset of $|N| < |N_o|$ vertices of the dual feasible region to construct a bounded approximate level set $\bar{L}_N(\alpha)$.

In order to avoid superfluous notational burden, in the sequel we will simply use L to denote an (approximate) level set whenever no confusion can arise.

5 Enumerating the set of candidates

In the previous sections we have shown that the set of candidates V, intersected with a level set L of the continuous relaxation, contains an optimal solution of (1). To obtain an optimal solution we need to completely enumerate this set, which is finite by assumption. In this section we show how such a complete enumeration can be organized in a way that takes advantage of the structure of the set of candidates.

By definition every candidate point $v \in V$ is a vertex of $C(v) \cap C$, which can be represented as

$$\left\{ x \in \mathbb{R}^n : \begin{array}{l} k_j^i + \langle \xi_j^i \rangle - 1 < T_j x \le k_j^i + \langle \xi_j^i \rangle, \ i = 1 \ldots r, \ j = 1 \ldots p \\ Ax \ge b, x \ge 0 \end{array} \right\}, \tag{10}$$

where $k_j^i = \lceil T_j v - \xi_j^i \rceil$, $i = 1 \ldots r$, $j = 1 \ldots p$. Since v is a vertex of this set, it satisfies n independent inequalities from this system with equality.

Remark 5.1 By Lemma 4.1 we only need to consider candidate points that are contained in the level set L. However, the inequalities defining L are not represented in (10), since their role differs from the inequalities in terms of the rows of T and A, and the non-negativities. Indeed, a vertex of $C(\cdot) \cap L$ that is not also a vertex of $C(\cdot) \cap C$ is not a candidate point as defined in Definition 3.1, and therefore need not be evaluated.

We can obtain a complete list of all candidate points in L by considering all choices for $k_j^i \in \mathbb{Z}$ such that $\{x : T_j x = k_j^i + \langle \xi_j^i \rangle\} \cap L$ is non-empty, and for every such choice considering all combinations of n independent equalities. This idea is the basis of our enumeration method. As will be explained in Section 6, it appears to be beneficial to consider subsets of candidate points that all lie on a common line segment defined by $n-1$ out of n equality constraints as mentioned above. On such a line segment, groups of candidate points are determined by one more independent equality whose right-hand side is varied. In order to give a detailed description of our enumeration method, we first need to introduce some notation.

For $j = 1, \ldots, p$, we define

$$\begin{aligned} t_j^u &= \max\{T_j x : x \in L\} \\ t_j^l &= \min\{T_j x : x \in L\}, \end{aligned}$$

and

$$R_j = \bigcup_{i=1}^{r} \left\{ k + \langle \xi_j^i \rangle : k \in \mathbb{Z}, \ t_j^l \le k + \langle \xi_j^i \rangle \le t_j^u \right\}.$$

Each set R_j contains all right-hand side values such that the inequality $T_j x \le r_j$, $r_j \in R_j$, may appear in the description (10) of some set $C(\cdot)$ that has a non-empty intersection with L. To enable uniform treatment of all (in)equalities in (10), we also define $R_j = \{b_{j-p}\}$, $j = p+1, \ldots, p+q$, and $R_j = \{0\}$, $j = p+q+1, \ldots, p+q+n$, to denote the singleton sets of possible right-hand side values for the inequalities defining the set $C = \{x \in \mathbb{R}^n : Ax \ge b, Ix \ge 0\}$, where I is the $n \times n$ identity matrix.

Next we define the $(p+q+n) \times n$ matrix $S = (T; A; I)$. Finally, let $J \subset \{1, \ldots, p+q+n\}$, $|J| = n-1$, be an index set such that the matrix S_J, consisting of the rows S_j, $j \in J$, has rank $n-1$, and let $R_J \subset \mathbb{R}^{n-1}$ be the cartesian product of R_j, $j \in J$.

Now we are ready to define the line segments that we have in mind. For a fixed $r \in R_J$, consider

$$H_J(r) = \{x : S_J x = r\} \cap L.$$

Either $H_J(r) = \emptyset$, or it is a line segment whose endpoints are given by

$$\begin{aligned} x^u_J(r) &= \max\{cx : x \in H_J(r)\} \\ x^l_J(r) &= \min\{cx : x \in H_J(r)\}. \end{aligned}$$

(if $c \perp H_J(r)$ any other objective vector can be used). Now consider a row S_j with $j \notin J$ (and S_j not perpendicular to $H_J(r)$). If, for some right-hand side $r_j \in R_j$, the hyperplane $S_j x = r_j$ intersects $H_J(r)$, then this intersection is a candidate point; we will say that this candidate point is *generated on* $H_J(r)$ *by* S_j. To obtain all candidate points on $H_J(r)$ generated by S_j we determine the intersections of $H_J(r)$ with all hyperplanes $S_j x = r_j$, where $r_j \in R_j$ only needs to be considered if r_j is in between $S_j x^u_J(r)$ and $S_j x^l_J(r)$. Repeating this procedure for every row S_j with $j \notin J$, and including $x^u_J(r)$ and/or $x^l_J(r)$ if it is on the boundary of the feasible set C, results in a list of all candidate points on $H_J(r)$.

By repeating the procedure above for every family of parallel line segments, i.e., for every possible subset J and every $r \in R_J$, all candidate points in the current level set L will be found. In fact, since every candidate point is on a line segment $H_J(\cdot)$ for n different sets J, it would be listed n times. This redundancy is easily removed by considering only candidates generated on $H_J(r)$ by rows S_j with $j > \max_J j$.

In Section 6.2 we will present some ideas to reduce the number of candidate points in which function evaluations are required. In particular, we will explain how to take advantage of the way we organized the enumeration.

6 Algorithm

We now have all but one of the ingredients that make up the algorithm to be presented in this section. The last ingredient concerns the determination of an appropriate (initial) level set. For this purpose we could choose any point in the feasible region C, evaluate the objective function in this point, and determine the level set using the continuous relaxation as described in the previous section. Since we need to solve the continuous relaxation to obtain a (partial) description of the level sets anyway, it seems reasonable to use one of its optimal solutions, say x^R, as initial point and $L(cx^R + Q(x^R))$ as the initial level set.

6.1 Basic form of the algorithm

The presentation of the algorithm is merely a summary of the ingredients exposed in the preceding sections. The algorithm consists of the following parts.

1. Compute a Gröbner basis for the second stage integer linear programming problem as explained in Section 2.

2. Solve the continuous relaxation (7) and obtain a (partial) list of vertices of the dual feasible region. Let x^R be an optimal solution.

3. Compute the objective value $cx^R + Q(x^R)$ (using the Gröbner basis), and construct the (partial) level set $L(cx^R + Q(x^R))$.

4. For every candidate point in $L(cx^R + Q(x^R))$ evaluate the objective function, using the Gröbner basis to compute the expected value function Q. The candidate points are enumerated according to the scheme proposed in Section 5. A candidate with smallest function value is an optimal solution to the problem.

Under our assumption that the set $L(cx^R + Q(x^R))$ is bounded, the number of candidate points to be evaluated in step 4 is finite. Therefore, the algorithm determines an optimal solution of (1) in finite time.

6.2 Improvements of the algorithm

In this section we propose two improvements of the algorithm presented above. They both aim at reducing the number of candidate points for which the objective function has to be evaluated. In general, this number can be enormous. Moreover, every function evaluation takes r evaluations of the second stage value function v, where we recall that r is the number of mass points in the support Ξ. Thus, although each evaluation of v by means of the Gröbner basis is cheap, it is still worthwhile to try to minimize the number of function evaluations needed.

The first idea relates to the use of the level set L. By Lemma 4.1 all minimizers of (1) are contained in every level set $L(c\bar{x} + Q(\bar{x}))$ for every feasible solution $\bar{x}$. Since we only need to evaluate candidate points that are in the level set, it seems to be advantageous to choose the level set as small as possible. This can be implemented as follows. Recall that the initial level set is $L(cx^R + Q(x^R))$, where x^R is an optimal solution of the continuous relaxation (7). As before, we start evaluating function values in candidate points according to the enumeration scheme presented in Section 5. However, as soon as we find a candidate with a lower objective value than x^R, say $\hat{x}$, we can use it to reduce the level set to $L(c\hat{x} + Q(\hat{x}))$. This is of course a subset of $L(cx^R + Q(x^R))$, so that in general the number of remaining candidates is decreased. Clearly, this procedure can be repeated each time a lower function value is obtained.

Thus, repeatedly updating the level set has the benefit of reducing the number of candidates to be evaluated. However, this benefit should be set off against the additional work that the updating brings about. Updating the level set itself comes virtually free, since only the right-hand sides in (9) are changed. Considering the enumeration scheme, only the sets R_j, $j = 1, \dots, p$, have to be updated. For each j, this boils down to recomputing t_j^u and t_j^l by solving an LP problem of size $(N+q) \times n$, where N is the number of inequalities in the polyhedral description of the (outer approximation of the) level set, and q is the number of rows of the matrix A. Our tentative conclusion is that it seems to be beneficial to update the level set, either every time a better solution is found or only if a significantly lower objective value is found. See Section 7 for some first results.

The second improvement motivates the organization of the complete enumeration as presented in Section 5. Based on the identification of common directions of increase of the first-stage objective cx and the expected value function $Q(x)$, we show that certain groups of candidate points can not be optimal solutions.

Lemma 6.1 *Let $H_J(r)$, $r \in R_J$, be a line segment with endpoints $x_J^l(r)$ and $x_J^u(r)$ as defined above. Let the candidate point v, not equal to $x_J^l(r)$ or $x_J^u(r)$, be generated on $H_J(r)$ only by rows T_j, $j \in I \subset \{1, \dots, p\}$. Assume that*

(i) $c(x_J^u(r) - x_J^l(r)) > 0$,

(ii) $T_j(x_J^u(r) - x_J^l(r)) > 0$ *for all* $j \in I$.

Then v is not an optimal solution of the integer recourse problem (1).

Proof. Let $\bar{v}$ be the neighboring candidate point of v on $H_J(r)$ in the direction $-(x_J^u(r) - x_J^l(r))$, and for $\lambda \in (0, 1)$ define $x_\lambda = \lambda\bar{v} + (1 - \lambda)v$ (see Figure 2). It is sufficient to show that for λ small enough $x_\lambda \in C(v)$, so that $Q(x_\lambda) = Q(v)$. Since $cx_\lambda < cv$ by assumption (i), this proves the result.

Using (4) we have

$$C(v) = \bigcap_{j=1}^{p} C_j(v)$$

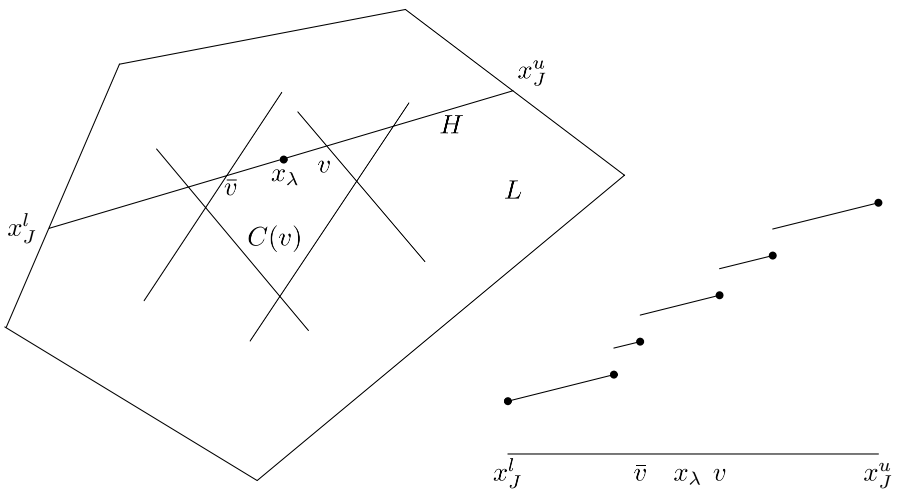

Figure 2: Illustrations by the proof of Lemma 6.1. The right figure shows the function $cx + Q(x)$ on H; the function value in candidate points is depicted by a $\bullet$.

where

$$C_j(v) = \bigcap_{i=1}^{r} \left\{ x : \lceil T_j v - \xi_j^i \rceil - 1 < T_j x - \xi_j^i \leq \lceil T_j v - \xi_j^i \rceil \right\}.$$

Depending on the different roles that a row T_j may play, we distinguish three cases and each time show that $x_\lambda \in C_j(v)$.

If $j \in J$ then $T_j v = T_j \bar{v} = T_j x_\lambda$, so that trivially $x_\lambda \in C_j(v)$.

If $j \in I$ then by assumption (ii) we have $T_j \bar{v} < T_j x_\lambda < T_j v$. Since $\bar{v}$ is a neighbor of v we have $T_j \bar{v} - \xi_j^i \geq \lceil T_j v - \xi_j^i \rceil - 1$ for all i. Hence $\lceil T_j v - \xi_j^i \rceil - 1 < T_j x_\lambda - \xi_j^i < \lceil T_j v - \xi_j^i \rceil$, which means that $x_\lambda \in C_j(v)$.

Finally, if $j \in \{1, 2, \ldots, p\} \setminus \{I \cup J\}$ then $\lceil T_j v - \xi_j^i \rceil - 1 < T_j v - \xi_j^i < \lceil T_j v - \xi_j^i \rceil$ for all i, where the second strict inequality follows by construction. Hence for λ small enough it also holds $\lceil T_j v - \xi_j^i \rceil - 1 < T_j x_\lambda - \xi_j^i < \lceil T_j v - \xi_j^i \rceil$ for all i, so that again $x_\lambda \in C_j(v)$. $\square$

In our algorithm we may use Lemma 6.1 as follows. Consider a family of line segments $H_J(r)$, $r \in R_J$. If for some $r \in R_J$ it holds that $c(x_J^u(r) - x_J^l(r)) > 0$, then this is true for every $r \in R_J$ since all these line segments are parallel. Similarly, if a row T_j, $j \notin J$, satisfies $T_j(x_J^u(r) - x_J^l(r)) > 0$ for one $r \in R_J$, then this inequality holds for all $r \in R_J$. Therefore, given a family of line segments $H_J(r)$, $r \in R_J$, and a row T_j, $j \notin J$, we check for an arbitrary $r \in R_J$ if both $c(x_J^u(r) - x_J^l(r)) > 0$ and $T_j(x_J^u(r) - x_J^l(r)) > 0$. If these conditions are satisfied, we may skip all candidate points that are generated by this row on any line segment $H_J(r)$, $r \in R_J$, since by Lemma 6.1 none of them can be optimal for (1) if they are generated only by this row T_j. Note that any candidate point that is also generated by a row T_i, $i \notin J$, $i \neq j$, such that $T_i(x_J^u(r) - x_J^l(r)) \leq 0$ will still be considered; hence, no possible optimal solutions will be discarded.

This improvement is very cheap to implement and may result in a significant reduction of the number of function evaluations needed.

7 Computational Results

In this section we provide some results to indicate computational possibilities of our algorithm and its improvements. All computations are done on a SPARCstation 4 with 110 MHz microSPARC II CPU. Subsequent examples all fit into the following frame

$$\max\{\frac{3}{2}x_1 + 4x_2 + Q(x) \ : \ x \in C\} \tag{11}$$

where

$$Q(x) = E_\xi v(\xi - Tx)$$

and

$$\begin{aligned} v(s) = \max\{16y_1 + 19y_2 + 23y_3 + 28y_4 \ : \\ &2y_1 + 3y_2 + 4y_3 + 5y_4 \le s_1, \\ &6y_1 + \ \ y_2 + 3y_3 + 2y_4 \le s_2, \ y_l \in \{0,1\}, l = 1, \ldots, 4\}. \end{aligned}$$

The random vector ξ has a uniform discrete distribution on $\{5, \ 5.5, \ldots, \ 15\} \times \{5, \ 5.5, \ldots, \ 15\}$, so that the support Ξ of ξ has cardinality 441 and $p_i = 1/441$ for $i = 1, \ldots, 441$. To test our algorithm we will create instances of (11) by specifying the set $C \subset \mathbb{R}^2$ and the 2×2-matrix T, respectively. The second stage of (11) coincides with the knapsack problem for which we computed a Gröbner basis in Section 2.

Our test problem (11) can be interpreted as a two-stage investment problem. Its first-stage decision x has to be selected from the set C and yields an immediate revenue $\frac{3}{2}x_1 + 4x_2$. Further revenue is gained from projects for which investment is done in the second stage after having observed the random vector $\xi \in \mathbb{R}^2$ leading to the budget $\xi - Tx$. Entries in T are always non-negative so that spending money in the first stage decreases possibilities in the second stage. We will also consider an instance where negative x_1, x_2 are permitted, in this way modelling the possibility to contract loans in the first stage to enlarge possibilities in the second stage.

For the feasible set C we will consider two instances, $C = \{x \in \mathbb{R}^2 : 0 \le x_i \le 5, i = 1, 2\}$ and $C = \{x \in \mathbb{R}^2 : x_i \le 5, i = 1, 2\}$. For the matrix T we will consider $T = \binom{1\,0}{0\,1}$ and $T = \begin{pmatrix} 2/3 & 1/3 \\ 1/3 & 2/3 \end{pmatrix}$. Since $\Xi \subset \{\xi \in \mathbb{R}^2 : 5 \le \xi_j \le 15, j = 1, 2\}$ we then have $\xi^i - Tx \ge 0$ for all outcomes ξ^i and all $x \in C$ in either instance of C. Therefore, $y = 0$ is always a feasible second-stage decision, and we end up with $Q(x) \in \mathbb{R}$ for all feasible x in either instance of C. Although weaker than Assumption (i) in Section 3, this *relatively complete recourse* property is sufficiently strong from the viewpoint of numerical testing, since our algorithm works with feasible points only.

Because of its clear interpretation, we selected the maximization problem (11) to present numerical results. Although we presented our theoretical results in a minimization setting, this provides no hardship since transformation of the relevant statements is straightforward and will not be discussed here.

Let us now investigate computational possibilities of our algorithm for various instances of (11). In all cases function values of the expected value function Q are computed using the formula

$$Q(x) = \frac{1}{441} \sum_{i=1}^{441} v(\lfloor \xi^i - Tx \rfloor). \tag{12}$$

Here, function values of v are taken from a list generated in advance by computing the Gröbner basis and performing all relevant reductions (cf. Section 2). Again we stress that our algorithm in no way depends on the method used to compute these function evaluations. For example, in the case of a second-stage problem whose size is prohibitive for Gröbner basis computations, it could still be possible to use a standard (mixed-)integer solver if the right-hand side parameters ξ can only attain a few distinct values.

Example 7.1 Let $C = \{x \in \mathbb{R}^2 : 0 \le x_i \le 5, i = 1, 2\}$ and $T = \binom{1\,0}{0\,1}$.
As mentioned in Section 3, the set V of candidate points then equals the finite set $\{(k_1/2, k_2/2) : k_1, k_2 \in \mathbb{Z}\} \cap C$, which has cardinality 121. Complete enumeration using formula (12) yields the optimal solution $x_1 = 0, x_2 = 4$, with value 61.3288. $\triangleleft$

Example 7.2 Let $C = \{x \in \mathbb{R}^2 : x_i \le 5, i = 1, 2\}$ and $T = \binom{1\,0}{0\,1}$.
Here, the constraint set is unbounded and 'negative investment' in the first stage is possible. Since the set of candidate points is not a priori finite, we employ the results from Section 4 to bound the set of candidate points using level sets of the continuous relaxation. Because the second-stage knapsack problem is of moderate size, all vertices of the dual feasible set M_D of its continuous relaxation can be computed using the vertex enumeration code PORTA ([6]). After less than 0.01 seconds of CPU time a list of 13 vertices was found. From the list we extracted the vertices

$$\begin{aligned} d_1 &= (0, \frac{8}{3}, 0, \frac{49}{3}, 15, \frac{68}{3}) \\ d_2 &= (0, \frac{23}{3}, 0, \frac{34}{3}, 0, \frac{38}{3}) \\ d_3 &= (\frac{28}{5}, 0, \frac{24}{5}, \frac{11}{5}, \frac{3}{5}, 0) \end{aligned}$$

and formed an outer approximation of the upper level set $L(\alpha) = \{x \in C : cx + Q_R(x) \ge \alpha\}$ given by

$$\bar{L}(\alpha) = \{x \in C : cx + d_l \begin{pmatrix} \bar{\xi} - Tx \\ e \end{pmatrix} \ge \alpha,\ l = 1, 2, 3\}, \tag{13}$$

where $e = (1\ 1\ 1\ 1)'$ (corresponding to the relaxed binary constraints). Note that since we have a maximization problem the function Q_R provides a concave upper bound and the set (13) is an upper level set.

Using formula (12) we find for $x = 0$ the objective function value 55.2517. This provides the initial level set $\bar{L}(55.2517)$ which turns out to be bounded (this check as well as the extraction of proper vertices of M_D are still done by hand in this initial phase of numerical testing). Table 1 reports progress of our algorithm for the present example. Choosing $J = \{2\}$, enumeration of candidate points was performed on some of the line segments

$$\begin{aligned} H_{\{2\}}(r) &= \{x \in \mathbb{R}^2 : T_2 x = r\} \cap \bar{L}(\alpha) \\ &= \{x \in \mathbb{R}^2 : x_2 = r\} \cap \bar{L}(\alpha), \qquad r \in R_{\{2\}}, \end{aligned}$$

where $R_{\{2\}} = \{-9.5, -9, \ldots, 5\}$ since $t_2^l = -9.9178$ and $t_2^u = 5$.

Enumerations started with $x_2 = 0$ followed by $x_2 = 0.5, 1, 1.5, 2$. Table 1 shows that points with increased objective function values were found during each enumeration along such a line segment. This would have permitted updates of $\bar{L}(\alpha)$ after each enumeration. We decided to update only after enumerations with $x_2 = 0$ and $x_2 = 2$. The impact on the number of candidate points is shown in the fourth column. After having found the point $(-4, 2)$ we ended up with a level set containing 427 candidate points of which 125 were already evaluated in previous enumerations.

x_2	α	# candidate points in $\{\mathbb{R} \times \{x_2\}\} \cap \bar{L}(\alpha)$	# candidate points in $\bar{L}(\alpha)$	current best point	current best value
0	55.2517	38	872	(-5,0)	62.2959
0.5	62.2959	27	521	(-5,0.5)	62.7744
1	62.2959	29	521	(-3.5,1)	63.2528
1.5	62.2959	31	521	(-4,1.5)	63.8163
2	62.2959	33	521	(-4,2)	64.4218
–	64.4218	–	427	–	–

Table 1: Numerical results for Example 7.2.

Complete enumeration of the remaining 302 points did not result in an improvement of the objective function value so that $(-4, 2)$ turned out to be optimal. (Note that here 'negative investment' in x_1 really paid!) Altogether a total of 460 candidate points had to be inspected. We conclude that in this case reduction of the level set yields a reduction of the number of objective function evaluations to almost 50%.

Adapting Lemma 6.1 to the maximization setting of our example, leads to possible elimination of candidate points along joint directions of decrease of cx and $Q(x)$. The function Q decreases along any direction in which $Tx = x$ increases, i.e., along any direction in $\mathbb{R}^2_+$. The intersection of $\mathbb{R}^2_+$ with directions of decrease of $cx = \frac{3}{2}x_1 + 4x_2$ is obviously $\{0\}$, so that in this case Lemma 6.1 does not yield any reduction of the number of candidate points. $\lhd$

Example 7.3 Let $C = \{x \in \mathbb{R}^2 : 0 \leq x_i \leq 5,\ i = 1, 2\}$ and $T = \binom{2/3\ 1/3}{1/3\ 2/3}$. The purpose of this example is to illustrate the effect of a priori elimination of non-optimal candidate points, as reflected in Lemma 6.1. Since C is bounded, we simply use the level set $L = C$.

Consider the family of line segments $H_{\{2\}}(r),\ r \in R_{\{2\}}$, given by

$$\begin{aligned} H_{\{2\}}(r) &= \{x \in \mathbb{R}^2 : T_2 x = r\} \cap C \\ &= \{x \in \mathbb{R}^2 : x_1 = 3r - 2x_2\} \cap C. \end{aligned}$$

For $x \in H_{\{2\}}(r)$ we have $cx = 9/2r + x_2$ and $T_1 x = 2r - x_2$, from which we see that cx is strictly decreasing on $H_{\{2\}}(r)$ along the direction where $T_1 x$ is strictly increasing. By Lemma 6.1 (adapted to the maximization setting at hand) it follows that, for each $r \in R_{\{2\}}$, only the endpoint of $H_{\{2\}}(r)$ with minimal value of $T_1 x$ can be an optimal solution. All other candidate points generated by T_1 on $H_{\{2\}}(r)$ can be eliminated right away.

Every intersection of lines in Figure 3 depicts a candidate point for the present example. Beside the points mentioned above, only those not on a line defined by $T_2 x = r,\ r \in R_{\{2\}} \subset \{k/2 : k \in \mathbb{Z}\}$, have to be inspected. In this way, enumeration was restricted to 19 out of 53 candidate points. Computing objective function values with the help of formula (12), the point $x = (0, 4.5)$ was identified as optimal solution (objective function value 61.4444). $\lhd$

The examples above indicate proper performance of the algorithm provided that the many computations of the second-stage value function, using a Gröbner basis or any other method, are computationally feasible. This is mainly a question of problem size. Despite this limitation the algorithm widens our abilities to solve integer stochastic programs. The only alternative deterministic algorithm that existed up to now is to tackle the large-scale mixed-integer equivalent with general purpose mixed-integer LP solvers. This approach already fails for problems still tractable

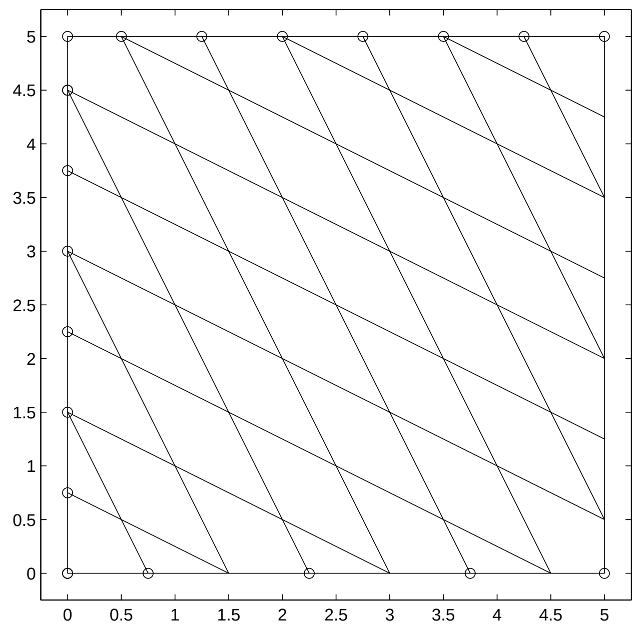

Figure 3: Remaining candidate points (depicted by $\circ$*) after applying Lemma 6.1.*

by our method. For example, the large-scale equivalent to (11) reads

$$
\begin{aligned}
\max\{\frac{3}{2}x_1 + 4x_2 + \frac{1}{441}\sum_{i=1}^{441}(16y_{1i} + 19y_{2i} + 23y_{3i} + 28y_{4i}) \ :\\
T_1x + 2y_{1i} + 3y_{2i} + 4y_{3i} + 5y_{4i} \leq \xi_1^i,\\
T_2x + 6y_{1i} + \ \ y_{2i} + 3y_{3i} + 2y_{4i} \leq \xi_2^i,\\
y_{li} \in \{0,1\},\ l = 1,\ldots,4,\ i = 1,\ldots,441,\ x \in C\}.
\end{aligned}
$$

This problem has 1764 Boolean and 2 continuous variables and 882 constraints (plus the number of constraints represented by C). To the instances specified in Examples 7.1 – 7.3 we applied the CPLEX Mixed-Integer Optimizer (version 4.0) [9]. We imposed an upper limit of 50.000 on the number of nodes in the branching tree. In all three cases we ended up with gaps around 25%. The x-parts of the corresponding feasible points were far off the optimal ones given in the Examples 7.1 – 7.3. In particular, we obtained the following results: For Example 7.1, the first feasible point (gap 24%) found after 1106 nodes was the best available. For Example 7.2, a first feasible point (gap 46%) was found after 350 nodes and the best available (gap 26%) after 32.249 nodes. For Example 7.3 again the first feasible point (gap 27%) found after 1778 nodes was the best available.

8 Concluding remarks

The main contribution in this paper lies in using structural properties of stochastic programming problems with integer recourse to construct a countable c.q. finite set containing an optimal solution. As a result, the integer recourse problem can be solved using an (implicit) enumeration scheme that is guided by the structure of the set of possible optimal solutions. Improvements to speed up the enumeration are proposed and tested on small example problems.

The algorithm presented in Section 6 is one of the first general purpose algorithms devised for two-stage stochastic integer programming. Compared to the

L-shaped method of Laporte and Louveaux [18], our setting is more general since we do not assume integral (binary) first-stage variables to obtain a countable (finite) solution set. The stochastic branch and bound method of Norkin et al. [24] can be applied to our model, but yields a stochastic solution.

At this point we wish to stress the modular framework of our algorithm. In the first place, we propose to use (approximate) level sets of the continuous relaxation (7) in order to obtain finiteness c.q. iteratively reduce the number of remaining candidate points. As indicated in Lemma 4.1, any convex lower bound of the objective function can be used for this purpose, and the resulting level sets will be smaller according as the approximation is better. For example, it would be interesting to study the use of bounds that may be obtained from Lagrangian relaxations of the second-stage problem.

Secondly, we propose to use Gröbner bases techniques to perform the many evaluations of the expected value function Q. This choice is motivated by the fact that, at least in theory, the use of Gröbner bases justifies our assumption that function evaluations are relatively inexpensive from a computational point of view. However, the course of the algorithm is not changed if the function evaluations are performed by any other means. Indeed, when using a Gröbner basis in the actual computations for our examples we obtained no significant speedup compared to using CPLEX to solve each instance of the second-stage integer problems from scratch. To some extent this may be due to our implementation, in which we used the general purpose package CoCoa to do the generalized divisions (cf. STEP 5 of our algorithm). It is reasonable to expect that the advantage of using Gröbner bases becomes more apparent as the second-stage problems become larger, but this is beyond the capabilities of the current version of CoCoa.

Several research groups are working on special purpose variants of Buchberger's algorithm for solving (deterministic) integer programs, and have obtained some very promising first results (see [13]). Any algorithmical progress in this field can be incorporated directly in our algorithm.

It would be interesting to further investigate the performance of our algorithm on larger problems, which will become feasible when computer codes that can determine Gröbner bases for moderately sized problems become available to us.

Acknowledgement

We wish to thank Rekha Thomas (Texas A&M University College Station) and Günter Ziegler (Technische Universität Berlin) for beneficial discussions, and two anonymous referees for their constructive comments.

References

[1] B. Bank and R. Mandel. *Parametric Integer Optimization.* Akademie-Verlag, Berlin, 1988.

[2] C.E. Blair and R.G. Jeroslow. The value function of a mixed integer program: I. *Discrete Mathematics*, 19:121–138, 1977.

[3] B. Buchberger. Gröbner bases: An algorithmic method in polynomial ideal theory. In N.K. Bose, editor, *Multidimensional Systems Theory*, chapter 6. D. Reidel Publishing Company, 1985.

[4] A. Capani and G. Niesi. *CoCoa User's Manual.* Dept. of Mathematics, University of Genova, release 3.0b edition, 1995.

[5] C.C. Carøe and J. Tind. L-shaped decomposition of two-stage stochastic programs with integer recourse. Technical report, Institute of Mathematics, University of Copenhagen, 1995.

[6] T. Christof. PORTA - a polyhedron representation transformation algorithm. Available via `http://www.iwr.uni-heidelberg.de/iwr/comopt/soft/PORTA/readme.html`.

[7] P. Conti and C. Traverso. Buchberger algorithm and integer programming. In *Proceedings AAECC-9 (New Orleans), Lecture Notes in Computer Science 539*, pages 130–139, Berlin, 1991. Springer-Verlag.

[8] D. Cox, J. Little, and D. O'Shea. *Ideals, Varieties and Algorithms.* Springer-Verlag, New York, 1992.

[9] *Using the CPLEX Callable Library.* CPLEX Optimization Inc., 1995.

[10] Y.M. Ermoliev, V.I. Norkin, and R.J-B. Wets. The minimization of semi-continuous functions: mollifier subgradients. *SIAM Journal on Control and Optimization*, 33(1):149–167, 1995.

[11] Yu. Ermoliev and R.J-B. Wets. *Numerical Techniques for Stochastic Optimization.* Springer-Verlag, Berlin, 1988.

[12] J.L. Higle and S. Sen. Stochastic decomposition: An algorithm for two-stage linear programs with recourse. *Mathematics of Operations Research*, 16:650–669, 1991.

[13] S. Hosten and B. Sturmfels. GRIN: An implementation of Gröbner bases for integer programming. In *Integer Programming and Combinatorial Optimization, Lecture Notes in Computer Science 920*, pages 267–276. Springer-Verlag, Berlin, 1995.

[14] P. Kall and S.W. Wallace. *Stochastic Programming.* Wiley, Chichester, 1994.

[15] W.K. Klein Haneveld, L. Stougie, and M.H. van der Vlerk. An algorithm for the construction of convex hulls in simple integer recourse programming. *Annals of Operations Research*, 64:67–81, 1996.

[16] W.K. Klein Haneveld and M.H van der Vlerk. On the expected value function of a simple integer recourse problem with random technology matrix. *Journal of Computational and Applied Mathematics*, 56:45–53, 1994.

[17] B.J. Lageweg, J.K. Lenstra, A.H.G. Rinnooy Kan, and L. Stougie. Stochastic integer programming by dynamic programming. In Yu. Ermoliev and R.J-B Wets, editors, *Numerical Techniques for Stochastic Optimization*, chapter 21. Springer-Verlag, Berlin, 1988.

[18] G. Laporte and F.V. Louveaux. The integer L-shaped method for stochastic integer programs with complete recourse. *Operations Research Letters*, 13:133–142, 1993.

[19] F.V. Louveaux and M.H. van der Vlerk. Stochastic programming with simple integer recourse. *Mathematical Programming*, 61:301–325, 1993.

[20] G.L. Nemhauser and L.A. Wolsey. *Integer and Combinatorial Optimization.* Wiley, New York, 1988.

[21] A. Prékopa. *Stochastic Programming.* Kluwer Academic Publishers, Dordrecht, 1995.

[22] R.T. Rockafellar. *Convex Analysis.* Princeton University Press, Princeton, New Jersey, 1970.

[23] A. Ruszczyński. A regularized decomposition method for minimizing a sum of polyhedral functions. *Mathematical Programming*, 35:309–333, 1986.

[24] A. Ruszczyński, Y. Ermoliev, and V. Norkin. On optimal allocation of indivisibles under uncertainty. *Operations Research*, to appear.

[25] R. Schultz. Continuity properties of expectation functions in stochastic integer programming. *Mathematics of Operations Research*, 18:578–589, 1993.

[26] R. Schultz. On structure and stability in stochastic programs with random technology matrix and complete integer recourse. *Mathematical Programming*, 70:73–89, 1995.

[27] R. Schultz, L. Stougie, and M.H. van der Vlerk. Two-stage stochastic integer programming: a survey. *Statistica Neerlandica*, 50(3):404–416, 1996.

[28] L. Stougie and M.H. van der Vlerk. Stochastic integer programming. In M. Dell'Amico, F. Maffioli, and S. Martello, editors, *Annotated Bibliographies in Combinatorial Optimization*, chapter 9, pages 127–141. Wiley, 1997.

[29] S.R. Tayur. A new algorithm to solve stochastic integer programs with application to plant management. Technical report, Carnegie Mellon University, Pittsburgh, in preparation.

[30] S.R. Tayur, R.R. Thomas, and N.R. Natraj. An algebraic geometry algorithm for scheduling in the presence of setups and correlated demands. *Mathematical Programming*, 69(3):369–401, 1995.

[31] R.R. Thomas. A geometric Buchberger algorithm for integer programming. *Mathematics of Operations Research*, 20:864–884, 1995.

[32] R.R. Thomas and R. Weismantel. Truncated Gröbner bases for integer programming. *Applicable Algebra in Engineering, Communication and Computing*, 8:241–256, 1997.

[33] R. Urbaniak, R. Weismantel, and G. Ziegler. A variant of Buchberger's algorithm for integer programming. *SIAM Journal on Discrete Mathematics*, 10:96–108, 1997.

[34] M.H. van der Vlerk. *Stochastic programming with integer recourse.* PhD thesis, University of Groningen, The Netherlands, 1995.

[35] R. Van Slyke and R.J-B. Wets. L-shaped linear programs with applications to control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17:638–663, 1969.

[36] S.W. Wallace and R.J-B. Wets. Preprocessing in stochastic programming: the case of linear programs. *ORSA Journal on Computing*, 4:45–59, 1992.