A New Algebraic Geometry Algorithm for Integer Programming

Dimitris Bertsimas • Georgia Perakis • Sridhar Tayur
Sloan School of Management, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139
Sloan School of Management, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139
Graduate School of Industrial Administration, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213

We propose a new algorithm for solving integer programming (IP) problems that is based on ideas from algebraic geometry. The method provides a natural generalization of the Farkas lemma for IP, leads to a way of performing sensitivity analysis, offers a systematic enumeration of all feasible solutions, and gives structural information of the feasible set of a given IP. We provide several examples that offer insights on the algorithm and its properties.

(*Integer Programming*; *Algebraic Geometry*; *Groebner Basis*)

1. Introduction

In this paper we introduce a new approach for solving integer programming problems (IPs). Our results are inspired from the observation that we can view any 0–1 IP as a system of quadratic equalities. We apply ideas from algebraic geometry to provide an algorithm for the problem that has several implications. Conti and Traverso (1991) introduced a very different approach for solving IPs that was also based on ideas from algebraic geometry. Nevertheless, unlike the approach in Conti and Traverso (1991) (see also Tayur et al. 1995, or Thomas 1995), our approach uses a specific right-hand side b (like Thomas and Weismantel 1997), and is directly motivated by 0–1 integer programming problems. Our algorithm may be viewed as a generalization of the Farkas lemma as well as a way of performing sensitivity analysis for IPs. Moreover, preliminary computational results indicate that our algorithm shows promise for problems that are either infeasible or have a small number of feasible solutions.

To make our results more accessible to the reader we will first focus on the 0–1 feasibility integer programming problem. Later in the paper we will illustrate how to extend our results for solving 0–1 as well as general integer optimization problems. Our formulation of the 0–1 feasibility integer programming problem is related to the work of Pitassi (1997), who formulates problems in logic as systems of polynomial equations, and considers the lengths of proofs of infeasibility based on the Nullstellensatz. This last idea was first suggested in a paper of Lovasz (1982).

Definition 1. Given an $m \times n$ matrix A, and an m-vector b, the *0–1 feasibility integer programming problem* is the problem of deciding whether there is an n-vector x with 0–1 coordinates such that

$$Ax = b, \qquad x \in \{0, 1\}^n. \tag{1}$$

This problem can be rewritten equivalently as the following system of equations,

$$Ax = b, \qquad x_j^2 = x_j, \qquad j = 1, \ldots, n.$$

The contributions of this paper are as follows:

1. We provide an algorithm for the 0–1 IP feasibility problem that systematically enumerates all feasible solutions or shows that none exists. The algorithm also provides a way to count exactly the number of solutions without the need to enumerate them.
2. We extend the algorithm for solving general IP optimization problems.

0025-1909/00/4607/0999$05.00
1526-5501 electronic ISSN

3. We show that the method leads to a way of performing sensitivity analysis in IPs.

4. We establish that the algorithm leads to a strong duality theory for IPs, in the sense that it provides a certificate for infeasibility.

5. We address the reverse problem, i.e., given a set of integer points in $\{0, 1\}^n$ we provide an underlying IP.

The paper is structured as follows. In §2, we review definitions and basic results from algebraic geometry to make the paper self contained. In §3, we present our algorithm for the 0–1 IP feasibility problem, and illustrate several of its properties. In §4, we extend our approach to the feasibility and optimization problems of general IPs. In §5 we illustrate the application of our algorithm to performing sensitivity analysis for IPs. The last section contains some concluding remarks.

2. Preliminaries

To make the paper self contained, we review in this section some basic definitions and results from an introductory text on computational algebraic geometry by Cox et al. (1997). The interested reader may consult this book for further details.

In this paper we work over an algebraically closed field k, which is the field of complex numbers $\mathscr{C}$. The polynomial ring over this field is represented by $k[x_1, \ldots, x_n]$.

Definition 2. Given polynomials $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$, the set $V(f_1, \ldots, f_s)$ such that

$$V(f_1, \ldots, f_s) := \{(a_1, \ldots, a_n) \in k^n : f_i(a_1, \ldots, a_n) = 0 \quad \forall\, i\},$$

is called the *affine variety* defined by $f_1, \ldots, f_s$.

The notion of polynomial ideals is closely connected to the notion of affine varieties.

Definition 3. A subset I of $k[x_1, \ldots, x_n]$, is an *ideal* if

(i) $0 \in I$;

(ii) if $f, g \in I$, then $f + g \in I$;

(iii) if $f \in I$, and $h \in k[x_1, \ldots, x_n]$, then $hf \in I$.

Remark. Given polynomials $f_1, \ldots, f_s$, we define $\langle f_1, \ldots, f_s \rangle$ as the set that consists of all polynomials that are obtained by $\sum_{i=1}^{s} h_i f_i$, with $h_i \in k[x_1, \ldots, x_n]$. It is not difficult to see that $\langle f_1, \ldots, f_s \rangle$ is an ideal. We call this "the ideal generated by $f_1, \ldots, f_s$." Note that all the ideals considered in this paper are polynomial ideals.

Definition 4. Given a *term order* on $k[x_1, \ldots, x_n]$ and a polynomial f in this ring, we define the *leading monomial* of f to be the highest monomial in f with respect to the term order.

We are now ready to introduce the notion of a Groebner basis for a given term order.

Definition 5. A subset of polynomials $Q := \{q_1, \ldots, q_t\}$ of an ideal F is a *Groebner basis* of F, if it has the property that all the leading monomials of F can be generated by the leading monomials of the polynomials in Q.

The details of the following result can be found in Cox et al. (1997).

Theorem A (Follows from Hilbert Basis Theorem). *For every polynomial ideal and every term order, there is a Groebner basis that has a finite number of elements.*

Given an ideal F generated by polynomials $f_1, \ldots, f_s$ and a term order, we can compute the Groebner basis of F using Buchberger's algorithm (see Cox et al. 1997, for further details).

We use $I(V)$ to denote the ideal that contains all polynomials that vanish on a given variety V and $V(I)$ to denote the variety $V(r_1, \ldots, r_s)$ where $R := \{r_1, \ldots, r_s\}$, is a Groebner basis of the ideal I.

Lemma A. *If $f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$, then $\langle f_1, \ldots, f_s \rangle \subset I(V(f_1, \ldots, f_s))$, although equality need not occur.*

Definition 6. Given $I := \langle f_1, \ldots, f_s \rangle \subset k[x_1, \ldots, x_n]$, the lth *elimination ideal* I_l is defined as $I_l = I \cap k[x_{l+1}, \ldots, x_n]$.

In the remainder of this paper we will use the following results from Cox et al. (1997).

Theorem B (The Elimination Theorem). *Let $I \subset k[x_1, \ldots, x_n]$ be an ideal and let G be a Groebner basis of I with respect to lex order where $x_1 > x_2 > \cdots > x_n$. Then, for every $0 \le l \le n$, the set*

$$G_l = G \cap k[x_{l+1}, \ldots, x_n]$$

is a Groebner basis of the lth elimination ideal I_l.

Theorem C (The Extension Theorem). *Let $I = \langle q_1, \ldots, q_s \rangle \subset k[x_1, \ldots, x_n]$ and let I_1 be the first elimination ideal of I. For each $1 \le i \le s$, we can write q_i in the form*

$$q_i = h_i(x_2, \ldots, x_n)x_1^{N_i} + \text{terms in which } x_1 \text{ has degree smaller than } N_i,$$

where $N_i \ge 0$ and $h_i \in k[x_2, \ldots, x_n]$ is nonzero. Suppose we have a partial solution $(a_2, \ldots, a_n) \in V(I_1)$. If $(a_2, \ldots, a_n) \notin V(h_1, \ldots, h_s)$, then there exists $a_1 \in k$ such that $(a_1, \ldots, a_n) \in V(I)$.

Corollary D. *Let $V := V(q_1, \ldots, q_s) \subset k^n$, and assume that for some i, q_i is of the form*

$$q_i = cx_1^N + \text{terms in which } x_1 \text{ has degree smaller than } N,$$

where $c \in k$ is nonzero and $N > 0$. If I_1 is the first elimination ideal, then in k^{n-1},

$$\pi_1(V) = V(I_1),$$

where π_1 is the projection on the last $n - 1$ coordinates.

Definition 7. An ideal I is *radical* if $f^m \in I$ for any integer $m \ge 1$ implies that $f \in I$.

Definition 8. Let $I \subset k[x_1, \ldots, x_n]$ be an ideal. The *radical* of I, denoted $\sqrt{I}$, is the set $\{f : f^m \in I$ for some integer $m \ge 1\}$.

Theorem E. (Nullstellensatz). *Let k be an algebraically closed field. If I is an ideal in $k[x_1, \ldots, x_n]$, then $I(V(I)) = \sqrt{I}$.*

3. An Algorithm for the 0–1 Feasibility IP

In the previous section we laid the foundation for presenting an algorithm for solving the 0–1 feasibility IP, Problem (1). We consider the following polynomials in $k[x_1, \ldots, x_n]$:

$$f_i = \sum_{j=1}^{n} a_{ij}x_j - b_i, \qquad i = 1, \ldots, m,$$

$$g_j = x_j^2 - x_j, \qquad j = 1, \ldots, n.$$

We let $\tilde{V} := V(f_1, \ldots, f_m, g_1, \ldots, g_n)$ be the variety they define.

Since k is the field of complex numbers, $\tilde{V}$ is the feasible set of the 0–1 IP with matrix A and right-hand side b. (The data is assumed to be in $\mathfrak{R}$.) This is based on the simple observation that either $\tilde{V}$ is empty, or if there is an element, it is actually in $\mathfrak{R}$ (it is in fact integral).

We consider the ideal $J := \langle f_1, \ldots, f_m, g_1, \ldots, g_n \rangle$. The algorithm we propose enumerates all feasible 0–1 solutions, or detects that no feasible solution exists.

Algorithm A.
Input: Matrix A and vector b.
Output: All feasible solutions $(a_1, \ldots, a_n)$ to Problem (1).

1. Find a Groebner basis G of J using lex order $x_1 > x_2 > \cdots > x_n$. If $G = \{1\}$, then the 0–1 IP has no feasible solutions. *Exit.*
2. If $G \neq \{1\}$: Consider for $1 \le l \le n - 1$, the sets $G_l = G \cap k[x_{l+1}, \ldots, x_n]$. Starting from $l = n - 1$, and working sequentially:
   - Find a_n in $V(G_{n-1})$.
   - Extend a_n to (a_{n-1}, a_n) such that $(a_{n-1}, a_n) \in V(G_{n-2})$.
   - $\vdots$
   - Find a_2 such that $(a_2, \ldots, a_n) \in V(G_1)$.
   - Find a_1 such that $(a_1, \ldots, a_n) \in V(G)$.

We next show that the algorithm correctly solves 0–1 feasibility IPs (Problem (1)).

Theorem 1. *Algorithm* A *either provides all feasible solutions for Problem* (1), *or provides a certificate of infeasibility whenever the Groebner basis $G = \{1\}$.*

Proof. Consider the elimination ideals $J_1, \ldots, J_{n-1}$, where $J_k = J \cap k[x_{k+1}, \ldots, x_n]$. If we find the Groebner basis G of J using *lex order* (via Buchberger's algorithm), then Theorem B implies that $G_k := G \cap k[x_{k+1}, \ldots, x_n]$ is a Groebner basis of J_k, further implying that $V(G_k) = V(J_k)$. Observe that one of the following two cases holds:

(a) The Groebner basis $G = \{1\}$. Then the ideal J coincides with $k[x_1, \ldots, x_n]$, indicating that $V(J)$ is empty, since we are working on an algebraically closed field. Therefore, $\tilde{V}$ is an empty set implying that we have an infeasible integer program.

(b) The Groebner basis $G \neq \{1\}$. In this case, we notice that G_{n-1} has elements $x_n - 1$ or x_n or $x_n^2 - x_n$ belonging to it. This follows from the observation that J_{n-1} is a polynomial ideal in one variable, namely a subset in $k[x_n]$ and therefore, needs only one generator. We interpret this to mean that points 1 or 0 or both 0 and 1 are *partial* solutions respectively. That is, we get an a_n in $V(J_{n-1})$. Subsequently we obtain a_{n-1} by extending the partial solution a_n to $(a_{n-1}, a_n) \in V(J_{n-2})$. That is, we take a partial solution in $V(J_k)$ and extend it to a partial solution in $V(J_{k-1})$ and so on. By Theorem C, this is always possible as the appropriate leading coefficients (starting from f_i and g_j) in our setting are constants. Continuing in this way and applying Theorem C, we can obtain a solution $(a_1, \ldots, a_n)$ in $V(J)$.

We are interested, however, in a point of $\tilde{V}$. It is easy to see that the projection π_k of $\tilde{V}$ to the last $n - k$ coordinates satisfies $\pi_k(\tilde{V}) \subset V(I_k)$ (see Lemma 1, page 120 of Cox, Little, and O'Shea 1997). We notice that a repeated application of Corollary D indicates that

$$\pi_k(\tilde{V}) = V(I_k),$$

and therefore, $(a_2, \ldots, a_n) \in V(J_1)$ obtained by the Algorithm A is in fact in $\pi_1(\tilde{V})$. We can find a point in $\tilde{V}$ using any one of the f_i. Thus, all solutions are found by this procedure, and no infeasible ones are picked up. □

Remarks.

1. It is important to observe that using different a_n from $V(J_{n-1})$, we can find *all* solutions to the given 0–1 integer program.

2. For a given ideal the Groebner basis is not typically unique. However, we may use the notion of the *reduced Groebner basis* (see Cox et al. 1997), which is unique for the ideal for a given term order. All our results are reported using the reduced Groebner basis for the lex order.

We next illustrate the use of Algorithm A. All computations in this paper have been done using our own implementation of Buchberger's algorithm in C++ on a personal computer.

Example 1. Consider the IP

$$x_1 + 2x_2 + 3x_3 + 4x_4 + 5x_5 + 15x_6 = 15, \qquad x_j \in \{0, 1\} \ \forall j.$$

The ideal we consider is

$$J = \langle x_1 + 2x_2 + 3x_3 + 4x_4 + 5x_5 + 15x_6 - 15, \; x_1^2 - x_1, x_2^2 - x_2, \ldots, x_6^2 - x_6 \rangle.$$

The sorted reduced Groebner basis of J with lex order is

$$G = \{x_6^2 - x_6, x_5 + x_6 - 1, x_4 + x_6 - 1, \; x_3 + x_6 - 1, x_2 + x_6 - 1, x_1 + x_6 - 1\}.$$

Therefore, we have $G_5 = \{x_6^2 - x_6\}$, indicating that $a_6 = 1$ and $a_6 = 0$ are both partial solutions. Starting from $a_6 = 1$, we get $a_1 = a_2 = \cdots = a_5 = 0$. Starting from $a_6 = 0$, we get $a_1 = \cdots = a_5 = 1$. Therefore, we have two feasible solutions: (0, 0, 0, 0, 0, 1) and (1, 1, 1, 1, 1, 0).

An interesting feature of the Groebner basis, for example, is the interpretation of the term $x_5 + x_6 - 1$. This implies that in all solutions exactly one of x_5 and x_6 is equal to 1. This example illustrates the structural information that the Groebner basis contains regarding a 0–1 IP.

The next example amplifies the observation that Algorithm A captures logical interactions between variables.

Example 2. Consider the IP

$$x_1 + 3x_2 + 2x_3 + 2x_4 + 4x_5 + 4x_6 = 4, \qquad x_j \in \{0, 1\} \ \forall j.$$

The sorted reduced Groebner basis with lex order is

$$G = \{x_6^2 - x_6, x_5x_6, x_5^2 - x_5, x_4x_6, x_4x_5, x_4^2 - x_4, \; x_3 - x_4, x_2 + x_4 + x_5 + x_6 - 1, \; x_1 + x_4 + x_5 + x_6 - 1\}.$$

The element $x_3 - x_4$ implies that both variables should always have the same value in any feasible solution. Other elements are interpreted accordingly.

3.1. On the Structure of the Reduced Groebner Basis Obtained from Algorithm A

In this section, we outline some structural properties of the reduced Groebner basis obtained from Algorithm A. Property 1 follows from Exercise 7, p. 200, of Cox et al. (1997).

PROPERTY 1. If the solution of Problem (1) is unique, then all the reduced Groebner basis elements are of the form $x_i - a_i$ with $a_i \in \{0, 1\}$.

The following example illustrates Property 1.

EXAMPLE 3. Consider the following 0–1 feasibility integer programming problem.

$$x_1 + x_2 + x_3 = 2,$$

$$x_1 + x_2 = 2,$$

$$x_2 + x_3 = 1,$$

$$x_i \in \{0, 1\}, \qquad i = 1, \dots, 3.$$

This problem has the unique feasible solution $x_1 = x_2 = 1$, $x_3 = 0$. Furthermore, Algorithm A yields this solution through the sorted reduced Groebner basis $G = \{x_3, x_2 - 1, x_1 - 1\}$.

PROPERTY 2. The polynomials in the reduced Groebner basis with the lexicographic term order $x_1 > \cdots > x_n$ can be partitioned in n sets $S_n, \dots, S_1$ as follows:

1. Set S_n contains only one polynomial, which is either x_n, $x_n - 1$ or $x_n^2 - x_n$.
2. Set S_{n-1} contains polynomials in x_n and x_{n-1}.
3. Set S_i, $i = n - 2, \dots, 1$ contains polynomials in $x_n, x_{n-1}, \dots, x_i$.

This property follows from the fact that the underlying term order is an elimination order. This "diagonalized structure" of the reduced Groebner basis—a generalization of Gaussian elimination for the $n \times n$ linear system of equations—is useful in enumerating the feasible 0–1 solutions. Starting with S_n, we assign values for $x_n = 0$ (if $x_n \in S_n$), or $x_n = 1$ (if $x_n - 1 \in S_n$), or $x_n = 0, 1$ (if $x_n^2 - x_n \in S_n$). Having chosen x_n, we backsolve for x_{n-1} using the polynomials in S_{n-1}, and proceed to assigning values to the other variables recursively. The next example illustrates Property 2.

EXAMPLE 4. Consider the 0–1 feasibility IP:

$$x_1 + 2x_2 + 3x_3 + 4x_4 + 6x_5 = 6, \qquad x_j \in \{0, 1\} \ \forall j.$$

The reduced Groebner basis for lex order is

$$G = \{x_5^2 - x_5, x_4x_5, x_4^2 - x_4, x_3 + x_4 + x_5 - 1, \\ x_2 + x_5 - 1, x_1 + x_4 + x_5 - 1\}.$$

The sets are as follows: $S_5 = \{x_5^2 - x_5\}$, $S_4 = \{x_4x_5, x_4^2 - x_4\}$, $S_3 = \{x_3 + x_4 + x_5 - 1\}$, $S_2 = \{x_2 + x_5 - 1\}$, $S_1 = \{x_1 + x_4 + x_5 - 1\}$.

Algorithm A enumerates all 0–1 solutions:

$$(0, 0, 0, 0, 1), \quad (0, 1, 0, 1, 0), \quad (1, 1, 1, 0, 0).$$

PROPERTY 3. Let $x_1 > x_2 > \cdots > x_n$ be the lex order. Polynomials in the reduced basis G under lex order can be partitioned as follows:

1. *Group B*: Binomials of the type $x_j^2 - x_j$, $j \in S = \{k, k + 1, \dots, n\}$, for some k, i.e., the binomials correspond to variables that are lower in the term order.
2. *Group A*: Square free polynomials. This group can be further partitioned in two subgroups:
   (a) *Group A*1: Polynomials that have a leading term x_i, $i \notin S$.
   (b) *Group A*2: Polynomials that are square free with a leading monomial term: $x_{i_1}x_{i_2}\cdots x_{i_r}$, $i_l \in S$.

Since the number of points in $V(G)$ is finite, we are dealing with "zero dimensional" varieties and so we have $|A1| + |B| = n$. In order to see why Property 3 holds we argue as follows. By minimality of the reduced Groebner basis we cannot have x_i^2 and x_i as leading terms. This justifies the partition to groups B and A1 such that $|A1| + |B| = n$. Moreover, because the Groebner basis is reduced, we cannot have a monomial term x_j in one of the polynomials in the reduced Groebner basis, and x_j appearing as a leading term in another polynomial. Finally, the square free polynomials with a leading monomial term: $x_{i_1}x_{i_2}\cdots x_{i_r}$ need to have $i_l \in S$, otherwise $x_{i_1}x_{i_2}\cdots x_{i_r} \in \langle x_{i_l}\rangle$ with x_{i_l} being the leading monomial term in a polynomial in the Groebner basis.

The following example illustrates Property 3.

EXAMPLE 5. Consider the following 0–1 feasibility IP

$$x_1 + x_2 + x_3 + x_5 + 2x_6 = 2,$$

$$x_2 + x_4 + x_7 = 1,$$

$$x_3 + x_4 + x_8 = 1,$$

$$x_i \in \{0, 1\}, \qquad i = 1, \ldots, 8.$$

The sorted reduced Groebner basis G for the term order $x_8 > x_7 > x_6 > x_5 > x_1 > x_2 > x_3 > x_4$ is

$$G = \{\mathbf{x_8} + x_4 + x_3 - 1, \qquad (A1)$$

$$\mathbf{x_7} + x_2 + x_4 - 1, \qquad (A1)$$

$$\mathbf{x_6} - x_1x_2 - x_1x_3 + x_1 - x_2x_3 + x_2 + x_3 - 1, \qquad (A1)$$

$$\mathbf{x_5} + 2x_1x_2 + 2x_1x_3 - x_1 + 2x_2x_3 - x_2 - x_3, \qquad (A1)$$

$$\mathbf{x_1^2} - x_1, \qquad (B)$$

$$x_1x_2x_3, \qquad (A2)$$

$$\mathbf{x_2^2} - x_2, \qquad (B)$$

$$x_2x_4, \qquad (A2)$$

$$\mathbf{x_3^2} - x_3, \qquad (B)$$

$$x_3x_4, \qquad (A2)$$

$$\mathbf{x_4^2} - x_4\}. \qquad (B)$$

Based on this example one might be tempted to think that terms in A2 are only monomials. However, this is not true, since if we use the usual term order $x_1 > \cdots > x_8$ in the above example we find that one of the polynomials in the reduced Groebner basis is $x_6x_7 - x_6x_8$.

We can use Property 3 in order to enumerate all feasible solutions more efficiently. We first enumerate all 0–1 solutions implied by the binomials in the set B. For a fixed set of values in the set B, we use the equations in the set A1 to assign values for the remaining variables. If any of the equations in A2 is violated, we reject this solution; otherwise we accept it.

3.2. The Number of 0–1 Solutions

We next show that the reduced Groebner basis provides exact information about the number of 0–1 feasible solutions to Problem (1). Let S be the feasible set of the 0–1 integer program (1). Let $I = I(S)$. Then I is a radical ideal[1] (see p. 173 of Cox et al. 1997). The next theorem holds for any order, not only the lexicographic one if the ideal is *radical* (see Proposition 8(ii), Chapter 5, p. 232 of Cox et al. 1997).

THEOREM 2. *The number of solutions of Problem* (1) *is equal to the cardinality of the set M of monomials that are not multiples of the leading monomials of the polynomials in the reduced Groebner basis.*

Note that if $G = \{1\}$, then $M = \emptyset$, and thus the number of feasible solutions is $|M| = 0$, i.e., the problem is infeasible. The following example illustrates the use of Theorem 2.

EXAMPLE 5 (CONTINUED). The leading monomials in the Groebner basis in Example 5 are:

$$x_8,\ x_7,\ x_6,\ x_5,\ x_1^2,\ x_1x_2x_3,\ x_2^2,\ x_2x_4,\ x_3^2,\ x_3x_4,\ x_4^2.$$

The set of monomials that are no multiples of the leading monomials is:

$$M = \{1,\ x_1,\ x_2,\ x_3,\ x_4,\ x_1x_2,\ x_1x_3,\ x_1x_4,\ x_2x_3\}.$$

Since $|M| = 9$, Example 5 has exactly 9 solutions. These solutions are:

$$(0, 0, 0, 1, 0, 1, 0, 0),\quad (1, 0, 0, 1, 1, 0, 0, 0),\quad (0, 1, 1, 0, 0, 0, 0, 0),$$
$$(1, 1, 0, 0, 0, 0, 0, 1),\quad (0, 1, 0, 0, 1, 0, 0, 1),\quad (1, 0, 1, 0, 0, 0, 1, 0),$$
$$(0, 0, 1, 0, 1, 0, 1, 0),\quad (1, 0, 0, 0, 1, 0, 1, 1),\quad (0, 0, 0, 0, 0, 1, 1, 1).$$

[1] We can easily verify that indeed $I(S)$ is an ideal. To prove that $I(S)$ is also radical, that is $I = \sqrt{I}$, pick an $f \in \sqrt{I}$. (Note that $I \subset \sqrt{I}$ always.) Then, by definition, $f^m = 0$ for some m, for all $a \in S$. This implies that $f(a) = 0$, for all $a \in S$, indicating that $f \in I$.

3.3. An Interpretation of Algorithm A as Farkas Lemma for 0–1 IPs

Farkas lemma, which is the central idea of duality in linear programming, provides a certificate of infeasibility for a linear programming problem. *What is a certificate of infeasibility of a* 0–1 *IP*? An obvious (and very inefficient) certificate is the enumeration of all possible 2^n vectors. Nevertheless, Algorithm A provides a potential certificate in a more efficient way. If the 0–1 IP is infeasible, then $G = \{1\}$. In other words, the certificate is the computation of the Groebner basis.

EXAMPLE 6. Consider the following example:

$$2x_1 + 2x_2 + 4x_3 + 6x_4 = 11, \qquad x_j \in \{0, 1\}.$$

Clearly, this is an infeasible IP, as the left-hand side is an even integer, while the right-hand side is an odd one. In this case $G = \{1\}$.

3.4. Constructing an IP from a Given Set of Points in $\{0, 1\}^n$

To this point, this paper has addressed the problem of finding the feasible points of a 0–1 feasibility integer programming problem (Problem (1)). In this subsection, we will address the reverse problem. That is, we will establish how to construct a 0–1 feasibility integer programming problem for a given set of integer points in $\{0, 1\}^n$.

THEOREM 3. *Suppose we are given a set of points S in $\{0, 1\}^n$ representing the feasible space of some* 0–1 *IP. We can provide an underlying IP and construct a Groebner basis of an ideal I such that $V(I) = S$.*

PROOF. Given a subset S of points in $\{0, 1\}^n$ we can enumerate the points $P^i = (p_1^i, \ldots, p_n^i)$ in $\{0, 1\}^n$, for $i = 1, \ldots, m$, that do not lie in the set S. The following inequality describes a set of points in R^n that excludes only point P^i,

$$\sum_{j=1}^{n} (x_j - p_j^i)^2 \geq 1.$$

Therefore, we can describe the set S through the following set of quadratic inequalities,

$$\sum_{j=1}^{n} (x_j - p_j^i)^2 \geq 1, \qquad i = 1, \ldots, m,$$

$$x_j^2 - x_j = 0, \qquad j = 1, \ldots, n.$$

Nevertheless, the observation that the variables x_j are 0 or 1, allows us to rewrite this as a set of linear inequalities combined with separable, quadratic equalities for each variable as follows:

$$\sum_{j=1}^{n} [x_j(1 - 2p_j^i) + p_j^i] \geq 1, \qquad i = 1, \ldots, m,$$

$$x_j^2 - x_j = 0, \qquad j = 1, \ldots, n.$$

Introducing a binary expression for the excess variables corresponding to each inequality yields the following set of linear equalities with integer variables,

$$\sum_{j=1}^{n} [x_j(1 - 2p_j^i) + p_j^i] - \sum_{k=0}^{\lceil \log n \rceil - 1} 2^k x_{n+1+(i-1)\lceil \log n \rceil + k} = 1,$$

$$i = 1, \ldots, m,$$

$$x_j^2 - x_j = 0, \qquad \forall j.$$

The previous representation formulates the set of points S in the form of Problem (1). For this new representation we can apply Buchberger's algorithm to find a Groebner basis. □

4. Optimization of IPs

In this section, we will generalize our results of §3 for solving general integer programming problems. In the next subsection, we will illustrate how to solve 0–1 IPs.

4.1. Optimization of 0–1 IPs

We consider the optimization problem

$$\begin{aligned} &\text{minimize} && c'x, \\ &\text{subject to} && Ax = b, \\ &x_j^2 = x_j, && \forall j. \end{aligned}$$

The largest objective function value is $\bar{Z} = \sum_{j:c_j \geq 0} c_j$. Let Z_{LP} be the value of the LP relaxation. Clearly, $\lceil Z_{\text{LP}} \rceil \leq Z_{\text{IP}} \leq \bar{Z}$. This observation allows us to apply binary search on Z_{IP} and solve the optimization problem as follows. We add the polynomial $h := \sum_{j=1}^{n} c_j x_j - C$ to the generators of J, for specific values of C in the range $[\lceil Z_{\text{LP}} \rceil, \bar{Z}]$. We thus need to apply Algorithm A at most $\log(\bar{Z} - \lceil Z_{\text{LP}} \rceil)$ times.

A more direct method is as follows. We work in $k[x_1, \ldots, x_n, y]$. Let

$$h := y - \sum_{j=1}^{n} c_j x_j$$

and we look at $\hat{V} := V(f_1, \ldots, f_m, g_1, \ldots, g_n, h)$. Following the same approach as the one for feasibility, with lex order $x_1 > x_2 > \cdots > x_n > y$, we notice that the Groebner basis $\hat{G}$ of $\hat{J} := \langle f_1, \ldots, f_m, g_1, \ldots, g_n, h \rangle$ is either $\{1\}$ (indicating infeasibility) or we will have $\hat{G}$ intersected with $k[y]$ which leads to a polynomial in y. We interpret this polynomial in y as follows: Every root of the polynomial is a feasible cost of the IP. Therefore, we can find the minimum root, and work upwards to get the associated x_j values.

Example 7. Consider the IP

$$\begin{aligned} \text{minimize} \quad & x_1 + 2x_2 + 3x_3, \\ \text{subject to} \quad & x_1 + 2x_2 + 2x_3 = 3, \\ & x_j^2 = x_j, \qquad j = 1, \ldots, 3. \end{aligned}$$

The reduced Groebner basis of

$$J = \langle y - x_1 - 2x_2 - 3x_3,\ x_1^2 - x_1,\ 2x_1 + 4x_2 + 4x_3 - 6,\ x_2^2 - x_2,\ x_3^2 - x_3 \rangle$$

is

$$G = \{12 - 7y + y^2,\ 3 + x_3 - y,\ -4 + x_2 + y,\ 1 - x_1\}.$$

The two roots of the polynomial $12 - 7y + y^2$ are $y = 3$ and $y = 4$. Thus the minimum value is $y = 3$, and the corresponding solution is (1, 1, 0).

We next illustrate that we can simplify the calculation of Algorithm A if we have partial information on the optimal cost.

Example 8. Consider the IP

$$\begin{aligned} \text{minimize} \quad & x_1 + 2x_2 + 3x_3 + 3x_4, \\ \text{subject to} \quad & x_1 + x_2 + 2x_3 + x_4 = 3, \\ & x_j^2 = x_j, \qquad j = 1, \ldots, 4. \end{aligned}$$

The reduced Groebner basis with lex order of $J = \langle x_1 + 2x_2 + 3x_3 + 3x_4 - y,\ 2x_1 + 2x_2 + 4x_3 + 2x_4 - 6,\ x_1^2 - x_1,\ x_2^2 - x_2,\ x_3^2 - x_3,\ x_4^2 - x_4 \rangle$ is

$$\begin{aligned} G = \{ & 120 - 74y + 15y^2 - y^3,\ -20 + 2x_4 + 9y - y^2, \\ & -6 + 6x_3 + y - x_3 y,\ x_3 - x_3^2, \\ & -23 - x_2 - x_3 - x_4 + 10y - y^2, \\ & 32 - 2x_1 - 2x_3 - 11y + y^2\}, \end{aligned}$$

which suggests that all feasible ys are $y = 4, 5, 6$, i.e., the roots of the first equation of the Groebner basis. If however, we have additional information that $4 \leq y \leq 5$, we can add the polynomial $(y - 4)(y - 5)$ and rerun Algorithm A. The reduced Groebner basis is then:

$$G = \{20 - 9y + y^2, -1 + x_3, -4 - x_2 + y, 5 - x_1 - y\}.$$

A natural question is to compare the performance of Algorithm A to branch and bound. Our next example addresses this issue.

Example 9. We consider the class of integer programming problems (with n odd):

$$\begin{aligned} \text{minimize} \quad & x_{n+1}, \\ \text{subject to} \quad & 2x_1 + 2x_2 + \cdots + 2x_n + x_{n+1} = n, \\ & x_i \in \{0, 1\}. \end{aligned}$$

It is easy to show that any branch and bound algorithm that uses linear programming relaxations to compute lower bounds, and branches by setting a fractional variable to either zero or one, will require the enumeration of an exponential number of subproblems when n is odd (see Bertsimas and Tsitsiklis 1997). It is thus interesting to observe the performance of Algorithm A. Applying Algorithm A to this problem we obtain relatively quickly that the first polynomial in the reduced Groebner basis is $x_{n+1} - 1$, suggesting that $x_{n+1} = 1$.

4.2. Optimization of General IPs

An arbitrary IP, in which the variables are only restricted to be nonnegative integers, can be reduced in a standard way to the 0–1 case as follows: If $x_j \in \{0, 1, \ldots, U_j\}$ with U_j known, then for each j, we write $x_j = \sum_{p=0}^{\lceil \log U_j \rceil - 1} 2^p x_j^p$, with the auxiliary variables x_j^p taking values either 0 or 1. We then substitute the above expression for x_j in the objective function and the constraints. Alternatively, we can include the polynomial

$$h = x_j(x_j - 1)\cdots(x_j - U_j)$$

to the ideal J and apply Algorithm A. The next example illustrates an application of this idea. Instead of considering $x_j \in \{0, 1, 2\}$, we consider without loss of generality the case $x_j \in \{-1, 0, 1\}$, i.e., $x_j^3 = x_j$.

Example 10. We consider the IP

$$2x_1 - 2x_2 + x_3 = 1,$$

$$3x_1 + x_2 + 2x_3 = 1,$$

$$x_j \in \{-1, 0, 1\}.$$

Then, we apply Algorithm A on the ideal $J = \langle 2x_1 - 2x_2 + x_3 - 1, 3x_1 + x_2 + 2x_3 - 1, x_1^3 - x_1, x_2^3 - x_2, x_3^3 - x_3 \rangle$. The reduced Groebner basis is

$$G = \{x_3 + 1, x_2, x_1 - 1\}.$$

The general case of x_j nonnegative integers can be reduced to the bounded case as follows. Papadimitriou and Steiglitz (1982), prove that if the IP $Ax = b$, $x_j \in Z^+$ has a solution, then it has a solution with $x \in \{0, 1, \ldots, M\}^n$, where $M = n(ma_{\max})^{2m+3}(1 + b_{\max})$, with $a_{\max} = \max|a_{ij}|$, and $b_{\max} = \max|b_i|$.

4.3. IPs with Inequality Constraints

In this subsection, we will illustrate how to solve IPs with inequality constraints. The key idea in showing this is to convert the given IP into an IP with linear equality constraints and integer variables.

Consider the general IP problem

$$\begin{aligned} &\text{minimize} \quad c'x, \\ &\text{subject to} \quad a_i'x \le b_i, \quad i = 1, \ldots, m, \\ &\qquad x_j \text{ integer}, \qquad j = 1, \ldots, n. \end{aligned}$$

We assume that all data in the problem is integral. For each inequality $i = 1, \ldots, m$, we will introduce a nonnegative integer slack variable s_i. Nevertheless, we can rewrite a binary expression of these variables as before. This observation allows us to convert our problem into an IP with linear equalities and integer variables.

The following example illustrates this.

Example 11. We consider the IP

$$x_1 + x_2 + x_3 \le 2,$$

$$x_2 + x_3 + x_4 \le 4,$$

$$x_i \in \{0, 1\}, \qquad i = 1, \ldots, 4.$$

We introduce slack variables s_1 and s_2 for the first and second constraint, respectively. Clearly, $s_1 \le 2$ and $s_2 \le 4$. Furthermore, we rewrite these variables as $s_1 = x_5 + 2x_6$ and $s_2 = x_7 + 2x_8 + 4x_9$, with $x_5, \ldots, x_9 \in \{0, 1\}$.

Therefore, we rewrite the IP as

$$x_1 + x_2 + x_3 + x_5 + 2x_6 = 2,$$

$$x_2 + x_3 + x_4 + x_7 + 2x_8 + 4x_9 = 4,$$

$$x_i^2 - x_i = 0, \qquad i = 1, \ldots, 9.$$

We are now able to solve this problem using Algorithm A, as we have shown in the previous subsections.

5. Sensitivity Analysis of IPs

Our results in this paper also allow us to perform sensitivity analysis for integer programming problems. That is, they allow us to address the problem of finding the optimal objective function value as a function of one of the right-hand side coefficients b_i. To achieve this we work in $k[x_1, \ldots, x_n, y, b_i]$ with lex order $x_1 > \cdots > x_n > y > b_i$, and find the Groebner basis G. We find that either (i) $G = \{1\}$, indicating that there is no value of b_i for which the problem is feasible, or (ii) $G \cap k[b_i]$ is a polynomial in b_i, and each of the roots of this polynomial represents a value for which the problem has a feasible solution. In this case, $G \cap k[y, b_i]$ are polynomials in y and b_i, and so we have an explicit representation of the *value function*.

Example 12. Consider the IP

$$\text{minimize} \quad x_1 + 2x_2 + 3x_3,$$

$$\text{subject to} \quad 2x_1 + 2x_2 + 4x_3 = b,$$

$$x_j^2 = x_j, \qquad j = 1, \dots, 3.$$

Suppose we are interested in finding the optimal solution value as a function of b, when $4 \leq b \leq 6$.

The reduced Groebner basis with lex order of

$$J = \langle x_1 + 2x_2 + 3x_3 - y,\ 2x_1 + 2x_2 + 4x_3 - b,\ (b-4)(b-5)(b-6),\ x_1^2 - x_1,\ x_2^2 - x_2,\ x_3^2 - x_3 \rangle$$

is

$$G = \{24 - 10b + b^2,\ 18 - 3b - 6y + by,\ -14 - b + 9y - y^2,\ -4 + b + 4x_3 - bx_3,\ 3 - 3x_3 - y + x_3y,\ -x_3 + x_3^2,\ -b - 2x_2 - 2x_3 + 2y,\ b - x_1 - x_3 - y\},$$

which implies that for $b = 4$, $y = 3$, the two solutions are (0, 0, 1) and (1, 1, 0). For $b = 5$, there is no solution, and for $b = 6$, the feasible y are roots of the equation $-20 + 9y - y^2 = 0$, i.e., $y = 4$ and $y = 5$.

6. Conclusions

In this paper we used ideas of algebraic geometry to present a method for solving integer programming problems. We started by presenting a method for solving the 0–1 feasibility integer programming problem, which we subsequently extended to solving general integer optimization problems. For the feasibility problem, our method provides a systematic enumeration of all feasible solutions. Our results may be viewed as a natural generalization of Farkas' lemma to integer programming and allow us to check whether a given problem is infeasible. Our results also lead to a way of performing sensitivity analysis. Finally, we also addressed the reverse problem, that is, how to provide an IP formulation for a given set of integer points in $\{0, 1\}^n$.

We have experimented with several integer programming problems of up to 100 variables. We have used our own implementation of Buchberger's algorithm.

In preliminary computational work, we have observed that Algorithm A is computationally faster, when the problem is either infeasible or it has very few solutions. In such situations, we have been able to solve problems with 100 variables on a personal computer. The following is such an example.

EXAMPLE 13. Let n be an even number. Consider the following 0–1 feasibility IP:

$$x_i + x_{i+1} = 1, \qquad i = 1, \dots, n-1,$$

$$x_j^2 = x_j, \qquad j = 1, \dots, n.$$

We have used $n = 100$. Algorithm A gives the Groebner basis

$$G = \{x_n^2 - x_n,\ x_{2k-1} + x_n - 1,\ k = 1, \dots, n/2,\ x_{2k} - x_n,\ k = 1, \dots, n/2 - 1\}.[2]$$

[2] The authors would like to thank the two reviewers of the paper for many insightful comments and suggestions that improved the paper. The first author's research was supported by NSF Grant DMI-9610486. The second author's research was supported by NSF Grant DMI-9634736.

References

Bertsimas, D., J. Tsitsiklis. 1997. *Introduction to Linear Optimization*. Athena Scientific, Belmont, MA.

Conti, P., C. Traverso. 1991. Buchberger algorithm and integer programming. *Proc. AAECC-9, New Orleans, LNCS* **539** 130–139.

Cox, D., J. Little, D. O'Shea. 1997. *Ideals, Varieties and Algorithms*. Springer, Berlin, Germany.

——, ——, ——. 1998. *Using Algebraic Geometry*. Springer, Berlin, Germany.

Lovasz, L. 1982. Bounding the independence number of a graph. *Ann. Discrete Math.* **16** 213–223.

Nemhauser, G., L. Wolsey. 1988. *Integer and Combinatorial Optimization*. Wiley, New York.

Papadimitriou, C., K. Steiglitz. 1982. *Combinatorial Optimization; Algorithms and Complexity*. Prentice-Hall, Englewood Cliffs, NJ.

Pitassi, T. 1997. Algebraic propositional proof systems. *DIMACS Ser. Discrete Math. Theoretical Comput. Sci.* **31** 215–244.

Tayur, S., R. Thomas, N. R. Natraj. 1995. An algebraic geometry algorithm for scheduling in presence of setups and correlated demands. *Math. Programming* **69** 369–401.

Thomas, R. 1995. A geometric Buchberger algorithm for integer programming. *Math. Oper. Res.* **20** 864–884.

——, R. Weismantel. 1997. Truncated Groebner bases for integer programming. *Algebra Engrg., Comm. Comput.* **8** (4) 241–257.

Urbaniak, R., R. Weismantel, G. Ziegler. 1997. A variant of the Buchberger algorithm for integer programming. *SIAM J. Discrete Math.* **1** (10) 96–108.

Accepted by Thomas M. Liebling; received April 1998. *This paper was with the authors* 11 *months for* 2 *revisions.*