The Minimised Geometric Buchberger Algorithm: An Optimal Algebraic Algorithm for Integer Programming

Qiāng Li*  Yi-ke Guo  Tetsuo Ida*  John Darlington

*Institute of Information Sciences and Electronics
University of Tsukuba
Tsukuba, Ibaraki, 305, Japan
liq.ida@score.is.tsukuba.ac.jp
http://www.score.is.tsukuba.ac.jp
Department of Computing
Imperial College
180 Queen's Gate, London SW7 2BZ, U.K.
yg,jd@doc.ic.ac.uk
http://www.doc.ic.ac.uk

Abstract

IP problems characterise combinatorial optimisation problems where conventional numerical methods based on the hill-climbing technique can not be directly applied. Conventional methods for solving integer programming are based on searching algorithms where heuristics such as branch and bound are applied to reduce the search space. Recently, various algebraic IP solvers have been proposed based on the theory of Gröbner bases. The key idea is to encode an IP problem $IP_{A,C}$ into a special ideal associated with the constraint matrix $A$ and the cost (object) function $C$. An important property of such an encoding is that its Gröbner basis corresponds directly to the test set of the IP problem. The main difficulty of these new methods is the size of the Gröbner bases generated. In the proposed algorithm, large Gröbner bases are caused by either introducing additional variables or by considering the generic IP problem $IP_{A,C}$. Some improvements have been proposed such as the Hosten and Sturmfels method (GRIN) designed to avoid additional variables and the truncated Gröbner basis method of Thomas which computes the Gröbner basis for a specific IP problem $IP_{A,C}(b)$ (rather than its generalisation $IP_{A,C}$). In this paper we propose a new algebraic algorithm for solving integer programming problems. The new algorithm, called the Minimised Geometric Buchberger Algorithm (MGBA), combines the Hosten and Sturmfels method (GRIN) and Thomas's truncated GBA to compute the fundamental segments of a IP problem $IP_{A,C}$ directly in its original space and also the truncated Gröbner basis for a specific IP problem $IP_{A,C}(b)$. We have carried out experiments to compare this algorithm with others such as the geometric Buchberger algorithm, the truncated geometric Buchberger algorithm, and the algorithm in GRIN. These experiments shows that the new algorithm offers significant performance improvement.

1 Introduction

In this paper, we consider the following integer programming problem:

$$IP_{A,C}(b) = \min \{ Cx : Ax = b, x \in \mathbb{Z}^n \}$$

where $C$ is the object vector in $\mathbb{R}^n$, $A$ is an $m \times n$ matrix of integers, and $b$ is a vector in $\mathbb{Z}^m$. We use $IP_{A,C}$ to denote a generic IP problem where $b$ is not taken into account.

IP problems characterise combinatorial optimisation problems where conventional numerical methods based on the hill-climbing technique can not be directly applied. Conventional methods for solving integer programming are based on searching algorithms where heuristics such as branch and bound can be applied to reduce the search space.

Recently, the tools of commutative algebra and algebraic geometry have brought new insights to IP via the theory of Gröbner bases [2]. The key idea is to encode an IP problem into a special ideal associated with the constraint matrix $A$ and the cost (object) function $C$. An important property of such an encoding is that its Gröbner bases correspond directly to the test sets of the IP problem. Thus, by employing an algebraic package such as MACAULAY [4] or MAPLE [1], the test sets of the IP problem can be directly computed. Using a proper test set (such as the minimal test set which corresponds directly to the reduced Gröbner base of the encoding ideal), the optimal value of the cost function can be computed by constructing a monotonic path from the initial non-optimal solution of the problem to the optimal solution. Thus, integer programming can be solved in a similar fashion to the simplex method for linear programming without using intensive heuristic searching algorithms.

The connection between test sets for integer programming and Gröbner bases of certain ideals was first established by Conti and Traverso [8]. The scheme involves two encoding mechanisms.
1. Encoding the cost function $C$ of $IP_{A,C}$ into a linear order on $\mathbb{Z}^n$. Thus, we define $\prec$ where:

$$x \prec y \iff \begin{cases} Cx < Cy & \text{if } x, y \in \mathbb{Z}^n \\ Cx = Cy & \text{if } x = y \end{cases}$$

2. Encoding $A$ into a polynomial ideal. This can be done by introducing a variable $u_i$ for the $i$th column of $A$ and forming an ideal:

$$I = \langle u_1^a - x_1, \ldots, u_n^a - x_n \rangle$$

where $a_1, \ldots, a_n$ are the columns of $A$.

With this translation, IP problems are transformed into solving the sub-algebra membership problem in $K[u]$ of determining whether $u^b$ belongs to the algebra generated by $u_1^a, \ldots, u_n^a$ [12].

In [9], Thomas proposed a geometric interpretation of Conti-Traverso method. The key idea of Thomas’s Geometric Buchberger Algorithm (GBA) is to relate the Gröbner bases of the encoded polynomial ideal of an IP to the notion of a test set for the IP. Each binomial is now directly interpreted as directed line segments, i.e., vectors, in a lattice of all feasible solutions of $IP_{A,C}$. The Buchberger algorithm is then directly applied to directed graphs, where nodes of the graph are lattice points corresponding to feasible solutions of $IP_{A,C}$ and the edges at the beginning correspond to the input basis of the binomial ideal $I$. Finding the reduced Gröbner basis amounts to rebuilding the graph such that the edges correspond to the members of the reduced Gröbner basis of $I$, which can be geometrically understood as a test set of the IP problem. Thus, by this graph, an optimal solution of $IP_{A,C}$ can be found along the directed path in the graph from a feasible solution. Thomas’s work provides not only a succinct understanding of an algebraic IP solver but also a practical computational procedure for its implementation. In particular, this “generate and test” approach provides great inherent parallelism. In [7], we presented a parallel implementation of GBA on a 32 node Fujitsu AP 1000+ MP machine. The experiment showed that the algebraic approach toward IP provides a very promising mechanism. It also showed that the new method can be improved in various ways.

The first problem is that the procedures formulated in Conti [8] and Thomas [9] are applied to an “extended” integer programming (EIP) with additional variables $(y, z)$, of the form:

$$\min \{ My + Cz \}$$

subject to $Iy + Ax = b$ and $(y, z) \in \mathbb{Z}^{m+m}$. $I$ is the $m \times m$ identity matrix and $M \in \mathbb{R}^m$ is a vector whose components have large magnitude (it is assured, without loss of generality, that all entries in $A, c$ and $b$ are nonnegative integers). In practice the additional variables will lead to a considerable increase in the space and time requirements of the algorithms considered.

The second problem is that the test set generated by both algorithms are generic in the sense that it is only determined by $A$ and $c$ for an IP system $IP_{A,C}$. Thus, the search space for computing the reduced Gröbner basis for such a generalised problem is quite large. In [10], Thomas proposed the “Truncated Gröbner basis” method by fixing $b$ to reduce the cardinality of the reduced Gröbner basis, but the size of Gröbner basis computed by the algorithm is still not optimal since the basis is for the EIP w.r.t a $IP_{A,C}(b)$, not the $IP_{A,C}(b)$ itself. So, many vectors in the reduced Gröbner basis are needed to move from an initial solution of EIP to an initial solution of IP. Therefore, ideally, we would like to work only with the variables of an IP system $IP_{A,C}$ directly and generate a set of fundamental segments for the problem itself that can be then refined to a (truncated) reduced Gröbner basis of $IP_{A,C}$ without introducing any extra variables. In [11], Hosten and Sturmfels proposed an algorithm in which a set of fundamental segments of $IP_{A,C}$ can be computed without going through EIP. This algorithm starts with a basis for the lattice $\text{ker}(A)$ and then proceeds to refine this to a set of fundamental segments for $IP_{A,C}$. But the algorithm is algebraic and generic. That is, the form of toric ideal $I_A$ is:

$$x^{\infty} - x^{-\infty}, u \in B(\text{lattice basis for ker}(A))$$

The space constructed is not a truncated space since the vector $b$ is not taken into account. Thus, the efficiency is still a problem when the method is applied to large scale IP problems due to the complexity of Buchberger’s algorithm.

In this paper we propose a new algebraic algorithm for solving integer programming. The new algorithm called the Minimized Geometric Buchberger Algorithm (MGBA), combines the Hosten and Sturmfels method (GRIN) and Thomas’s truncated GBA to compute the fundamental segments of an IP problem $IP_{A,C}$ directly in its original space and then the truncated Gröbner basis for the fixed $b$.

This paper is organized as follows. In section 2 and section 3, we give a brief sketch of the approach of the GRIN method [11] and the Truncated Gröbner basis method [10] separately. Section 4 describes some connections between the geometry of integer program and the combinatorics of Gröbner bases. In section 5, we present the new Buchberger algorithm to compute test set for integer programming. We also show some computational results to compare with other algorithms such as the algorithm in GRIN, the geometric Buchberger algorithm (GBA) and the truncated geometric Buchberger algorithm (TGBA) in section 6. Finally, we present a conclusion in section 7.

2 The Algorithm coded in GRIN

GRIN (Gröbner basis for Integer programming) is an experimental software system developed by Serkan Hosten and Bernd Sturmfels for computing the Gröbner basis of toric ideals, in particular, for solving integer programs using Gröbner bases. The algorithm coded in GRIN introduces the new method for computing the reduced Gröbner basis of the toric ideal which operates entirely in $k[z_1, \ldots, z_{\dim}]$ rather than in the auxiliary polynomial ring $k[y_1, \ldots, y_m, z_1, \ldots, z_{\dim}]$. In GRIN, two algorithms are implemented. Here we discuss only one.

Firstly we define the toric ideal and the ideal quotients as follows.

**Definition 2.1** The toric ideal $I_A$ is a binomial ideal constructed from matrix $A$:

$$I_A = \langle x^u - x^v : u, v \in \mathbb{Z}^n, u - v \in \text{ker}(A) \rangle$$

The algorithm in GRIN works in two stages: stage 1 computes the the generating set of toric ideal $I_A$, i.e the set of fundamental segments of integer programming $IP_{A,C}$, and...
then the reduced Gröbner basis $G_i$ with respect to the cost vector $c$, of the toric ideal $I_A$ (i.e. the test set of $IP_{A,C}$) based on the generating set; stage 2 solves $IP_{A,C}(b)$ by reducing (w.r.t. $G_i$) an arbitrary feasible solution of the program to the optimal solution.

**Definition 2.2** If $f$ is a polynomial in $k[x_1, \ldots, x_n]$ and $J \subseteq k[x_1, \ldots, x_n]$ is an ideal, then the following two subsets of $k[x_1, \ldots, x_n]$ are again ideals:

$$(J : f) = \{ g \in k[x_1, \ldots, x_n] : gf \in J \},$$

$$(J : f^\infty) = \{ g \in k[x_1, \ldots, x_n] : f^r g \in J \text{ for some } r \in \mathbb{N} \}.$$ 

A basis formula involving ideal quotients is $(1 : fg) = ((1 : f):g)$. A general method for computing Gröbner bases of the ideals from generators of $J$ can be found in [5]. If $J$ is an homogeneous ideal and $f$ is one of the variables, say, $f = x_n$, then the algorithm for computing the Gröbner bases of the ideals from $J$ is provided by the following lemma in [3].

**Lemma 2.1** Fix the graded reverse lexicographic term order induced by $x_1 > \ldots > x_n$, and let $G$ be the reduced Gröbner basis of a homogeneous ideal $J \subseteq k[x_1, \ldots, x_n]$. Then the set

$$G' = \{ f \in G : x_n \text{ does not divides } f \} \cup \{ f/x_n : f \in G \text{ and divides } f \}$$

is a Gröbner basis of $(J : x_n)$. A Gröbner basis of $(J : f^\infty)$ is obtained by dividing each element $f \in G$ by the highest power of $x_n$ that divides $f$.

The term order used in Lemma 2.1 makes sense whenever the ideal $J$ is homogeneous with respect to some positive grading $d_i = d > 0$. By iterating the Gröbner basis computation $n$ times with respect to different reverse lexicographic term orders, that is, by applying Lemma 2.1 one variable at a time, one can compute the ideal quotient

$$(J : (x_1, x_2, \ldots, x_n)^\infty) = ((\cdots (J : x_1^\infty) : x_2^\infty) \cdots) : x_n^\infty)$$

So, if we find the relationship between the toric ideal $I_A$ and ideal quotient, we can compute the generating set of $I_A$ and Gröbner basis of $I_A$ based on the generating set. We assume that $A - \{a_1, \ldots, a_n\}$ is a subset of $N^n$, so that the toric ideal $I_A$ is homogeneous. We have a method to check whether $I_A$ is homogeneous from the following lemma in [3].

**Lemma 2.2** The ideal $I_A$ is homogeneous if and only if there exists a vector $w \in Q^n$ such that $w \cdot a_i = 1$ for $i = 1, \ldots, n$.

Let $\text{ker}(A) \subseteq \mathbb{Z}^n$ denote the integer kernel of the $d \times n$-matrix $A$ with column vectors $a_i$. With any subset $C$ of the lattice $\text{ker}(A)$ we associate a subideal of $I_A$:

$$J_C := \langle x^v - x^w : v \in C \rangle$$

where $v = v^+ - v^- : v \in C >$ is the usual decomposition into positive and negative part.

**Lemma 2.3** A subset $C$ spans the lattice $\text{ker}(A)$ if and only if

$$(J_C : (x_1, \ldots, x_n)^\infty) = I_A$$

From Lemma 2.1 and Lemma 2.3, we can prove the following Proposition.

**Proposition 2.1** Let $J_0 := \langle x^v - x^w : v \in C \rangle$ and $J_i := (J_{i-1} : x_i^\infty) \ (i = 1, \ldots, n)$ with the graded reverse lexicographic term order by making $x_i$ the reverse lexicographically cheapest variable. Then $J_n$ is a generating set for $I_A$.

The lemmas and proposition stated above give the following algorithm which computes a Gröbner basis of a toric ideal.

**Algorithm**

1. Find any lattice basis $B$ for $\text{ker}(A)$.

2. (Optional) Replace $B$ by a reduced lattice basis $B_{red}$.

3. Let $J_0 := \langle x^v - x^w : v \in B_{red} \rangle$.

4. For $i = 1, \ldots, n$ Compute $J_i := (J_{i-1} : x_i^\infty)$ using Lemma 2.1, that is, by making $x_i$ the reverse lexicographically cheapest variable.

5. Compute the reduced Gröbner basis of $J_n = I_A$ for the desired term order. If the term order is obtained from an objective function $c$, then the computed reduced Gröbner basis is the minimal test set of $IP_{A,C}$.

**Example:**

Let $d = 4, n = 8$ and consider the matrix $A$

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 & 0 & 1 & 4 & 5 \\ 2 & 3 & 4 & 1 & 1 & 4 & 5 & 0 \\ 3 & 4 & 1 & 2 & 4 & 5 & 0 & 1 \\ 4 & 1 & 2 & 3 & 5 & 0 & 1 & 4 \end{bmatrix}$$

Here $A$ is the $4 \times 8$ matrix all of whose columns add up to 10. According to Lemma 2.2, the toric ideal $I_A$ is homogeneous, where $w = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2})$.

**STEP 1 and STEP 2:**

Compute basis for the lattice $\ker(A)$. In this case we get the reduced basis $B_{red}$ as follow.

$$B_{red} = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & -1 & 0 & -1 \\ 0 & 1 & 0 & -3 & 0 & 0 & 0 & 2 \\ 1 & 0 & 1 & 0 & -1 & 0 & -1 & 0 \\ 2 & 0 & -2 & 0 & -1 & 0 & 1 & 0 \end{bmatrix}$$

**STEP 3 and STEP 4:**

We need to make eight Gröbner basis computations with respect to certain reverse lexicographic orders, starting with $J_0 := \langle x^v - x^w, v \in B_{red} \rangle$.

After each Gröbner basis computation we need to divide out certain variables. What we get after these eight Gröbner bases computations is the set of fundamental segments of $A$.

For example, by splitting the vectors of $B_{red}$ into positive and negative part, we get the binomial ideal $J_0$ associated with $B_{red}$:

$$J_0 := \langle x_2 x_4 - x_2 x_8, x_2 x_8 + x_1 x_2, x_1 x_2 - x_2 x_7, x_2 x_7 - x_2 x_8 \rangle$$

Entering the loop in Step 4, we first compute the reduced Gröbner basis for $J_0$ with respect to the reverse lexicographic order that makes $x_1$ the cheapest variable. Here $x_1 < x_2 < x_3 < x_4 < x_5 < x_6 < x_7 < x_8$. The result is
\[ G_0 = \{ x_1^2 x_2 - x_1 x_3^2, x_1^3 - x_3 x_2, x_2 x_3^2 - x_1 x_2^3, x_2 x_5 - x_3 x_1, x_2 x_6 - x_4 x_2 \} \]

Next we divide each binomial in \( G_0 \) by \( x_1 \) whenever possible. So for example, \( x_2 x_1^2 - x_1^2 x_2 \) when divided by \( x_1 \) gives \( x_3^2 - x_2 x_1^2 \); and none of the other can be divided by \( x_1 \) (the cheapest variable in the above order).

Then we get a new set \( J_1 \) which consists of all the binomials in \( G_0 \) divided by \( x_1 \) whenever possible.

\[ J_1 = \{ x_3^2 - x_2^2, x_2 x_3 - x_2^2, x_2^3 - x_2^2, x_2 x_5 - x_3 x_1, x_2 x_6 - x_4 x_2 \} \]

Now, we now compute the reduced Gröbner basis \( G_1 \) for \( J_1 \) by using the reverse lexicographic order that makes \( x_2 \) the cheapest variable. For example, we use the order \( x_2 > x_3 > x_4 > x_5 > x_7 > x_6 > x_5 > x_4 > x_3 > x_2 \). The result is:

\[ G_1 = \{ x_2 x_3^2 - x_2^2 x_3, x_2 x_5 - x_3 x_1, x_2 x_6 - x_4 x_2, x_2 x_7 - x_3 x_2, x_2 x_8 - x_3 x_3 \} \]

Dividing each binomial in \( G_1 \) by \( x_2 \) whenever possible, we get:

\[ J_2 = \{ x_3^2 - x_3 x_1, x_3 x_2 - x_3, x_2 x_3 - x_3, x_2^2 - x_2, x_2^2 - x_2, x_2 x_2 - x_2, x_2 x_3 - x_2, x_2 x_4 - x_2, x_2 x_5 - x_2 \} \]

STEP 5:

Now, we can use \( J_3 \) as a generating set to compute the reduced Gröbner basis of \( I_A \) with a fixed term order. Here, the reduced Gröbner basis of \( I_A \) with respect to the lexicographic term order given by \( x_2 > x_3 > x_4 > x_5 > x_6 > x_7 > x_8 \) equals:

\[ G = \{ x_3 - x_2^3, x_2 x_4 - x_3, x_2 x_5 - x_3, x_2 x_6 - x_3, x_2 x_7 - x_3, x_2 x_8 - x_3, x_2 x_9 - x_3, x_2 x_10 - x_3, x_2 x_11 - x_3 \} \]

3 Truncated Gröbner bases

The computation of the entire reduced Gröbner basis associated with the family of programs \( IP_{A,C} \), is often expensive or impossible. In practice, we are often interested in solving \( IP_{A,C} \) for a fixed right hand side vector \( b \), which typically requires only a subset of the entire Gröbner basis. In [10], Thomas proposed a truncated Buchberger algorithm called b-Buchberger algorithm for toric ideals that finds a sufficient test set for \( IP_{A,C} \). This is a proper subset of the reduced Gröbner basis of \( I_A \), with respect to \( c \). So, by the algorithm, we can produce a minimal test set for the family of integer programs whose right hand side vector is smaller than or equal to \( b \) in a specific sense, which greatly improves the computation.

Lemma 3.1 The toric ideal \( I_A = \bigoplus_{c \in C(A)} I_A(\beta) \) where:

\[ C(A) = \{ \sum_{i=1}^{n} m_i \alpha_i : m_i \in \mathbb{N} \} \subseteq \mathbb{N}^n \] and \( I_A(\beta) \) is the \( k \)-vector space spanned by the binomials \( \{ x^u - x^v : Au - \beta, u, v \in \mathbb{N} \} \).

\( C(A) \) is a monoid and the \( IP_{A,C} \) is feasible if and only if \( b \) lies in \( C(A) \). Let \( M \) denote the set of all monomials in \( k[x] = k[x_1, ..., x_n] \) where \( k \) is a field. The monoids \( M \) and \( N^m \) are isomorphic via the usual identification of a monomial \( x^m \) with its exponent vector. Under this identification, the monoid homomorphism \( \pi_A \) induces a multivariate grading of \( M \) and hence \( k[x] \), where the \( \pi_A \)-degree of \( x^m \) denoted \( \pi_A(x^m) = \pi_A(\alpha) = Au \in C(A) \). Let \( M(f) \) denote the monomials in a polynomial \( f \in k[x] \).

Definition 3.1 A polynomial \( 0 \neq f \in k[x] \) is said to be \( \pi_A \)-homogeneous if \( \pi_A(s) = \pi_A(t) \) for all monomials \( s, t \in M(f) \). The \( \pi_A \)-degree of a homogeneous polynomial \( f \), denoted \( \pi_A(f) \), equals the \( \pi_A \)-degree of any monomial in \( M(f) \).

With above Lemma 3.1 and Definition 3.1, we have the following lemma:

Lemma 3.2 The toric ideal \( I_A \) is homogeneous with respect to the grading induced by \( \pi_A \).

Associated with the monoid \( C_N(A) \) there is a “natural” partial order \( \preceq \) such that for \( b_1, b_2 \in C_N(A), b_1 \preceq b_2 \) if and only if \( b_1 - b_2 \in C_N(A) \). Notice that when \( C_N(A) = N^m \), the partial order \( \preceq \) coincides with the component wise partial order \( \preceq \), where \( b_1 \preceq b_2 \) if and only if \( b_1 - b_2 \geq 0 \).

The lemmas stated above give the following algorithm:

b-Buchberger algorithm for toric ideals:

Input: A finite homogeneous binomial basis \( F \) of \( I_A \) and the refined cost vector \( c \).

Output: A truncated (with respect to \( b \)) Gröbner basis of \( I_A \) with monomial order given by \( c \).

\[ i = -1, G_0 = F \]

Repeats:

\[ i = i + 1 \]

\[ G_{i+1} = G_i \cup \{ \text{norm} \, f_{A,c} \cdot (S - \text{bin}_i(g_1, g_2) : g_1, g_2 \in G_i, \pi_A(S - \text{bin}_i(g_1, g_2)) \preceq \{0\} \} \]

Until \( G_{i+1} = G_i \)

Reduce \( G_{i+1} \) modulo the leading monomials of its elements.

The algorithm b-Buchberger described above considers an \( S \)-binomial \( g = x^u - x^v \) for reduction if and only if \( \pi_A(g) = Au = A \preceq b \). This amounts to checking feasibility if the system \( \{ x \in \mathbb{N}^n : Ax = b - Au \} \) which is as hard as solving the original integer program \( IP_{A,C} \). Therefore, in order to implement the algorithm in practice, Thomas proposed two relaxations of the above check. Consider the \( S \)-binomial \( g = x^u - x^v \) in \( I_A \) for reduction if:

- \( b - Au \in C(A) \) where \( C(A) = \{ ax : x \in R_+^n \} \), i.e., check feasibility of the linear programming relaxation of the original check.
- \( b - Au \in C(A) \cap ZA \) where \( ZA = \{ ax : x \in \mathbb{Z}^n \} \). This is a relaxation of the original check since in general, \( C_N(A) \) is strictly contained in \( C(A) \cap ZA \).
In our implementation, we use the second check by introducing the Hermite normal form, see the Minimised Buchberger Geometric Algorithm in section 5. When \( C_N(A) = N \), we just use \( b - Au \geq 0 \), as the following example.

Example (continued) Consider the example in section 2. Suppose we are interested only in right hand side vectors \( b = (n_1, n_2, n_3, n_4) \) which satisfy \( n_1 \leq 8 \) and \( n_2 \leq 8 \). According to the condition, the truncated \( \text{Gr"obner} \) basis equals \( \{ x_1 - x_3 x_2, x_3 x_4 - x_5 x_7, x_2 x_4 - x_6 x_8 \} \). Because here \( C_N(A) = N \), the degree \( Au \) of these three binomials are \( (3, 9, 12) \), \( (4, 6, 4) \) and \( (6, 4, 6, 4) \), which satisfy \( b - Au \geq 0 \), while for the other seven binomials, their degree \( Au \) are so large that \( b - Au < 0 \). Thus, they lie outside of \( C_N(A) \).

4 A Geometry of the Buchberger Algorithm

The Buchberger algorithm for integer programming is just a special case of the general Buchberger algorithm. However, there are many features in the special situation of "toric ideal" we consider here. First of all, in a toric ideal, \( S \)-pair of any two binomials is a binomial. Also the reduction of a binomial by a binomial leads to a binomial. Thus the reduced \( \text{Gr"obner} \) basis of the ideal will consist of only binomials which always have the form \( z^+ - z^- \) with \( a \in \ker(A) \). With these features, we can get an entirely geometric formulation of the algorithm dealing with only lattice vectors in \( Z^n \) rather than polynomials. The translation from binomials to their correspondent vectors is straightforward. For example, we translate \( x_1 x_3 x_2 x_3 - x_5 x_7 x_4 x_6 \) directly into the vector \([2, 0, 1, 0, 3, 0, 0, 3, 0, 0, 2, 0] \).

The following two algorithms compute the reduced \( \text{Gr"obner} \) basis of a lattice, that is, the finite subset \( \tilde{G} \subseteq L^{n,0} \) corresponding to the reduced \( \text{Gr"obner} \) basis of \( I_A \). Also, by the two algorithms we can interpret the reduction steps geometrically and compute the reduced \( \text{Gr"obner} \) basis with geometric term.

Algorithm 4.1 "Reduction"

The following algorithm computes the reduction of a vector \( f \in Z^n \) by a set \( \tilde{G} \) of integer vectors.

INPUT \( \tilde{G} \subseteq L^{n,0} \), \( f \geq 0 \).

REPEAT if there is some \( g \in \tilde{G} \) with \( g^+ \leq f^+ \), then replace \( f \) by \( f - g \).

if there is some \( g \in \tilde{G} \) with \( g^- \leq f^- \), then replace \( f \) by \( f + g \).

OUTPUT \( f := f \).

Algorithm 4.2 "Buchberger algorithm on lattice vectors"

The following algorithm computes the reduced \( \text{Gr"obner} \) basis of the lattice \( L \), for a fixed term order \( \succ \).

First Step: Construct a \( \text{Gr"obner} \) basis

INPUT A basis \( \{ a_1, \ldots, a_n \} \subseteq L \) such that the binomials \( z^+ - z^- \) generate \( I_A \).

SET \( \tilde{G}_0 := 0 \), \( \tilde{G} := \{ a_1, \ldots, a_n \} \).

REPEAT While \( \tilde{G}_{id} \neq \tilde{G} \), repeat the following steps:

\( \tilde{G}_{id} := \tilde{G} \)

\( (S \text{-pairs}) \) construct the pairs \( g := a - a^\prime \geq 0 \) with \( a, a^\prime \in \tilde{G} \).

(reduction) reduce the vectors \( g \) by the vectors in \( \tilde{G}_{id} \).

If \( g \neq 0 \), set \( \tilde{G} := \tilde{G} \cup \{ g \} \).

Second Step: Construct a minimal \( \text{Gr"obner} \) basis

REPEAT if for some \( g \in \tilde{G} \) the point \( g^+ \) can be reduced by some \( g^\prime \in \tilde{G} \setminus g \), then delete \( g \) from \( \tilde{G} \).

Third Step: Construct the reduced \( \text{Gr"obner} \) basis

REPEAT if for some \( g \in \tilde{G} \) the point \( g^- \) can be reduced by some \( g^\prime \in \tilde{G} \setminus g \), then replace \( g \) by the corresponding reduced vector: \( \tilde{G} := \tilde{G} \setminus g \).

OUTPUT \( \tilde{G}_{red} := \tilde{G} \).

5 The Minimised Geometric Buchberger Algorithm

Combining the GRIN method, the truncated \( \text{Gr"obner} \) bases and geometric Buchberger algorithm together, we propose a new Buchberger algorithm for integer programming in which we can obtain considerable improvements in the efficiency and applicability of the \( \text{Gr"obner} \) basis approach. We formulate the algorithm in the original space and interpret the reduction steps geometrically.

From the objective function \( c \) we obtain a linear order on \( R^n \) as follows: we choose the lexicographic term order \( \prec_0 \) as a "tie breaker" on the points that have the same objective function value under \( c \); that is, we define

\[
\begin{align*}
 x \prec_0 y &\iff \begin{cases} 
 c^T x < c^T y, \\
 c^T x = c^T y \text{ and } x \prec y
\end{cases}
\end{align*}
\]

Throughout the section we use the following notation. \( N \) denotes the set \( \{ 1, \ldots, n \} \). For a vector \( v \in Z^n \) we denote by \( v^+ \) the vector with \( v_i^+ = v_i \) if \( v_i \geq 0 \) and \( v_i^+ = 0 \) otherwise. Accordingly, \( v^- \) is the vector with \( v_i^- = -v_i \) if \( v_i \leq 0 \) and \( v_i^- = 0 \) otherwise. Clearly \( v = v^+ - v^- \).

Minimised Buchberger Geometric Algorithm

1. Compute lattice basis \( B \) for \( \ker(A) \)

\begin{enumerate}
\item Use the Hermite normal form algorithm in [6] to compute \( C \) and \( H \), such that

\[
AC - (H \cdot 0)
\]

where \( C \in Z^{n \times m}, H \in Z^{m \times m} \)

\[
H = \begin{cases}
 h_{ij} = 0 & i < j \\
 h_{ij} > 0 \\
 h_{ij} \leq 0 & |h_{ij}| = h_{ij} \quad i > j
\end{cases}
\]

\item By the Hermite theorem in [6]

\[
x = C_1 H^{-1} b + C_2 z
\]

where \( C = (C_1, C_2), \quad C_1 \in Z^{n \times m}, \quad C_2 \in Z^{m \times (n-m)} \)

\[
z \in Z^{n-m}
\]

\[
Ax = 0, \quad \text{i.e.} \quad b = 0, \quad x = C_2 z, \quad \text{so we get} \quad B = C_2
\]
\end{enumerate}
2. Reduce $B$ into reduced lattice basis $B_{red}$
   Here we can use the reduced basis algorithm in [6] to compute the reduced lattice basis $B_{red}$.

3. Compute a generating set for the toric ideal $I_A$
   (1) Let $J_0 := \langle x^u - x^w : u \in B_{red} \rangle$ and interpret each binomial in $J_0$ as a vector by reading off its exponents.
   (2) For $i = 1, 2, \ldots, n$, Compute $J_i := \langle x^s, \ldots, x^u \rangle$ geometrically by making $x_i$ the reverse lexicographically cheapest variable.

4. Compute the truncated GB $G_i(b)$ of $I_A$ with order $<_e$

   **INPUT:** generating set $J_n$ of toric ideal $I_A$
   (1) Construct a Gröbner basis
   In the first step of Algorithm 4.2 we add a related check of the truncated Gröbner basis into the computation of $S$-vector:
   $b - Au \in (C(A) \cap ZA$ where $ZA = \{ Az : z \in Z^n \}$
   By Hermit's theorem, we can check whether there is feasible solution for $S = \{ z \in Z^n : Az = b - Au \}$ by:
   $S$ is not empty if and only if $H^{-1}(b-Au) \in Z^n$
   (2) Construct a minimal Gröbner basis
   This step is same as the second step of Algorithm 4.2.
   (3) Construct the reduced Gröbner basis
   This step is same as the third step of Algorithm 4.2.

   In this algorithm, from step 3 “Compute a generating set for toric ideal $I_A$”, we translate all computation into geometric terms. For the example of section 2, we interpret each binomial of $J_0$ as a vector by reading off its exponents. For example, $x_2x_3 - x_4x_5$ corresponds to the vector $[(101000000) \rightarrow \{000000001\}]$. So, we translate $J_0$ into a set of vectors as follows:
   $J_0 = \{(101000000) \rightarrow \{000000001\}, (000000002) \rightarrow \{000000000\}, (101000000) \rightarrow \{000001010\}, (200000000) \rightarrow \{020010000\}\}$
   Then we use Algorithm 4.2 to compute the reduced Gröbner basis for $J_0$ with respect to reverse lexicographic order that makes $x_1$ the cheapest variable. Here $x_2 < x_3 < x_4 < x_5 < x_6 < x_7 < x_8$. The result is
   $G_0 = \{(103000000) \rightarrow \{200000000\}, (003000000) \rightarrow \{010000000\}, (002010000) \rightarrow \{200000000\}, (000001010) \rightarrow \{101000000\}, (000000000) \rightarrow \{010010000\}\}$
   Next we divide each vector in $G_0$ by $x_1$ whenever possible, just as removing the common factor $x_1$ from two monomials. So for example $[(103000000) \rightarrow \{200000000\}]$ when divided by $x_1$ gives $[003000000] \rightarrow \{010000000\}$ and none of the other can be divided by $x_1$. (the cheapest variable in the above order).
   Then we get a new set $J_1$ which consists of all the vectors in $G_0$ divided by $x_1$ whenever possible.
   
   $J_1 = \{(003000000) \rightarrow \{100000000\}, (000300000) \rightarrow \{010000000\}, (002010000) \rightarrow \{200000000\}, (000001010) \rightarrow \{101000000\}, (000000000) \rightarrow \{010100000\}\}$
   
   Now we compute reduced Gröbner basis $G_1$ for $J_1$ by using the reverse lexicographic order that makes $x_2$ the cheapest variable. For example we use the order $x_1 > x_8 > x_7 > x_6 > x_5 > x_4 > x_3 > x_2$. The result is
   $G_1 = \{(003000000) \rightarrow \{100000000\}, (000300000) \rightarrow \{010000000\}, (002010000) \rightarrow \{200000000\}, (000001010) \rightarrow \{101000000\}, (000000000) \rightarrow \{010100000\}\}$

   Dividing each binomial in $G_1$ by $x_2$ whenever possible, we get $J_2$
   $J_2 = \{(003000000) \rightarrow \{100000000\}, (000300000) \rightarrow \{010000000\}, (002010000) \rightarrow \{200000000\}, (000001010) \rightarrow \{101000000\}, (000000000) \rightarrow \{010100000\}\}$

   Then we can repeat the process, every time computing reduced Gröbner basis $G_i$ for $J_i$ by using reverse lexicographic order that makes $x_{i+1}$ the cheapest variable and then dividing each vector in $G_i$ by $x_{i+1}$ to get $J_{i+1}$. Finally we get $G_n$ that is the fundamental ideals of $A$ i.e. the generating set of the toric ideal $I_A$

   $G_n = \{(040000000) \rightarrow \{100000000\}, (003000000) \rightarrow \{100000000\}, (010100000) \rightarrow \{000001010\}, (002010000) \rightarrow \{200000000\}, (001020000) \rightarrow \{300000000\}, (000020100) \rightarrow \{000020000\}, (000012000) \rightarrow \{030000000\}, (000010100) \rightarrow \{101000000\}\}$

   In step 4, we can use $J_n$ as the fundamental ideals to compute truncated reduced Gröbner basis of $I_A$ with fixed right hand side $b$ and cost function $c$. Here, we suppose the right hand side vectors $b = (b_1, b_2, b_3, b_4)$ which satisfy $b_1 \leq 8$ and $b_2 \leq 8$ and the $<_e$ is the lexicographic term order given by $x_1 > x_2 > x_3 > x_4 > x_5 > x_6 > x_7 > x_8$. So, the test set for IP is the truncated reduced Gröbner basis is
   $G_i(b) = \{(300000000) \rightarrow \{000102000\}, (101000000) \rightarrow \{000001010\}, (010100000) \rightarrow \{000001000\}\}$

   **Theorem 5.1** The algorithm stated above terminates after a finite number of steps. The output of the algorithm is the unique minimal test set for the integer programming $IP_{A,C}(b)$.

   **Proof.** Finiteness of the algorithm is clear since the Buchberger algorithm and b-Buchberger algorithm all terminate in finitely many steps.

   By Proposition 2.1, we obtain the generating set of the toric ideal $I_A$ in step 3. Because the generating set of the toric ideal $I_A$ is a set of fundamental ideals for $IP_{A,C}$. The geometric Buchberger algorithm and b-Buchberger algorithm guarantee that we can obtain the truncated reduced Gröbner basis $G_i(b)$ for $IP_{A,C}(b)$ with the term order $<_e$ in
step 4. Now, we study $G, (b)$ from a completely geometric point of view. As in [9], we can build a connected, directed graph for only one fiber (b-fiber) of $IP_{A,C}(b)$. The nodes of the graph are all the lattice points in the fiber and the edges are the translations of elements in $G, (b)$ by nonnegative integral vectors. By Theorem 2.1.8. in [9], the graph in a fiber is a unique sink at the unique optimum in the fiber. So, the reduced Gröbner basis $G, (b)$ is a test set for $IP_{A,C}(b)$. By Corollary 2.1.30. in [9], we can prove $G, (b)$ is the unique minimal test set for $IP_{A,C}(b)$, depends on $A, <$, and $b$.

6 Experiments and Comparisons

We have developed an implementation system MGBAS by language C on a Sun UltraSpArc workstation on which we ran the Minimum Geometric Buchberger Algorithm (MGBA) on randomly generated matrices $A$ of various sizes (ranging from $3 \times 7$ to $8 \times 16$) with nonnegative entries in a range between 0 and 20. We generated random right hand sides $b$ to compute truncated Gröbner basis $G, (b)$. Once $G, (b)$ was found, we selected many random feasible solutions $v$ (by $x = C_1 H^{-1} b + C_2 z$), and we reduced each $v$ to the corresponding optimal solution $u$ using the truncated Gröbner basis $G, (b)$. For each test instance $(A, c, b)$ three comparisons were made with the geometric Buchberger algorithm, the truncated geometric Buchberger algorithm, and the algorithm in GRIN.1

We also have implemented GBAS and the TGBAS system with language C for the geometric Buchberger algorithm and the truncated geometric Buchberger algorithm respectively on the Sun UltraSpArc. For each instance in the experiments, we can see the reduced Gröbner basis of GBAS is the biggest one among all algorithms because of the introduction of additional variables. Also the running time is longest. For the TGBAS, because the algorithm coded in it computes the reduced Gröbner basis by fixing $b$, we can see the size of the reduced Gröbner basis in TGBAS is less than that in GBAS. But it is greater than that in GRIN and MGBA, because the algorithm coded in TGBAS still introduces the additional variables to compute the reduced Gröbner basis.

For the algorithm coded in GRIN, we did not have the GRIN system, so we were unable to compute the running times. But we can count on the cardinality of reduced Gröbner basis for the algorithm for each instance. Actually, our new algorithm MGBA is the combination of the GRIN method and the truncated Gröbner bases method. So, from MGBAS, we can also compute the reduced Gröbner basis of GRIN for each instance. From these experiments, we show that the cardinality of the reduced Gröbner basis of GRIN is greater than that of the MGBA but less than that of the GBA and TGBA.

In the table 1, we present a typical example from our examples, which were performed on a SUN UltraSpArc. The $3 \times 7$ matrix $A$ has entries which are chosen uniformly at random from $[0, 20]$. The reduced lattice basis $B_{red}$ of $ker(A) \cap Z^7$ consists of the rows of the second matrix $L$. It took 0.24 seconds to find a generating set of $IA$. We chose a cost function $c$ and right side $b$, then computed truncated Gröbner basis $G, (b)$ from the generating set. The performance is dramatically improved to only 0.1 second. The size of $G, (b)$ is also reduced to 17. Finally we chose a feasible vector $v$ and reduced it modulo $G, (b)$ to get the optimal solution $u$. The reduction took less than 0.02 seconds.

As a comparison we solved the same problem using system GBAS, TGBAS, and the algorithm in GRIN. For GBAS, the size of the reduced Gröbner basis $G, (b)$ is 56 and the running time is 1462.96 seconds. For TGBAS, the size of the reduced Gröbner basis $G, (b)$ is 272 and the running time is 55.96 seconds. For the algorithm in GRIN, the size of the reduced Gröbner basis $G, (b)$ is 31.

$$A = \begin{pmatrix} 3 & 1 & 11 & 2 & 3 & 5 & 3 \\ 4 & 5 & 0 & 1 & 7 & 4 & 6 \\ 5 & 6 & 1 & 9 & 2 & 3 & 3 \end{pmatrix}$$

$$L = \begin{pmatrix} -5 & 0 & 0 & 2 & 2 & 1 & 0 \\ -5 & -4 & -1 & 0 & 0 & 0 & 0 \\ -5 & 7 & 1 & -2 & -3 & 2 & 0 \\ -2 & 5 & 1 & -2 & -3 & 0 & 1 \end{pmatrix}$$

We conclude with a summary of the results of the experiments we have carried out with MGBAS, in the table 2. The range of the entries used in the problems are given in the second column. The third and fourth columns give the size of truncated Gröbner basis and the time for computing it for each problem with system MGBAS. The fifth and sixth columns give the size of the reduced Gröbner basis and the time for computing it with system GBAS. The seventh and eighth columns give the size of reduced Gröbner basis and the time for computing it with system TGBAS. The last column gives the size of reduced Gröbner basis with GRIN system. The timings are in CPU seconds on a SUN UltraSpArc.

---

1 Since we cannot obtain the GRIN system, we are unable to compare the execution performance with the system. However, from the performance figure shown in [11], we can predicate the new algorithm would be more efficient.
7 Conclusions

We have proposed a new algorithm Minimised Geometric Buchberger Algorithm for integer programming. It combines the GRIN method and the truncated Gröbner bases method to compute a generating set of the Gröbner bases in the original space and refine it into a minimal test set i.e. a truncated reduced Gröbner basis of \( IP_{\alpha,b} \) with fixed right hand side. Our primary experiments indicate that the algorithm is much faster than others such as the geometric Buchberger algorithm, the truncated geometric Buchberger algorithm and the algorithm in GRIN.

Yet we are still far from applying our algorithm to large scale problem instances. There is still much room for further improvements in the efficiency and applicability of the Gröbner basis approach. The future research includes the improvement of the implementation and the parallelisation of the algorithm. Due to the high degree of inherent parallelism in the algorithm, we expect that the parallelisation will result in a practical IP solver.

References


