ABANT IZZET BAYSAL UNIVERSITY

THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

DEPARTMENT OF MATHEMATICS

GRÖBNER BASES AND INTEGER PROGRAMMING

MASTER OF SCIENCE

ZÜBEYDE DİNÇ

BOLU, DECEMBER 2017

APPROVAL OF THE THESIS

GRÖBNER BASES AND INTEGER PROGRAMMING submitted by **Zübeyde DİNÇ** in partial fulfillment of the requirements for the degree of **Master of Science** in **Department of Mathematics, The Graduate School of Natural and Applied Sciences of ABANT IZZET BAYSAL UNIVERSITY** in **29/12/2017** by

Examining Committee Members	**Signature**
Supervisor Assoc. Prof. Dr. Erol YILMAZ Abant Izzet Baysal University	.....
Member Assoc. Prof. Dr. Ergün YARANERİ Istanbul Technical University	.....
Member Assist. Prof. Dr. Sibel Kılıçarslan CANSU Abant Izzet Baysal University	.....

Graduation date: ...../...../20....

Assoc. Prof. Dr. Ömer ÖZYURT .....

Director of Graduate School of Natural and Applied Sciences

To my family, to my supervisor, and to my friends

I hope this thesis is useful not only for theoric math but also applied math for the precious searchers..

DECLARATION

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Zübeyde DİNÇ

ABSTRACT

GRÖBNER BASES AND INTEGER PROGRAMMING
M.S. THESIS
ZÜBEYDE DİNÇ,
ABANT IZZET BAYSAL UNIVERSITY GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
DEPARTMENT OF MATHEMATICS
(SUPERVISOR : ASSOC. PROF. DR. EROL YILMAZ)

BOLU, DECEMBER 2017

Solving integer programming problems using Gröbner bases are investigated. The main tool of this process is the toric ideals. A new method for computation of the toric ideals is proposed.

KEYWORDS: Integer programming problems, Gröbner bases, toric ideals, saturation, pseudo division

ÖZET

GRÖBNER TABANLARI VE TAMSAYI PROGRAMLAMA
YÜKSEK LİSANS TEZİ
ZÜBEYDE DİNÇ,
ABANT İZZET BAYSAL ÜNİVERSİTESİ FEN BİLİMLERİ ENSTİTÜSÜ
MATEMATİK ANABİLİM DALI
(TEZ DANIŞMANI : DOÇ. DR. EROL YILMAZ)

BOLU, ARALIK 2017

Gröbner tabanları kullanılarak tamsayı programlama probleminin çözümü araştırılmıştır. Bu işlemin ana aracı torik idealleridir. Torik ideallerin hesaplanması için yeni bir yöntem önerilmiştir.

ANAHTAR KELİMELER: Tamsayı programlama problemleri, Gröbner tabanları, toric idealleri, doyum, sahte bölme

TABLE OF CONTENTS

1. INTRODUCTION

The concept of Gröbner basis for polynomial ideals was first introduced by Buchberger (1965) in his Ph.D. thesis and named after his advisor Gröbner. Buchberger did not only define the concept, he also described an algorithm to obtain a Gröbner basis from any given set of generator of a polynomial ideal. Gröbner basis is a powerful tool solving many problems in polynomial rings. After implementation of Buchberger's algorithm on most computer algebra systems, it was shown that Gröbner basis can help to solve problems in different areas such as robotics, cryptography, signal processing and petroleum industry. The basic idea is to reformulate a particular problem as a question about polynomials, then solve it by Gröbner bases.

The general linear programming problem is a special kind of optimization problem in which a linear objective function is either maximized or minimized subject to some constraints. An integer program problem differ from a linear program problem in that the coefficients and solutions are required to be integer. In fact, it is difficult to find integer solutions to an optimization problem. There are several algorithms for solving integer programming problems. The most of them are extensions of algorithms for linear programming problems. For a detailed information for integer programming problems see (Taha, 2003).

The Gröbner basis techniques gave a new perception into integer programming problems. The connection between integer programming and Gröbner bases was first given by (Conti and Traverso, 1991). They found a Gröbner basis of certain ideal related to coefficient matrix of a given integer programming problem. These ideals are called toric ideals of integer matrices. Detailed explanation of this method can be found in (Adams). The main shortcut of this idea is the size of Gröbner basis to be computed. Because they needed extra variables to obtain a generating set for the corresponding toric ideal. Since Buchberger's algorithm is very sensitive to the number of variables, their method is not efficient for the large integer programming problems. Hosten and Sturmfels (1995) design a new method, so called GRIN, to avoid additional variables. We can refer to Sturmfels (1996) for the complete treatment of this method. Thomas (1995) aproached the same problem in geometric point of view. Thomas and Weismantel (1997) defined the truncated Gröbner bases for integer programming. Then Li et al (2001) combined these two methods and found

a new algorithm for integer programming problems. Ikegami and Kaji (2003) extended Conti's algorithm to solve integer programming with modulo arithmetic conditions. Blanca and Puerto (2009, 2011) use same techniques for solving integer programing problems with several objective functions.

In this thesis, we will explain Gröbner basis techniques for integer programming problems. We give some basic terminology about Gröbner bases in the second chapter. Toric ideals which connects integer programming problems to Gröbner bases are considered in the third chapter. We will give a new method for computing toric ideals there. This is the most important contribution we made to the subject. Solution of integer programming problems using Gröbner basis techniques illustrated with examples in the following chapter. We use the computer algebra system Mathematica for the computations throughout the thesis.

2. GRÖBNER BASIS

Monomial Ordering

Let k be a field. The division algorithm in $k[x]$ is a well-known process. In order to generalize this division algorithm into multivariable polynomial ring $k[x_1, \ldots, x_n]$, we have to define an ordering on monomials.

Definition 2.1.1. *A monomial ordering is a total order relation $>$ on the set of monomials $x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_n^{\alpha_n}$ in $k[x_1, \ldots, x_n]$ such that*

(i) $>$ is compitable with multiplication.

(ii) $>$ is a well-ordering. That means every non-empty set of monomials has a smallest element under $>$.

There are infinitely many monomials orderings. We introduce three of them.

Definition 2.1.2. *(Lexicographic Order) Let $m_1 = x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_n^{\alpha_n}$ and $m_2 = x_1^{\beta_1} x_2^{\beta_2} \ldots x_n^{\beta_n}$ be two monomials. We say $m_1 >_{lex} m_2$ if the first nonzero $\alpha_i - \beta_i$ is positive for $i = 1, 2, \ldots, n$.*

For example $x_1^3 x_2^2 x_3 >_{lex} x_1^2 x_2^5 x_3^9$.

Definition 2.1.3. *(Degree Reverse Lexicographic Order) Let $m_1 = x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_n^{\alpha_n}$ and $m_2 = x_1^{\beta_1} x_2^{\beta_2} \ldots x_n^{\beta_n}$ be two monomials. We say $m_1 >_{grevlex} m_2$ if $\alpha_1 + \alpha_2 + \ldots + \alpha_n > \beta_1 + \beta_2 + \ldots + \beta_n$ or if $\alpha_1 + \alpha_2 + \ldots + \alpha_n = \beta_1 + \beta_2 + \ldots + \beta_n$ then the last non-zero $\alpha_i - \beta_i$ is negative for $i = 1, 2, \ldots, n$.*

For example, $x_1 x_2^4 x_3 >_{grevlex} x_1^2 x_2^2 x_3^2$, since $\{1+4+1 = 2+2+2\}$, but the difference of exponents of x_3 is $\{1 - 2 = -1\}$, negative.

Definition 2.1.4. *(Weighted Lexicographic Order) Given $w = (w_1, \ldots, w_n)$, each w_i is non-negative integers. Let $m_1 = x_1^{\alpha_1} x_2^{\alpha_2} \ldots x_n^{\alpha_n}$ and $m_2 = x_1^{\beta_1} x_2^{\beta_2} \ldots x_n^{\beta_n}$ be two monomials. We say $m_1 >_{wlex} m_2$ if $w_1\alpha_1 + w_2\alpha_2 + \ldots + w_n\alpha_n > w_1\beta_1 + w_2\beta_2 + \ldots + w_n\beta_n$ and ties broken by lexicographic order.*

For example, let $w = (1, 2, 4)$. Then $x_1x_3^2 >_{wlex} x_1^2x_2x_3$, since $\{1 \times 1 + 4 \times 2 > 1 \times 2 + 2 \times 1 + 4 \times 1\}$.

Once a monomial ordering is chosen, then precisely the terms of any polynomial can be ordered with respect to this ordering. For example, let

$$f = 7x_1x_2^2x_3 - 3x_3^2 + 4x_1^3 + 2x_1^2x_2^2.$$

We would like to reorder terms of f in decreasing order. Therefore, with respect to lexicographic order,

$$f = 4x_1^3 + 2x_1^2x_2^2 + 7x_1x_2^2x_3 - 3x_3^2,$$

with respect to degree reverse lexicographic order,

$$f = 7x_1x_2^2x_3 + 2x_1^2x_2^2 + 4x_1^3 - 3x_3^2$$

and with respect to weighted lexicographic order is

$$f = 7x_1x_2^2x_3 - 3x_3^2 + 2x_1^2x_2^2 + 4x_1^3$$

where $w = (1, 2, 4)$.

Definition 2.1.5. *Let $f \in k[x_1, \ldots, x_n]$ be a non-zero polynomial and $<$ be a monomial ordering. Then f can be uniquely written as*

$$f = c_1m_1 + c_2m_2 + \ldots + c_sm_s$$

where $m_1 > m_2 > \ldots > m_s$ and $c_i \in k$. The leading monomial of f is $LM(f) = m_1$, the leading coefficient of f is $LC(f) = c_1$ and the leading term of f is $LT(f) = c_1m_1$.

For example, let $f = 4x_1^3 + 2x_1^2x_2^2 + 7x_1x_2^2x_3 - 3x_3^2$ as before. Then $LM(f) = x_1^3$, $LC(f) = 4$ and $LT(f) = 4x_1^3$ with respect to lexicographic order.

Division Algorithm

In this section we introduce a division algorithm in $k[x_1, \ldots, x_n]$ which is a generalization of the division algorithm in one variable polynomial ring $k[x]$. The main difference is that we can divide a polynomial $f \in k[x_1, \ldots, x_n]$ by more than one polynomial simultaneously. Let us illustrate this with an example.

Example 2.2.1. *Let* $f = x_1^2x_2 + x_1x_2^2 + x_2^2$, $f_1 = x_1x_2 - 1$ *and* $f_2 = x_2^2 - 1$. *We will use lexicographic order. Hence* $LT(f) = x_1^2x_2$, $LT(f_1) = x_1x_2$ *and* $LT(f_2) = x_2^2$. *Since* $LT(f)/LT(f_1) = x_1$, *we redefine* f *as*

$$f := f - x_1f_1 = x_1x_2^2 + x_1 + x_2^2.$$

Since again $LT(f)/LT(f_1) = x_2$, *we have*

$$f := f - x_2f_1 = x_1 + x_2^2 + x_2.$$

Now, $LT(f) = x_1$ *is divisible neither* $LT(f_1)$ *nor* $LT(f_2)$. *Therefore, we move* x_1 *to the remainder and continue by*

$$f := f - x_1 = x_2^2 + x_2.$$

Then $LT(f)/LT(f_2) = 1$ *and*

$$f := f - f_2 = x_2 + 1.$$

Since $LT(f_1)$ *and* $LT(f_2)$ *do not divide* x_2 *and* 1, *the remainder is* $x_1 + x_2 + 1$. *Hence*

$$f = (x_1 + x_2)f_1 + f_2 + x_1 + x_2 + 1.$$

This is a good illustration of the multivariable division algorithm. We are now ready to give the statement of the division algorithm.

Theorem 2.2.2. *(Division Algorithm in* $k[x_1, \ldots, x_n]$*) For a given monomial ordering* $>$, $F = (f_1, \ldots, f_s) \in k[x_1, \ldots, x_n]^s$ *and* $f \in k[x_1, \ldots, x_n]$, *there exist polynomials* $a_1, a_2, \ldots, a_s, r \in k[x_1, \ldots, x_n]$ *such that*

$$f = a_1f_1 + \ldots + a_sf_s + r$$

where no term of r *is divisible by any* $LT(f_1), \ldots, LT(f_s)$. *Furthermore, if* $a_if_i \neq 0$, $LM(f) \geq LM(a_if_i)$ *for* $i = 1, \ldots, n$.

There are however some shortcomings of this algorithm. The most important one is the remainder is not unique. For example, let $I = (f_1, f_2) = (x_1^2, x_1x_2 - x_2^2)$ and let $f = x_1^2x_2$. If we divide f first by f_1, $f = x_2f_1$. So remainder is 0. On the other hand, by dividing first by f_2, we have

$$f = (x_1 + x_2)f_2 + x_2^3.$$

The Gröbner basis which we define in the next section solves this problem.

Gröbner Bases

Definition 2.3.1. *Let $I \subseteq k[x_1, \ldots, x_n]$ be an ideal. Fix a monomial ordering in $\{g_1, g_2, \ldots, g_n\}$ is called Gröbner basis for I if*

$$< LM(g_1), \ldots, LM(g_s) >=< LM(f) : f \in I > .$$

A generating set of an ideal is not always a Gröbner basis.

For example, let $I =< g_1, g_2 >=< x_1^2 - x_2, x_1^3 - x_3 >$. Clearly,

$$f = x_1 g_1 - g_2 = -x_1 x_2 + x_3 \in I$$

but $LM(f) = x_1 x_2 \notin < LM(g_1) = x_1^2, LM(g_2) = x_1^3 >$ with respect to lexicographic order.

Now, we give two important properties of Gröbner bases.

Proposition 2.3.2. *If $\{g_1, \ldots, g_s\}$ is a Gröbner basis for an ideal $I \subseteq k[x_1, \ldots, x_n]$, then $I =< g_1, \ldots, g_n >$.*

Theorem 2.3.3. *Let $G = \{g_1, \ldots, g_s\}$ be a Gröbner basis for an ideal $I \subseteq k[x_1, \ldots, x_n]$. If $f \in k[x_1, \ldots, x_n]$, then the remainder of f upon division by G is unique.*

The next step is to give a criterian for a Gröbner basis. First, we have to define a special polynomial which is called as S-polynomial.

Definition 2.3.4. *Let f_1, f_2 be two polynomials. The S-polynomial of f_1 and f_2 is defined by*

$$S(f_1, f_2) = \frac{LCM(LM(f_1), LM(f_2))}{LT(f_1)} f_1 - \frac{LCM(LM(f_1), LM(f_2))}{LT(f_2)} f_2. \qquad (2.1)$$

Example 2.3.5. *Let $f_1 = x_1^2 x_2^3 - x_1 x_2^4$ and $f_2 = x_1^4 x_2 + x_2^2$ with lexicographic order. Hence $LM(f_1) = x_1^2 x_2^3$, $LM(f_2) = x_1^4 x_2$ and $LCM(LM(f_1), LM(f_2)) = x_1^4 x_2^3$.*

Then

$$
\begin{aligned}
S(f_1, f_2) &= \frac{x_1^4 x_2^3}{x_1^2 x_2^3} f_1 - \frac{x_1^4 x_2^3}{x_1^4 x_2} f_2 \\
&= x_1^2 f_1 - x_2^2 f_2 \\
&= -x_1^3 x_2^4 + x_2^4
\end{aligned} \tag{2.2}
$$

Buchberger introduced the concept of Gröbner bases in his dissertation. He also gave a criterion for a generating set being a Gröbner basis for an ideal.

Theorem 2.3.6. *(Buchberger's Criterion) Suppose $G = \{g_1, \ldots, g_s\}$ is a generating set for an ideal I. Then G is a Gröbner basis of I if and only if for all $i \neq j$, the remainder of $S(g_i, g_j)$ upon division by G is zero.*

Buchberger Algorithm

Buchberger's criterion suggests the following algorithm for finding a Gröbner basis of an ideal from a given generating set of this ideal.

Theorem 2.4.1. *(Buchberger's Algorithm) Let $G = \{f_1, \ldots, f_s\} \subseteq k[x_1, \ldots, x_n]$ and I be the ideal generated by G. Compute an S-polynomials $S(f_i, f_j)$ for $i \neq j$, and divide it by G. If the remainder is not zero, then enlarge G by the remainder. Repeat this process until all of the remainders of the S-polynomials are zero. The result is a Gröbner basis for I.*

Using the ascending chain condition of ideals in $k[x_1, \ldots, x_n]$, one can show that this algorithm terminates after finitely many steps.

Definition 2.4.2. *A Gröbner basis G of an ideal is called minimal if for each $f \in G$, $LM(f) \neq LM(g)$ for all $g \in G \setminus \{f\}$. A minimal Gröbner basis G is called reduced if for each $f \in G$, no term of f is divisible by $LM(g)$ for all $g \in G \setminus \{f\}$.*

A minimal Gröbner basis can be obtained from a Gröbner basis by simply keeping only one of the polynomials with same leading monomial and droping other from the bases. A reduced Gröbner basis can be obtained from a minimal Gröbner basis by dividing each polynomial in the basis by other polynomials of the basis and replacing original polynomial by the remainder.

Example 2.4.3. *Let* $I = \langle f_1 = x_4x_5 - 1, f_2 = x_2 - x_3x_5^3x_6, f_3 = x_1 - x_3^2x_5x_6\rangle$ *and* $w = (2, 3, 0, 0, 5, 5)$. *We will find the reduced Gröbner basis with respect to wlex order. Then,* $LT(f_1) = x_4x_5$, $LT(f_2) = -x_3x_5^3x_6$, $LT(f_3) = -x_3^2x_5x_6$.

Now, we can compute S*- polynomials;*

$$S(f_1, f_2) = x_3x_5^2x_6f_1 - (-x_4)f_2 = x_2x_4 - x_3x_5^2x_6 = f_4.$$

$$S(f_1, f_3) = x_3^2x_6f_1 - x_4f_3 = x_1x_4 - x_3^2x_6 = f_5.$$

$$S(f_2, f_3) = x_3f_2 - x_5^2f_3 = x_2x_3 - x_1x_5^2 = f_6.$$

Similarly, $f_6 = x_2x_3 - x_1x_5^2$.

$$S(f_1, f_4) = x_3x_5x_6f_1 - x_4f_4 = x_3x_5x_6 - x_2x_4^2 = f_7.$$

$$S(f_2, f_4) = f_2 - x_5f_4 = x_2 - x_2x_4x_5 = -x_2f_1 + 0.$$

So, there no new polynomial to add generating set of the ideal.

$$S(f_3, f_4) = x_5f_3 - x_3f_4 = x_1x_5 - x_2x_3x_4 = f_8.$$

$$S(f_1, f_5) = x_3^2x_6f_1 - x_4x_5f_4 = x_3^2x_6 - x_1x_4^2x_5 = -f_5 + 0.$$

Therefore, there is no polynomial to add generating set of the ideal.

$$S(f_1, f_7) = x_3x_6f_1 - x_4x_5f_7 = -x_3x_6 + x_2x_4^3 = f_9$$

$$S(f_3, f_7) = f_3 - x_3f_7 = -x_1 + x_2x_3x_4^2 = f_{10}.$$

$$S(f_1, f_8) = f_1 - x_3f_8 = -x_1 + x_2x_3x_4^2 = -x_2x_4^2f_{10} + x_2^2x_5^5 - x_1x_5.$$

So, $f_{11} = x_2^2x_5^5 - x_1x_5$. *One can show that the remainder of other* S*-polynomials are zero. Hence* $G = \{f_1, f_2, f_3, f_4, f_5, f_6, f_7, f_8, f_9, f_{10}, f_{11}\}$,

$$G = \{x_4x_5 - 1, x_2 - x_3x_5^3x_6, x_1 - x_3^2x_5x_6, x_2x_4 - x_3x_5^2x_6, x_1x_4 - x_3^2x_6,$$

$$x_2x_3 - x_1x_5^2, x_3x_5x_6 - x_2x_4^2, x_1x_5 - x_2x_3x_4, x_2x_4^3 - x_3x_6, x_1 - x_2x_3x_4^2, x_2^2x_4^5 - x_1x_6\}.$$

Since leading terms of $f_2, f_3, f_4, f_5, f_6, f_7$ and f_8 are divisible by the leading terms of others, they can be droped from the basis. Thus, the reduced Gröbner basis is

$$G = \{x_2x_3x_4^2 - x_1, x_4x_5 - 1, x_1x_5 - x_2x_3x_4, x_3x_6 - x_2x_4^3, x_1x_6 - x_2^2x_4^5\}.$$

There are several applications of Gröbner basis theory. One of them is so called Elimination Theorem.

Definition 2.4.4. *Let $I = \langle f_1, \ldots, f_s \rangle$ be an ideal in polynomial ring $k[x_1, \ldots, x_n]$. The ideal $I_k = I \cap k[x_{k+1}, \ldots, x_n]$ is called the k-th elimination ideal of I.*

Theorem 2.4.5. *(The Elimination Theorem)*
Let G be a Gröbner basis of an ideal I with respect to lex order where $x_1 > x_2 > \ldots > x_n$. Then

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

is a Gröbner basis of I_k.

3. TORIC IDEALS

Toric ideals are main tool to translate an integer programming problem into a problem about polynomials. This chapter devoted to computations of toric ideals. There are several algorithms in the literature for finding a generating set for a toric ideal. They all involve one or more Gröbner bases computations. Since Buchberger's algorithm is sensitive to the number of variables, the main purpose is to find a method which computes Gröbner bases with fewer variables.

Computation of Toric Ideals

Definition 3.1.1. *Let k be a field. For a given $m \times n$ matrix $A = (a_{ij})$ of integers, the toric ideal I_A associated to A is the kernel of the k-algebra homomorphism*

$$\phi : k[x_1, x_2, \ldots, x_n] \longrightarrow k[t_1, t_2, \ldots, t_m, t_1^{-1}, \ldots, t_m^{-1}]$$

given by

$$\phi(x_i) = t_1^{a_{i1}} \cdots t_m^{a_{in}}$$

for $i = 1, 2, \ldots, m$.

We give a couple of example of toric ideals.

Example 3.1.2. *Let*

$$A_2 = \begin{pmatrix} 2 & 1 & 2 & 1 & 0 & 0 \\ 1 & 2 & 0 & 0 & 2 & 1 \\ 0 & 0 & 1 & 2 & 1 & 2 \end{pmatrix}.$$

We have the homomorphism

$$\phi : k[x_1, x_2, x_3, x_4, x_5, x_6] \longrightarrow k[t_1, t_2, t_3, t_1^{-1}, t_2^{-1}, t_3^{-1}]$$

defined by

$$x_1 \mapsto t_1^2 t_2, x_2 \mapsto t_1 t_2^2, x_3 \mapsto t_1^2 t_3, x_4 \mapsto t_1 t_3^2, x_5 \mapsto t_2^2 t_3, x_6 \mapsto t_2 t_3^2.$$

The followings are some elements of I_{A_2}:

$$\{x_1x_6 - x_2x_4, x_2^2x_3 - x_1^2x_5, x_4x_5 - x_2x_3x_6\}.$$

Example 3.1.3. *Let*

$$A_3 = \begin{pmatrix} 3 & -2 & 1 & 0 \\ 4 & 1 & -1 & -1 \end{pmatrix}.$$

Hence

$$\phi : k[x_1, x_2, x_3, x_4] \longrightarrow k[t_1, t_2, t_1^{-1}, t_2^{-1}]$$

defined by

$$x_1 \mapsto t_1^3t_2^4, x_2 \mapsto t_1^{-1}t_2, x_3 \mapsto t_1t_2^{-1}, x_4 \mapsto t_2^{-1}.$$

For example, $x_2x_3^2 - x_4 \in I_{A_3}$.

We can not describe all elements of a toric ideal in this way. We need to find a generating set. The first idea of finding generating sets for toric ideals by using Gröbner bases due to (Conti and Traverso, 1991). We give a different treatment of this idea. First, we consider the case when all entries of the matrix A is non-negative.

Theorem 3.1.4. *Let $>$ be lexicographic monomial order in $k[t_1, \ldots, t_m, x_1, \ldots, x_n]$ in which all t_i's precedes all the x_j's. If G is the Gröbner basis for the ideal*

$$J = \langle x_1 - t^{a_1}, x_2 - t^{a_2}, \ldots, x_n - t^{a_n} \rangle$$

with respect to $>$, then the set of polynomials in G involving only x_i's is a Gröbner basis for I_A.

Proof. Definition of toric ideal and Theorem 2.4.2 of Adams and Loustaunau (1994) implies that $I_A = J \cap k[x_1, \ldots, x_n]$. The conclusion easily follows from Elimination Theorem. □

Example 3.1.5. *Consider the above matrix A_2. Define the ideal*

$$I = \langle x_1 - t_1^2t_2, x_2 - t_1t_2^2, x_3 - t_1^2t_3, x_4 - t_1t_3^2, x_5 - t_2^2t_3, x_6 - t_2t_3^2 \rangle.$$

Gröbner basis of I relative to lexicographic order with $t_1 > t_2 > t_3 > x_1 > x_2 > \ldots > x_6$ is

$\{x_3x_6^2 - x_4x_5^2, x_2 - x_3, x_1x_6 - x_3x_4, x_1x_5^2 - x_3^2x_6, t_3^3x_5 - x_6^2, t_3^3x_3 - x_4x_5, t_3^6x_1 - x_4^2x_6, t_2x_6 - t_3x_5, t_2x_4x_5 - t_3x_3x_6, t_2x_4^2 - t_3^4x_1, t_2x_3x_4 - t_3x_1x_5, t_2x_1x_5 - t_3x_3^2, t_2t_3^2 - x_6, t_2^2x_4 - t_3^2x_3, t_2^2t_3 - x_5, t_2^3x_1 - x_3^2, t_1x_6 - t_2x_4, t_1x_5 - t_3x_3, t_1x_3 - t_2x_1, t_1t_3^2 - x_4, t_1t_2x_4 - t_3^2x_1, t_1t_2^2 - x_3, t_1^2t_2 - x_1\}$.

Hence

$$I_{A_2} = \langle x_3x_6^2 - x_4x_5^2, x_2 - x_3, x_1x_6 - x_3x_4, x_1x_5^2 - x_3^2x_6\rangle.$$

If some of the entries of the matrix A are negative, then t^{a_i} cannot viewed as a monomial in $k[t_1, t_2, \ldots, t_m]$. In this case we introduce a new variable $w = t_1^{-1} \ldots t_m^{-1}$. Hence each term

$$t^{a_i} = t_1^{a_{i1}} t_2^{a_{i2}} \ldots t_m^{a_{im}}$$

can be rewritten in the form

$$w^{e_j} t_1^{a_{i1}+e_j} t_2^{a_{i2}+ej} \ldots t_m^{a_{im}+e_j}$$

where $e_j = max\{|a_{ij}|, a_{ij} < 0\}$.

Now, we can give a new version of Theorem 3.1.4.

Theorem 3.1.6. *Let $>$ be the lexicographic order in $k[t_1, \ldots, t_d, w, x_1, \ldots, x_n]$ in which all $t_i > w > x_j$ for $i = 0, 1, \ldots m$ and $j = 1, \ldots, n$. If G is the Gröbner basis for the ideal*

$$J = \langle wt_1t_2 \ldots t_d - 1, x_1 - w^{e_j}t^{a_1+e_j}, x_2 - w^{e_j}t^{a_2+e_j}, \ldots, x_n - w^{e_j}t^{a_n+e_j}\rangle$$

with respect to $>$, then the polynomials in G involving only x_j's is a Gröbner basis for I_A.

Example 3.1.7. *Consider the A_3 given above. Construct the ideal I,*

$$I = \langle wt_1t_2 - 1, x_1 - t_1^3t_2^4, x_2 - w^2t_2^3, x_3 - wt_1^2, x_4 - wt_1\rangle.$$

A Gröbner basis of I relative to lexicographic order with $t_1 > t_2 > w > x_1 > x_2 > x_3 > x_4$ is

$G = \{x_2x_3^2 - x_4, x_1x_4^7 - x_3^3, x_1x_2x_4^6 - x_3, x_1x_2^2x_3x_4^5 - 1, w - x_2x_3x_4, t_2 - x_1x_2^2x_3x_4^4, t_1 - x_1x_2x_4^5\}$.

Hence

$$I_A = \langle x_2x_3^2 - x_4, x_1x_4^7 - x_3^3, x_1x_2x_4^6 - x_3, x_1x_2^2x_3x_4^5 - 1\rangle.$$

Unfortunately, this method cannot handle the problem if the number of variables becomes larger. Sturmfels (1995) gave an example toric ideal of (4×8) matrix whose Gröbner basis contains more than 200 polynomials. A more effective algorithm which computes several Gröbner bases in n variables $x_1, \ldots, x_n$ is due to (Sturmfels 1995). Before explanation of this method we need more technical results about toric ideals.

The lattice kernel of the toric ideal I_A is

$$\ker(A) = \{v \in \mathbb{Z}^n | Av = 0\}.$$

For $v \in \ker(A)$, let $v^+ = (u_1, \ldots, u_n)$ where $u_i = v_i$ if $v_i \geq 0$ and $u_i = 0$ if $v_i < 0$, and $u^- = (w_1, \ldots, w_n)$ where $w_i = -v_i$ if $v_i < 0$ and $w_i = 0$ if $v_i \geq 0$. Then we can give an infinite set of generators for I_A.

$$I_A =< \{x^{u^+} - x^{u^-} | u \in \ker(A)\} >$$

where $x = (x_1, \ldots, x_n)$. The following lemma gives a finite generating set for a toric ideal.

Lemma 3.1.8. *(Sturmfels, 1995, Lemma 12.2)*

For any subset C of the lattice ker(A) define the ideal

$$J_C = \langle x^{v^+} - x^{v^-} | v \in C \rangle.$$

The subset C spans the ker(A) if and only if

$$J_C : (x_1 \cdots x_n)^\infty = I_A.$$

Hence the computation of a toric ideal has two steps. The first is the computation of lattice kernel basis C and then finding $J_C : (x_1 \cdots x_n)^\infty$ which is called saturation. The classical techniques of the linear algebra is not enough to solve the first step of the problem. Since a basis for nullspace of a matrix with integer entries may contain non-integer vectors. This step has a solution by computing the Hermite normal form of A (see Schrijver 1986, section 5.3).

Hermite Normal Forms

Definition 3.2.1. *Let $B = (b_1, b_2, \ldots, b_k)$ be a set linearly independent vectors in $\mathbb{R}^n$. The set*

$$L(B) = \sum_{i=1}^{k} \lambda_i b_i : \lambda_1, \lambda_2, \ldots, \lambda_k \in \mathbb{Z}$$

is called a lattice generated by B and we say the lattice $L(B)$ has dimension k. If $k = n$, the lattice $L(B)$ is called full-dimensional.

Recall that a square matrix U with integer entries is called unimodular if

$$det(U) = \pm 1.$$

Lemma 3.2.2. *Let A and B be two $n \times k$ matrices. If $B = AU$ for some unimodular matrix U, then $L(A) = L(B)$ where $L(A)$ and $L(B)$ are lattices generated by columns of A and B respectively.*

Proof. Let U be a unimodular matrix such that $B = AU$. Then for every $y = Bx$ with $x \in R^n$, $y = A(Ux)$. Furthermore, x is an integer if and only if Ux is an integer. This implies that $L(A) = L(B)$. □

Definition 3.2.3. *(Elementary Column Operations) Let B be a matrix. We define the elementary column operations on B :*

(i) Swap two columns in B ($c_i \leftrightarrow c_j$);

(ii) Multiply a column by (-1) ($c_i \rightarrow -c_i$) and

(iii) Add an integer multiple of a column to another column ($c_i \rightarrow c_i + mc_j$).

Notice that we can specify an appropriate unimodular matrices for each of elementary column operations. Hence if the matrix A is obtained from B by a sequence of the above elementary column operations, then there exists a unimodular matrix U such that $A = BU$.

Definition 3.2.4. *(Extended Euclidean Algorithm) Suppose a_0 and a_1 are positive integers. The Euclidean algorithm finds the greatest common divisor of a_0 and a_1, written as $gcd(a_0, a_1)$. This algorithm mainly computes the remainder sequence $a_0, a_1, a_2, \ldots, a_k$ where $a_{i-2} = a_{i-1}a_{i-1} + a_i$, $a_i \in \mathbb{N}$, $0 \leq a_i < a_{i-1}$ for $i = 2, \ldots, k$. Furthermore, if a_k divides a_{k-1}, then $a_k = gcd(a_0, a_1)$. The extended Euclidean algorithm keeps track of unimodular matrices*

$$U_j = \prod_{i=1}^{j} \begin{pmatrix} 0 & 1 \\ 1 & -q_i \end{pmatrix} .$$

Then

$$(a_0 \quad a_1)U_k = (a_k \quad 0).$$

Let us explain this with an example.

Example 3.2.5. *We will find a unimodular matrix U such that*

$$(1398 \quad 324)U = (gcd(1398, 394) \quad 0).$$

The Euclidean Algorithm implies

$$1398 = 4 \times 324 + 102$$

$$324 = 3 \times 102 + 18$$

$$102 = 5 \times 18 + 12$$

$$18 = 1 \times 12 + 6$$

$$12 = 2 \times 6 + 0.$$

Hence $gcd(1398, 324) = 6$.

If we write these calculations in terms of matrix multiplications

$$(1398 \quad 324)\begin{pmatrix} 0 & 1 \\ 1 & -4 \end{pmatrix} = (324 \quad 102)$$

$$(324 \quad 102)\begin{pmatrix} 0 & 1 \\ 1 & -3 \end{pmatrix} = (102 \quad 18)$$

$$(102 \quad 18)\begin{pmatrix} 0 & 1 \\ 1 & -5 \end{pmatrix} = (18 \quad 12)$$

$$(18 \quad 12)\begin{pmatrix} 0 & 1 \\ 1 & -1 \end{pmatrix} = (12 \quad 6)$$

$$(12 \quad 6)\begin{pmatrix} 0 & 1 \\ 1 & -2 \end{pmatrix} = (6 \quad 0).$$

Since

$$\begin{pmatrix} 0 & 1 \\ 1 & -4 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -3 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -5 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & -2 \end{pmatrix} = \begin{pmatrix} -19 & 54 \\ 82 & -233 \end{pmatrix},$$

$$(1398 \quad 324)\begin{pmatrix} -19 & 54 \\ 82 & -233 \end{pmatrix} = (6 \quad 0).$$

We may extend to this technique beyond to two positive integers.

Definition 3.2.6. *(Hermite Normal Form)*

A $n \times n$ matrix B is called in Hermite normal form if it has the form $B = [H \mid 0]$ where $H = (h_{ij})$ is a square matrix such that

(i) H is lower triangular;

(ii) H is non-negative and each row has a unique maximum entry which is on the main diagonal.

Notice that these conditions implies that H is an invertible matrix.

Theorem 3.2.7. *Let A be $n \times k$ matrix of rank n with rational entries. Then A can be transformed into the Hermite normal form by a sequence of elementary column operations. In other words, there exists a unimodular matrix U such that $AU = B$ where B is a Hermite normal form.*

Proof. We may assume all entries of A are integers. Otherwise, we can proceed with the matrix δA where δ is the least common multiple of all denominators of A. The following algorithm converts A into a matrix in the Hermite normal form. Let $D_0 = A$. Produce a sequence of matrices $B_1, B_2, \ldots, B_n$ of the form

$$B_k = \begin{pmatrix} H_k & 0 \\ C_k & D_k \end{pmatrix}$$

where H_k is $k \times k$ matrix in the Hermite normal form as follows:

Let $d_1, d_2, \ldots, d_{n-k}$ be entries of the first row of D_k. By multiplying necessary columns by -1, we may assume that all d_i's are non-negative. Then using the technique of extended Euclidean algorithm we can find a unitary matrix U such that when multiplying this row by U produce only one non-zero entry, say d, in the new row vector. After interchanging columns we guarantee that d is in the first column.

Let $c_1, \ldots, c_k$ be the entries of the first row of c_k. Then we add $\lfloor c_i/d \rfloor$ times $(k+1)$-th column to i-th column of B_k. Thus, all entries in the first row of c_k are non-negative and smaller than d. □

Example 3.2.8.

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

Let $D_0 = A$. Extended Euclidean algorithm gives

$$U_1 = \begin{bmatrix} 1 & -2 & 0 & -1 & 0 & -2 & 0 & -2 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Then

$$B_1 = D_0 U_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & -3 & 1 & -2 & 1 & -3 & 2 & -4 \\ 0 & 0 & 1 & 2 & 2 & 0 & 1 & 2 \\ 1 & -1 & 2 & 0 & 1 & -1 & 1 & -2 \end{bmatrix}.$$

Now, we can identify D_1 and C_1 as

$$D_1 = \begin{bmatrix} -3 & 1 & -2 & 1 & -3 & 2 & -4 \\ 0 & 1 & 2 & 2 & 0 & 1 & 2 \\ -1 & 2 & 0 & 1 & -1 & 1 & -2 \end{bmatrix}$$

and

$$C_1 = \begin{bmatrix} 2 \\ 0 \\ 1 \end{bmatrix}.$$

Again using extended Euclidean algorithm, we can find B_2 as follows:

$$U_2 = \begin{bmatrix} -1 & -1 & 0 & 0 & -1 & 0 & -1 \\ 0 & -1 & 0 & -1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

and

$$B_2 = D_1 U_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 6 & 1 & 0 & 3 & 3 \\ 1 & -1 & 2 & -1 & 0 & 1 & 1 \end{bmatrix}.$$

Therefore

$$D_2 = \begin{bmatrix} 1 & 6 & 1 & 0 & 3 & 3 \\ -1 & 2 & -1 & 0 & 1 & 1 \end{bmatrix}.$$

$$C_2 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}.$$

Applying same process one more times

$$U_3 = \begin{bmatrix} 1 & -6 & -1 & 0 & -3 & -3 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

and

$$B_3 = D_2U_3 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 8 & 0 & 0 & 4 & 4 \end{bmatrix}.$$

Therefore

$$D_3 = \begin{bmatrix} 8 & 0 & 0 & 4 & 4 \end{bmatrix}, \quad C_3 = [-1].$$

Similarly

$$U_4 = \begin{bmatrix} 1 & 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

and then

$$B_4 = D_3U_4 = \begin{bmatrix} 4 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Since the entry of C_3 is negative, we must add element of C_3 to first column of B_4, then recalculate it. Hence

$$U_5 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

and

$$B_4 = \begin{bmatrix} -1 & 4 & 0 & 0 & 0 & 0 \end{bmatrix}, U_4 = \begin{bmatrix} 3 & 4 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Combining the results,

$$B = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 3 & 4 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Hence the corresponding unimodular matrix is

$$U = U_1U_2U_3U_4U_5 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 & -1 & -1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 3 & 2 & 4 & 4 & 0 & 0 & 2 & 1 \\ -1 & 0 & -2 & -1 & -2 & 0 & -3 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 2 & 0 \\ -2 & -2 & -2 & -3 & 1 & 0 & 0 & 0 \end{bmatrix}.$$

It is easy to see that $AU = B$ where U is unimodular and the first 4 columns of B is in Hermite normal form.

Notice that multiplying A by the last four column of U gives 4×4 zero matrix. This implies that the last four columns of U are lattice basis for ker(A).

Hence we obtain a method to find lattice basis for ker(A) where A is an $m \times n$ matrix of integers. First find a unimodular matrix U such that

$$AU = B = [H_k|0]$$

where H_k is in Hermite normal form. Then the last $n - k$ columns of U is the lattice basis for ker(A).

Saturation

Let $I \subseteq k[x_1, \ldots, x_n]$ be an ideal and $g \in k[x_1, \ldots, x_n]$ be a polynomial. Then the quotient ideal is defined as follows

$$I : g = \{f \in k[x_1, \ldots, x_n] : gf \in I\}.$$

An algorithm for finding a generating set for quotient ideal is described in (Cox et al, 1996, Section 4.4).

If $I =< f_1, \ldots, f_s >$, then find a Gröbner basis of $< tf_1, \ldots, tf_s, (1-t)g >$ with respect to lexicographic order in which $t > x_i$. Retain all basis elements which do not depend on t. Then divide each of these elements by g to get a generating set for $I : g$.

Definition 3.3.1. *The saturation of I with respect to g is defined as follows*

$$I : g^\infty = \{f \in k[x_1, \ldots, x_n] : g^m f \in I \quad \textit{for some} \quad m > 0\}.$$

Hence an algorithm for finding a generating set for saturation can be same as above algorithm except in last step we must divide each element by highest power of g which divides this element. Recall that we need to compute $I : (x_1 \ldots x_n)^\infty$. Let us illustrate this with an example.

Example 3.3.2. *Let A be the matrix given in example 3.2.8. Hence a lattice basis of ker(A) is the rows of the following matrix*

$$C = \begin{pmatrix} 0 & -1 & 1 & 0 & -2 & 0 & 1 & 1 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 2 & -3 & 0 & 2 & 0 \\ 1 & -1 & 0 & 1 & -1 & 0 & 0 & 0 \end{pmatrix}.$$

Hence $J_C = \langle x_3x_7x_8 - x_2x_5^2, x_6 - x_2, x_4^2x_7^2 - x_2x_5^3, x_1x_4 - x_2x_5\rangle$. *Define the ideal,let*

$$J = \langle t(x_3x_7x_8 - x_2x_5^2), t(x_6 - x_2), t(x_4^2x_7^2 - x_2x_5^3), t(x_1x_4 - x_2x_5), (1-t)(x_1x_2x_3x_4x_5x_6x_7x_8)\rangle.$$

If we find Gröbner Basis of J_C *using degree lexicographic order with* $t > x_1 > x_2 > x_3 > x_4 > x_5 > x_6 > x_7 > x_8$, *the result is*

$$\{x_1x_2x_3x_4^3x_5x_6x_7^3x_8 - x_1x_2x_3x_4x_5^4x_6^2x_7x_8, x_1x_2x_3^2x_4x_5x_6x_7^2x_8^2 - x_1x_2x_3x_4x_5^3x_6^2x_7x_8,$$
$$x_1x_2x_3^2x_4x_5^2x_6x_7x_8^2 - x_1x_2x_3x_4^3x_5x_6x_7^2x_8, x_1x_2x_3^3x_4x_5x_6x_7x_8^3 - x_1x_2x_3x_4^3x_5^2x_6^2x_7x_8,$$
$$x_1x_2^2x_3x_4x_5x_6x_7x_8 - x_1x_2x_3x_4x_5x_6^2x_7x_8, x_1^2x_2x_3x_4x_5^3x_6x_7x_8 - x_1x_2x_3x_4^2x_5x_6x_7^3x_8,$$
$$x_1^2x_2x_3x_4^2x_5x_6x_7x_8 - x_1x_2x_3x_4x_5^2x_6^2x_7x_8, x_1^2x_2x_3^2x_4x_5x_6x_7x_8^2 - x_1x_2x_3x_4^2x_5x_6^2x_7^2x_8,$$
$$x_1^3x_2x_3x_4x_5^2x_6x_7x_8 - x_1x_2x_3x_4x_5x_6^2x_7^3x_8, tx_5^2x_6^6x_7^4 - x_1^5x_2x_3x_4x_5x_6x_7x_8, tx_5^3x_6^5x_7^2 -$$
$$x_1^3x_2x_3x_4x_5x_6x_7x_8, tx_5^4x_6^4 - x_1x_2x_3x_4x_5x_6x_7x_8, tx_4x_5x_6^5x_7^4 - x_1^4x_2x_3x_4x_5x_6x_7x_8,$$
$$tx_4x_5^2x_6^4x_7^2 - x_1^2x_2x_3x_4x_5x_6x_7x_8, tx_4^2x_7^2 - tx_5^3x_6, tx_4^2x_5^3x_6^4x_7 - x_1x_2x_3^2x_4x_5x_6x_7x_8^2,$$
$$tx_3x_7x_8 - tx_5^2x_6, tx_3x_5^3x_6x_8 - tx_4^2x_5^2x_6x_7, tx_2 - tx_6, tx_1x_5^3x_6 - tx_4x_5x_6x_7^2, tx_1x_4 - tx_5x_6\}.$$

Then, we retain the polynomials which includes only x_i *terms. After that we divide each polynomial with highest possible power of* $x_1x_2x_3x_4x_5x_6x_7x_8$. *Finally, we obtain*

$$I_A = J_C : (x_1 \cdots x_8)^\infty = \langle x_4^2x_7^2 - x_5^3x_6, x_3x_7x_8 - x_5^2x_6, x_3x_5x_8 - x_4^2x_7, x_3^2x_8^2 - x_4^2x_5x_6, x_2 -$$
$$x_6, x_1x_5^2 - x_4x_7^2, x_1x_4 - x_5x_6, x_1x_3x_8 - x_4x_6x_7, x_1^2x_5 - x_6x_7^2\rangle$$

by Lemma 3.1.8.

Notice that we compute a Gröbner basis in $n + 1$ variables. If I is a homogeneous ideal, then Sturmfels(1995) suggested an algorithm which involves computing several Gröbner bases in n variables. It depends on the following observation.

$$I : (x_1 \dots x_n)^\infty = ((\dots (I : x_1^\infty) : x_2^\infty) \dots) : x_n^\infty.$$

Sturmfels(1995) also gave the following lemma.

Lemma 3.3.3. *(Sturmfels, 1995, Lemma 12.1) Fix the degree reverse lexicographic monomial order with* $x_j > x_i$ *for* $j \neq i$ *and let* G *be the Gröbner basis of a homogeneous ideal* I. *Then a Gröbner basis of* $I : x_i^\infty$ *is obtained by dividing each element* $f \in G$ *by the highest power of* x_i *that divides* f.

Hence, this lemma and the above observation gave a method for computing the saturation $I : (x_1 \dots x_n)^\infty$ when I is a homogeneous ideal. Let us illustrate this method by

finding a generating set for the toric ideal of the matrix given in Example 3.2.8.

Example 3.3.4. *We form the ideal* $J_C = \langle x_3x_7x_8 - x_2x_5^2, x_6 - x_2, x_4^2x_7^2 - x_2x_5^3, x_1x_4 - x_2x_5\rangle$ *in Example 3.3.2. The Gröbner basis of* J_C *relative to degree reverse lexicographic order with* $x_8 > x_7 > x_6 > x_5 > x_4 > x_3 > x_2 > x_1$ *is*

$G_1 = \{x_6 - x_2, x_2x_5 - x_1x_4, x_3x_7x_8 - x_1x_4x_5, x_4^2x_7^2 - x_1x_4x_5^2, x_1x_4^3x_5x_7 - x_1x_3x_4x_5^2x_8, x_1^2x_4^4x_7 - x_1^2x_3x_4^2x_5x_8, x_1x_3^2x_4x_5^2x_8^2 - x_1^2x_4^4x_5^2, x_1^2x_3^2x_4^2x_5x_8^2 - x_1^3x_4^5x_5, x_1^3x_3^2x_4^3x_8^2 - x_1^4x_4^6\}$.

If we divide each polynomial in G_1 *by highest power of* x_1 *that divides the polynomial, then*

$J_1 = J_C : x_1^\infty = \langle x_6 - x_2, x_2x_5 - x_1x_4, x_3x_7x_8 - x_1x_4x_5, x_4^2x_7^2 - x_1x_4x_5^2, x_4^3x_5x_7 - x_3x_4x_5^2x_8, x_4^4x_7 - x_3x_4^2x_5x_8, x_3^2x_4x_5^2x_8^2 - x_1x_4^4x_5^2, x_3^2x_4^2x_5x_8^2 - x_1x_4^5x_5, x_3^2x_4^3x_8^2 - x_1x_4^6\rangle$.

The Gröbner basis of J_1 *relative to degree reverse lexicographic order with* $x_8 > x_7 > x_6 > x_5 > x_4 > x_3 > x_1 > x_2$ *is*

$G_2 = \{x_6 - x_2, x_1x_4 - x_2x_5, x_3x_7x_8 - x_2x_5^2, x_4^2x_7^2 - x_2x_5^3, x_2x_4x_5x_7^2 - x_1x_2x_5^3, x_4^4x_7 - x_3x_4^2x_5x_8, x_4^3x_5x_7 - x_3x_4x_5^2x_8, x_1^2x_2x_5^3 - x_2^2x_5^2x_7^2, x_2x_4^2x_5^2x_7 - x_2x_3x_5^3x_8, x_1x_2x_3x_5^3x_8 - x_2^2x_4x_5^3x_7, x_3^2x_4^3x_8^2, -x_2x_4^5x_5, x_3^2x_4^2x_5x_8^2 - x_2x_4^4x_5^2, x_3^2x_4x_5^2x_8^2 - x_2x_4^3x_5^3, x_2x_3^2x_5^3x_8^2 - x_2^2x_4^2x_5^4\}$.

If we divide each polynomial in G_2 *by highest power of* x_2 *that divides each polynomial, then*

$J_2 = J_1 : x_2^\infty = J_C : (x_1x_2)^\infty = \langle x_6 - x_2, x_1x_4 - x_2x_5, x_3x_7x_8 - x_2x_5^2, x_4^2x_7^2 - x_2x_5^3, x_4x_5x_7^2 - x_1x_5^3, x_4^4x_7 - x_3x_4^2x_5x_8, x_4^3x_5x_7 - x_3x_4x_5^2x_8, x_1^2x_5^3 - x_2x_5^2x_7^2, x_4^2x_5^2x_7 - x_3x_5^3x_8, x_1x_3x_5^3x_8 - x_2x_4x_5^3x_7, x_3^2x_4^3x_8^2 - x_2x_4^5x_5, x_3^2x_4^2x_5x_8^2 - x_2x_4^4x_5^2, x_3^2x_4x_5^2x_8^2 - x_2x_4^3x_5^3, x_3^2x_5^3x_8^2 - x_2x_4^2x_5^4\rangle$.

The Gröbner basis of J_2 *using degree reverse lexicographic order with* $x_8 > x_7 > x_6 > x_5 > x_4 > x_1 > x_2 > x_3$ *is*

$G_3 = \{x_6 - x_2, x_1x_4 - x_2x_5, x_2x_5^2 - x_3x_7x_8, x_4^2x_7^2 - x_3x_5x_7x_8, x_4x_5x_7^2 - x_1x_5^3, x_1^2x_5^3 - x_3x_7^3x_8, x_4^4x_7 - x_3x_4^2x_5x_8, x_4^3x_5x_7 - x_3x_4x_5^2x_8, x_4^2x_5^2x_7 - x_3x_5^3x_8, x_1^2x_3x_5x_7x_8 - x_2x_3x_7^3x_8, x_3x_4x_7^3x_8 - x_1x_3x_5^2x_7x_8, x_2x_4^5x_5 - x_3^2x_4^3x_8^2, x_2^2x_3x_5x_7^3x_8 - x_1^2x_3^2x_7^2x_8^2, x_2^3x_3x_7^5x_8 - x_1^4x_3^2x_7^2x_8^2\}$.

If we divide each polynomial in G_3 by highest power of x_3 that divides the polynomial, then

$J_3 = J_2 : x_3^\infty = J_C : (x_1x_2x_3)^\infty = \langle x_6 - x_2, x_1x_4 - x_2x_5, x_2x_5^2 - x_3x_7x_8, x_4^2x_7^2 - x_3x_5x_7x_8, x_4x_5x_7^2 - x_1x_5^3, x_1^2x_5^3 - x_3x_7^3x_8, x_4^4x_7 - x_3x_4^2x_5x_8, x_4^3x_5x_7 - x_3x_4x_5^2x_8, x_4^2x_5^2x_7 - x_3x_5^3x_8, x_1^2x_5x_7x_8 - x_2x_7^3x_8, x_4x_7^3x_8 - x_1x_5^2x_7x_8, x_2x_4^5x_5 - x_3^2x_4^3x_8^2, x_2^2x_5x_7^3x_8 - x_1^2x_3x_7^2x_8^2, x_2^3x_7^5x_8 - x_1^4x_3x_7^2x_8^2\rangle$.

The Gröbner basis of J_3 relative to degree reverse lexicographic order with $x_8 > x_7 > x_6 > x_5 > x_1 > x_2 > x_3 > x_4$ is

$G_4 = \{x_6 - x_2, x_2x_5 - x_1x_4, x_3x_7x_8 - x_1x_4x_5, x_1x_4x_5^2 - x_4^2x_7^2, x_1x_5^3 - x_4x_5x_7^2, x_3x_4^2x_5x_8 - x_4^4x_7, x_1^2x_4^2x_5 - x_2x_4^2x_7^2, x_3x_4x_5^2x_8 - x_4^3x_5x_7, x_3x_5^3x_8 - x_4^2x_5^2x_7, x_1^2x_5x_7x_8 - x_2x_7^3x_8, x_1x_5^2x_7x_8 - x_4x_7^3x_8, x_1x_3x_4^3x_8 - x_2x_4^4x_7, x_2^2x_4^2x_7^2 - x_1^3x_4^3, x_2^2x_7^3x_8 - x_1^3x_4x_7x_8, x_3^2x_4^3x_8^2 - x_1x_4^6\}$.

If we divide each polynomial in G_4 by highest power of x_4 that divides each polynomial, then

$J_4 = J_3 : x_4^\infty = J_C : (x_1x_2x_3x_4)^\infty = \langle x_6 - x_2, x_2x_5 - x_1x_4, x_3x_7x_8 - x_1x_4x_5, x_1x_5^2 - x_4x_7^2, x_1x_5^3 - x_4x_5x_7^2, x_3x_5x_8 - x_4^2x_7, x_1^2x_5 - x_2x_7^2, x_3x_5^2x_8 - x_4^2x_5x_7, x_3x_5^3x_8 - x_4^2x_5^2x_7, x_1^2x_5x_7x_8 - x_2x_7^3x_8, x_1x_5^2x_7x_8 - x_4x_7^3x_8, x_1x_3x_8 - x_2x_4x_7, x_2^2x_7^2 - x_1^3x_4, x_2^2x_7^3x_8 - x_1^3x_4x_7x_8, x_3^2x_8^2 - x_1x_4^3\rangle$.

The Gröbner basis of J_4 relative to degree reverse lexicographic order with $x_8 > x_7 > x_6 > x_1 > x_2 > x_3 > x_4 > x_5$ is

$G_5 = \{x_6 - x_2, x_1x_4 - x_2x_5, x_4^2x_7 - x_3x_5x_8, x_4x_7^2 - x_1x_5^2, x_1x_3x_8 - x_2x_4x_7, x_3x_7x_8 - x_2x_5^2, x_2x_7^2 - x_1^2x_5, x_3^2x_8^2 - x_2x_4^2x_5\}$.

Notice that no polynomial in G_5 is divisible by x_5. Hence $J_5 = J_4 : x_5^\infty = J_C : (x_1 \cdots x_5)^\infty = \langle G_5\rangle$. The Gröbner basis of J_5 relative to degree reverse lexicographic order $x_8 > x_7 > x_1 > x_2 > x_3 > x_4 > x_5 > x_6$ is

$G_6 = \{x_2 - x_6, x_1x_4 - x_5x_6, x_6x_7^2 - x_1^2x_5, x_4^2x_7 - x_3x_5x_8, x_4x_7^2 - x_1x_5^2, x_1x_3x_8 - x_4x_6x_7, x_3x_7x_8 - x_5^2x_6, x_3^2x_8^2 - x_4^2x_5x_6\}$.

Since no polynomial in G_6 divisible by x_6, $J_6 = J_5 : x_6^\infty = J_C : (x_1 \cdots x_6)^\infty = \langle G_6 \rangle$. The Gröbner basis of J_6 relative to degree reverse lexicographic order $x_8 > x_1 > x_2 > x_3 > x_4 > x_5 > x_6 > x_7$ is

$G_7 = \{x_2 - x_6, x_1x_4 - x_5x_6, x_5^2x_6 - x_3x_7x_8, x_1x_5^2 - x_4x_7^2, x_3x_5x_8 - x_4^2x_7, x_1^2x_5 - x_6x_7^2, x_1x_3x_8 - x_4x_6x_7, x_3^2x_8^2 - x_4^2x_5x_6\}$.

Again no polynomial in G_7 is divisible by f_7. So $J_7 = J_6 : x_7^\infty = J_C : (x_1 \cdots x_7)^\infty = \langle G_7 \rangle$. Finally, the Gröbner basis of J_7 using degree reverse lexicographic order with $x_1 > x_2 > x_3 > x_4 > x_5 > x_6 > x_7 > x_8$ is

$G_8 = \{x_2 - x_6, x_1x_4 - x_5x_6, x_4x_6x_7 - x_1x_3x_8, x_4^2x_7 - x_3x_5x_8, x_5^2x_6 - x_3x_7x_8, x_1x_5^2 - x_4x_7^2, x_1^2x_5 - x_6x_7^2, x_5x_6^2x_7 - x_1^2x_3x_8, x_4^2x_5x_6 - x_3^2x_8^2, x_6^3x_7^3 - x_1^4x_3x_8\}$.

Since no polynomial in G_8 is divisible by x_8, $J_8 = J_7 : x_7^\infty = J_C : (x_1 \cdots x_8)^\infty = \langle G_8 \rangle$. Hence $I_A = \langle G_8 \rangle$ by Lemma 3.1.8.

Two examples produce two different generating set for the same toric ideal. It is not however difficult to see that they are equivalent. Notice that Sturmfels' method involves n Gröbner basis computations in n variables. The Cox's method computes only one Gröbner basis in $n + 1$ variables. It seems Sturmfels' method does more computations. This is not the case because Buchberger's algorithm for computation of Gröbner bases is variable sensitive. That means that in general computing n Gröbner basis in n variables takes less time than computing a Gröbner basis in $n + 1$ variables. Also notice that if given ideal is not homogeneous, then it can be homogenized with an extra variable. In the next section, we suggest a new method for computing the saturation. Our method will involve n Gröbner basis computations in fewer variables.

A New Method for Saturation Computation

Let A be $m \times n$ matrix. In order to find a generating set for the toric ideal I_A, we need to compute a saturation

$$J : (x_1 \dots x_n)^\infty$$

for an ideal $J \subset k[x_1, \ldots, x_n]$ of the form

$$< x^{\alpha_1} - x^{\beta_1}, \ldots, x^{\alpha_s} - x^{\beta_s} > .$$

This kind of ideals are called difference binomial ideals.

Using properties of difference binomial ideals, we try to give a new method for the saturation computation.

Definition 3.4.1. *Let $\phi_i : k[x_1, \ldots, x_n] \longrightarrow k[x_{i+1}, \ldots, x_n]$ be a map given by*

$$\phi_i(f) = f \mid_{x_1=\ldots=x_i=1} .$$

This map is called i-th projection map.

Lemma 3.4.2. *(Pseudo Division Algorithm)*

Fix a monomial ordering in $k[x_{i+1}, \ldots, x_n]$. Given the difference binomials $f, f_1, \ldots, f_s \in k[x_1, \ldots, x_n]$ such that $\phi_i(f) \neq 0$, there exist polynomials $h_1, \ldots, h_s$, a difference binomial r and a monomial $x_1^{\alpha_1} \ldots x_i^{\alpha_i}$ such that

$$x_1^{\alpha_1} \ldots x_i^{\alpha_i} f = h_1 f_1 + \ldots + h_s f_s + r$$

where

$$\phi_i(f) = \phi_i(h_1)\phi_i(f_1) + \ldots + \phi_i(h_s)\phi_i(f_s) + \phi_i(r)$$

is the standard expression obtained by division algorithm in $k[x_{i+1}, \ldots, x_n]$.

Proof. If $LT(\phi_i(f))$ is not divisible by any $LT(\phi_i(f_j))$ for $j = 1, \ldots, s$, then $r = f$ and $h_1 = h_2 = \ldots = h_s = 0$. Suppose that $LT(\phi_i(f_j))$ divides $LT(\phi_i(f))$ for some j. Therefore there is a term $cx_{i+1}^{\alpha_{i+1}} \ldots x_n^{\alpha_n}$ such that

$$LT(\phi_i(f)) = cx_{i+1}^{\alpha_{i+1}} \ldots x_n^{\alpha_n} LT(\phi_i(f_j)).$$

Furthermore the polynomials f and f_j contains monomials of the form $x_1^{\beta_1} \ldots x_i^{\beta_i} LM(\phi_i(f))$ and $x_1^{\gamma_1} \ldots x_i^{\gamma_i} LM(\phi(f_j))$ respectively. Then the construction

$$f' = x_1^{\gamma_1} \ldots x_i^{\gamma_i} f - cx_1^{\beta_1} \ldots x_i^{\beta_i} x_{i+1}^{\alpha_{i+1}} \ldots x_n^{\alpha_n} f_j$$

is a difference binomial. Now, we can apply same process to f'. Continuing this way until reaching a binomial n such that $h = 0$ or $LT(\phi_i(h))$ is not divisible by any $LT(\phi_i(f_j))$, one

can get desired expression. Termination of this process after finitely many steps guaranteed by the division algorithm. □

tab Let us express the process with an example.

Example 3.4.3. *Let* $f = x_1^2x_2x_3^4x_4 - x_1x_2^3x_3^2x_4^2$, $f_1 = x_1^3x_2x_3x_4^2 - x_1x_2^2x_3^2x_4$, *and* $f_2 = x_1^2x_2x_3x_4 - x_1x_2^2x_4^2$ *be polynomials. Hence* $\phi_2(f) = x_3^4x_4 - x_3^2x_4^2$, $\phi_2(f_1) = x_3x_4^2 - x_3^2x_4$, $\phi_2(f_2) = x_3x_4 - x_4^2$.

Let $>$ *be the lexicographic order in* $k[x_3, x_4]$. $LT(\phi_2(f)) = x_3^4x_4$, $LT(\phi_2(f_1)) = -x_3^2x_4$ *and* $LT(\phi_2(f_2)) = x_3x_4$. *Clearly* $LT(\phi_2(f_1))$ *divides* $LT(\phi_2(f))$. *Then we can calculate pseudo division, since* $LT(\phi_2(f)) = x_3^2LT(\phi_2(f_1))$

$$g_1 = x_1{x_2}^2f + {x_1}^2x_2{x_3}^2f_1 = x_1^5x_2^2x_3^3x_4^2 - x_1^2x_2^4x_3^2x_4^2.$$

Then $\phi_2(g_1) = x_3^3x_4^2 - x_3^2x_4^2$ *and* $LT(\phi_2(g_1)) = x_3^3x_4^2$ *is divisible by* $LT(\phi_2(f_2))$.

We apply pseudo division again, since $LT(\phi_2(g_1)) = x_3^2x_4LT(\phi_2(f_2))$

$$g_2 = {x_1}^2x_2g_1 - {x_1}^5{x_2}^2{x_3}^2x_4f_2 = x_1^6x_2^4x_3^2x_4^3 - x_1^4x_2^5x_3^2x_4^2.$$

Hence $\phi_2(g_2) = x_3^2x_4^3 - x_3^2x_4^2$, $LT(\phi_2(g_2)) = x_3^2x_4^3$ *is divisible by* $LT(\phi_2(f_1))$.

The next step is, since $LT(\phi_2(g_2)) = -{x_4}^2LT(\phi_2(f_1))$

$$g_3 = -x_1{x_2}^2g_2 + {x_1}^6{x_2}^4{x_4}^2f_1 = x_1^9x_2^5x_3x_4^4 - x_1^5x_2^7x_3^2x_4^2$$

$\phi_2(g_3) = x_3x_4^4 - x_3^2x_4^2$, $LT(\phi_2(g_3)) = x_3x_4^4$ *is divisible by only* $LT(\phi_2(f_2))$.

We apply pseudo division again, since $LT(\phi_2(g_3)) = x_4LT(\phi_2(f_1))$

$$g_4 = x_1{x_2}^2g_3 + {x_1}^5{x_2}^7x_4f_1 = x_1^{10}x_2^7x_3x_4^4 - x_1^8x_2^8x_3x_4^3$$

$\phi_2(g_4) = x_3x_4^4 - x_3x_4^3$, $LT(\phi_2(g_4)) = x_3x_4^4$ *is divisible by any* $LT(\phi_2(f_2))$.

The next step is, since $LT(\phi_2(g_4)) = {x_4}^3LT(\phi_2(f_2))$,

$$g_5 = x_1^2x_2g_4 - {x_1}^{10}{x_2}^7{x_4}^3f_2 = x_1^{11}x_2^9x_4^5 - x_1^{10}x_2^9x_3x_4^3$$

$\phi_2(g_5) = -x_3x_4^3 + x_4^5$, $LT(\phi_2(g_5)) = -x_3x_4^3$ *is divisible by only* $LT(\phi_2(f_2))$.

We apply one more pseudo division, since $LT(\phi_2(g_5)) = -{x_4}^2 LT(\phi_2(f_2))$

$$g_6 = x_1^2 x_2 g_5 - {x_1}^{10}{x_2}^9{x_4}^2 f_2 = x_1^{13} x_2^{10} x_4^5 - x_1^{11} x_2^{11} x_4^4$$

$\phi_2(g_6) = x_4^5 - x_4^4$, $LT(\phi_2(g_5)) = x_4^5$ *is not divisible by* $LT(\phi_2(f_1))$ *or* $LT(\phi_2(f_2))$.

The algorithm came to end.

Now, we use back substition,

$$\begin{aligned} g_6 &= x_1^2 x_2 g_5 - {x_1}^{10}{x_2}^9{x_4}^2 f_2 \\ &= x_1^2 x_2 x_1^2 x_2 g_4 - {x_1}^{10}{x_2}^7{x_4}^3 f_2 - {x_1}^{10}{x_2}^9{x_4}^2 f_2 \\ &= x_1^3{x_2}^2 g_4 - ({x_1}^{12}{x_2}^8{x_4}^3 + {x_1}^{10}{x_2}^9{x_4}^2 f_2) f_2 \\ &= x_1^3{x_2}^2(x_1{x_2}^2 g_3 + {x_1}^5{x_2}^7 x_4 f_1) - ({x_1}^{12}{x_2}^8{x_4}^3 + {x_1}^{10}{x_2}^9{x_4}^2 f_2) f_2 \\ &= x_1^4{x_2}^4 g_3 - x_1^8{x_2}^9 x_4 f_1 - ({x_1}^{12}{x_2}^8{x_4}^3 + {x_1}^{10}{x_2}^9{x_4}^2 f_2) f_2. \end{aligned} \tag{3.1}$$

Contiuning this way,

$$g_6 = x_1^9{x_2}^9 f + ({x_1}^{10}{x_2}^8{x_3}^2 - {x_1}^9{x_2}^9 x_4 + x_1^{11}x_2^8{x_4}^2) f_1 - ({x_1}^{11}{x_2}^8 x_4 - {x_1}^{10}{x_2}^9{x_4}^2 + x_1^{12}x_2^8{x_4}^3) f_2.$$

Hence

$$x_1^9{x_2}^9 f = -({x_1}^{10}{x_2}^8{x_3}^2 - {x_1}^9{x_2}^9 x_4 + x_1^{11}x_2^8{x_4}^2) f_1 + ({x_1}^{11}{x_2}^8 x_4 - {x_1}^{10}{x_2}^9{x_4}^2 + x_1^{12}x_2^8{x_4}^3) f_2 + g_6.$$

is the desired expression.

Definition 3.4.4. *Let* f *and* g *be two binomials. Given a monomial ordering in* $k[x_{i+1}, \ldots, x_n]$*, compute the* S*- polynomial*

$$S(\phi_i(f), \phi_i(g)) = m_1 \phi_i(f) - c_2 m_2 \phi_i(g).$$

Since one of monomials f *and* g *must be of the form* $x_1^{\alpha_1} \ldots x_n^{\alpha_i} LM(\phi_i(f))$ *and*

$x_1^{\beta_1} \ldots x_i^{\beta_i} LM(\phi_i(g))$ *respectively, we define pseudo* S*-polynomial as*

$$S_{\phi_i}(f, g) = c_1 x_1^{\beta_1} \ldots x_i^{\beta_i} m_1 f - c_2 x_1^{\alpha_1} \ldots x_n^{\alpha_i} m_2 g.$$

Example 3.4.5. *Let* $f = x_1^2 x_2 x_3^2 x_4 - x_1 x_2^2 x_3 x_4^2$ *and* $g = x_1^2 x_3 x_4^2 - x_2^3 x_4^3$.

Computing projections

$$\phi_2(f) = x_3^2 x_4 - x_3 x_4^2$$

and

$$\phi_2(g) = x_3x_4^2 - x_4^3.$$

Then

$$S(\phi_2(f), \phi_2(g)) = x_4\phi_2(f) - x_3\phi_2(g) = 0.$$

The pseudo S- polynomial is

$$\begin{aligned} S_{\phi_2}(f,g) &= x_1^2x_4f - x_1^2x_2x_3g \\ &= x_1^2x_2^4x_3x_4^3 - x_1^3x_2^2x_3x_4^3. \end{aligned} \tag{3.2}$$

Theorem 3.4.6. *Let $I =< f_1, \ldots, f_s >\subseteq k[x_1, \ldots, x_n]$ be an ideal. The following algorithm obtains a generating set for the saturation ideal $I :< x_1, \ldots, x_n >$.*

Input $F = f_1, \ldots, f_s$, $t = s$

For $i = n - 1$ to 1

For each pair (j, k), $(j \neq k)$

Pseudo divide $S_{\phi_i}(f_j, f_k)$ by F

$r =$ remainder of division

If $\phi(r) \neq 0$

$t = t + 1, f_t = r/x_i^\infty, F = F \cup \{f_t\}$

$F = F/x_i^\infty$
Return F

In the algorithm F/x_i^∞ means divide each element of I by the highest power of x_i.

Proof. Notice that in each step of algorithm, we compute a Gröbner basis for $\phi_i(I)$. By construction, each new $f_t \in I :< x_1, \ldots, x_i >^\infty$. Hence, if $F = \{f_1, \ldots, f_k\}$ is the set of polynomials at the end of the algorithm, $< F >\subseteq I :< x_1, \ldots, x_i >^\infty$.

Conversely, if $f \in I :< x_1, \ldots, x_i >^\infty$, then $\phi_i(f) \in \phi_i(I)$ for some i. Since $\phi_i(F)$ contains a Gröbner basis for $\phi_i(I)$,$f \in< F >$. Hence $< F >= I :< x_1, \ldots, x_i >^\infty$. □

Example 3.4.7. *Let* $I =< f_1 = x_1x_3 - x_2x_4, f_2 = x_1^2x_2 - x_3x_4^2 >$.
Start with $i = 3$. $\phi_3(f_1) = x_4$, $\phi_3(f_2) = -x_4^2$.

Compute S*-polynomial,*

$$S(\phi_i(f_1), \phi_i(f_2)) = x_4\phi_3(f_1) - \phi_3(f_2) = x_4^2 - x_4^2 = 0.$$

Hence pseudo S*-polynomial is*

$$\begin{aligned} S_{\phi_3}(f_1, f_2) &= x_3x_4f_1 - x_2f_2 \\ &= x_1x_3^2x_4 - x_1^2x_2^2 = g_1. \end{aligned}$$

$\phi_3(g_1) = x_4$ *is divisible by* $LT(\phi_3(f_1))$.
Then, we can apply pseudo division

$$g_2 = S_{\phi_3}(g_1, f_1) = x_2g_1 + x_1x_3^2f_1 = x_1^2x_3^3 - x_1^2x_2^3, \phi_3(g_2) = 0.$$

Therefore, there is no new polynomial to generating set of ideal.
Let $i = 2$,

$$\begin{aligned} \phi_2(f_1) = x_3 - x_4, \phi_2(f_2) &= -x_3x_4^2 \\ S(\phi_2(f_1), \phi_2(f_2)) &= x_4^2(x_3 - x_4) - x_3x_4^2 \\ &= -x_4^3 \neq 0. \\ S_{\phi_2}(f_1, f_2) &= x_4^2f_1 - x_1f_2 = x_1^3x_2 - x_2x_4^3 = g_1. \end{aligned}$$

$\phi_2(g_1) = x_4^3$ *is not divisible either* $LT(\phi_2(f_1)) = x_3$ *or* $LT(\phi_2(f_2)) = -x_3x_4^2$. *We have a remainder, so we divide elements of* g_1 *with highest power of* x_2. *Hence* $f_3 = g_1/x_2 = x_1^3 - x_4^3$ *and* $\phi_2(f_3) = -x_4^3$, *so add new element* f_3 *to the generating set of ideal.*

Finally, let us consider the case $i = 1$,

$$\phi_1(f_1) = x_3 - x_2x_4, \phi_1(f_2) = x_2 - x_3x_4^2, \phi_1(f_3) = -x_4^3.$$

Applying pseudo division,

$$\begin{aligned} S(\phi_1(f_1), \phi_1(f_2)) &= x_3x_4(x_3 - x_2x_4) - x_2(x_2 - x_3x_4^2) \\ &= x_3^2x_4 - x_2^2 \neq 0. \end{aligned}$$

$$S_{\phi_1}(f_1, f_2) = x_3x_4f_1 - x_2f_2 = x_1x_3^3x_4 - x_1^2x_2^2 = g_1$$

$\phi_1(g_1) = x_3^2x_4 - x_2^2$, $LT(\phi_1(g_1)) = x_3^2x_4$ *is not divisible by any* $LT(\phi_1(f_j))$ *for* $j = 1, 2, 3$. *We have a remainder, divide elements of* g_1 *with highest power of* x_1.
Hence $f_4 = g_1/x_1 = x_3^2x_4 - x_1x_2^2$ *and* $\phi_1(f_4) = x_3^2x_4 - x_2^2 \neq 0$. *We will add new element* f_4 *to generating set of ideal* I.

The next step is,

$$S(\phi_1(f_1), \phi_1(f_4)) = x_3^2(x_3 - x_2x_4) - x_2(x_3^2x_4 - x_2^2)$$

$$= x_3^3 - x_2^3 \neq 0.$$

$$S_{\phi_1}(f_1, f_4) = x_3^2f_1 - x_2f_4 = x_1x_3^3 - x_1x_2^3.$$

$\phi_1(g_1) = x_3^2 - x_2^3$, $LT(\phi_1(g_1)) = -x_2^3$ *is not divisible by any* $LT(\phi_1(f_j))$ *for* $j = 1, 2, 3$. *We have a remainder, divide elements of* g_1 *with highest power of* x_1.
Therefore $f_5 = g_1/x_1 = x_3^3 - x_2^3$ *and* $\phi_1(f_5) = f_5$. *We will add new element* f_5 *to generating set of ideal* I.

One can show that the other pseudo S*-polynomials give zero remainders. Hence* $F = F/x_1^\infty = \{f_1, f_2, f_3, f_4, f_5\}$ *is a generating set for ideal* $I :< x_1x_2x_3x_4 >^\infty$.

This new method uses fewer variable than Sturmfel's method in each step. Hence it generally takes less time to compute a generating set for toric ideal.

4. INTEGER PROGRAMMING PROBLEMS

Introduction

Linear programming problem can be described as finding maximum (or minimum) value of a linear function subject to given linear constraints. Linear programming emerged as a discipline in World War II. It is used to solve complex planning problem in military. Later, it was occured that linear programming have applications in many areas such as commerce, military, industry, transportation, agriculture etc.

There are several algorithms to solve linear programming problems. The best known of these algorithms is the simplex method developed by George Dantzig. In this method, linear programming problem is converted in a standard form. Then it can be put in a matrix and solved by using elementary row operations. Hence it is assumed that the values of the solution variables can be rational numbers. This assumption is however unrealistic in real life. For example, in a production problem about furnitures, it does not make sense to come up with answer like "fifty five and a half chairs".

The brunch of linear programming in which all coefficients and variables are integer is called integer programming. Solving integer programming problems is much more difficult than solving linear programming problems. The most popular methods are Branch-and-Bound and Cutting-Plane algorithms. These algorithms involve several execution of simplex method. For understanding of details are these algorithms and simplex method we can recommend (Taha, 2006).

In this chapter we present a method for solving an integer programming problem using Gröbner bases. The method depends on defining a toric ideal for each integer programming problem. The method is first introduced by (Conti and Traverso, 1991). The subject further developed by (Sturmfels, 1995). We detailed Sturmfels' treatment of the subject.

Solving Integer Programming Problems

An integer programming problem in the standard form is

$$\begin{aligned} \min \quad & c_1x_1 + c_2x_2 + \ldots + c_nx_n \\ \text{subject to} \quad & \\ & a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ & a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ & \vdots \qquad\qquad\qquad\qquad \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m \\ & x_j \in \mathbb{Z}_{\geq 0}, \quad j = 1, \ldots, n. \end{aligned}$$

Each n-tuple $(x_1, x_2, \ldots, x_n)$ satisfying above equations is called a feasible point of the integer programming problem. Among the feasible points, a point minimizing of the value $c_1x_1 + c_2x_2 + \ldots + c_nx_n$ is called an optimal point.

We will translate an integer programming problem into a problem about polynomials. Let us introduce a new variable u_i for each equation above satisfiying

$$u_i^{\sum_{j=1}^{n} a_{ij}x_j} = u_i^{b_i}$$

for $i = 1, 2, \ldots, m$. Hence

$$\prod_{j=1}^{n}(\prod_{i=1}^{m} u_i^{a_{ij}})^{x_j} = \prod_{i=1}^{m} {u_i}^{b_i}.$$

The following proposition is immediate.

Proposition 4.2.1. *For a field k, let* $\phi : k[w_1, \ldots, w_n] \to k[u_1, \ldots, u_m, u_1^{-1}, \ldots, u_m^{-1}]$ *be a homomorphism given by*

$$w_j \mapsto u_1^{a_{j_1}}, u_2^{a_{j_2}}, \ldots, u_m^{a_{j_m}}$$

for $j = 1, 2, \ldots, n$. *Then* $(x_1, \ldots, x_n)$ *is a feasible point of the standard integer programming problem if and only if*

$$\phi(w_1^{x_1} w_2^{x_2} \ldots w_n^{x_n}) = u_1^{b_1} u_2^{b_2} \ldots u_n^{b_n}.$$

Definition 4.2.2. *Given a standard integer programming problem, we define a weight vector as follows. If each $c_j \geq 0$ for $j = 1, 2, \ldots, n$, then $w = (c_1, c_2, \ldots, c_n)$. Otherwise we assume that all $a_{ij} \geq 0$. In this case, there exists a sufficiently large positive integer λ such that*

$$w_j = c_j + \lambda \sum_{i=1}^{m} a_{ij} \geq 0.$$

Hence $w = (w_1, w_2, \ldots, w_n)$ will be the desired weight vector.

Now, we are ready connect integer programming problem into toric ideals. The following theorem is basically equivalent to Theorem 5.5 in (Sturmfels, 1995).

Theorem 4.2.3. *(Sturmfels, 1996, Theorem 5.5)*

Given an integer programming problem in the standard form. Let $A = (a_{ij})$ be the coefficient matrix of the integer programming problem. Suppose that w is the weight vector defined above and G is a Gröbner basis of the toric ideal I_A with respect to wlex order. If $(x_1, \ldots, x_n)$ is a feasible solution and the remainder of $(w_1^{x_1} w_2^{x_2} \ldots w_n^{x_n})$ upon division by G is $(w_1^{\gamma_1} w_2^{\gamma_2} \ldots w_n^{\gamma_n})$, then $(\gamma_1, \ldots, \gamma_n)$ is an optimal solution of the integer programming problem.

Proof. Since $(x_1, \ldots, x_n)$ is a feasible point

$$\phi(w_1^{x_1} w_2^{x_2} \ldots w_n^{x_n}) = u_1^{b_1} u_2^{b_2} \ldots u_n^{b_n}.$$

Moreover, we know that the toric ideal I_A is the kernel of ϕ. Since G is a Gröbner basis of I_A,

$$(w_1^{x_1} w_2^{x_2} \ldots w_n x_n) - (w_1^{\gamma_1} w_2^{\gamma_2} \ldots w_n^{\gamma_n}) \in I_A = ker(\phi).$$

Hence

$$\phi(w_1^{\gamma_1} w_2^{\gamma_2} \ldots w_n^{\gamma_n}) = u_1^{b_1} u_2^{b_2} \ldots u_n^{b_n}$$

implies that $(\gamma_1, \ldots, \gamma_n)$ is also a feasible point. We have to show that $(\gamma_1, \ldots, \gamma_n)$ is an optimal point. Suppose the contrary. Hence there exists a feasible point $(z_1, z_2, \ldots, z_n)$ such that

$$c_1 z_1 + c_2 z_2 + \ldots + c_n z_n < c_1 \gamma_1 + \ldots + c_n \gamma_n.$$

This implies $w_1^{z_1} w_2^{z_2} \ldots w_n^{z_n} <_{wlex} w_1^{\gamma_1} w_2^{\gamma_2} \ldots w_n^{\gamma_n}$ which is a contradiction to $w_1^{\gamma_1} w_2^{\gamma_2} \ldots w_n^{\gamma_n}$ being remainder upon division by G since

$$(w_1^{x_1} w_2^{x_2} \ldots w_n x_n) - (w_1^{\gamma_1} w_2^{\gamma_2} \ldots w_n^{\gamma_n}) \in \ker(A) = I_A.$$

□

This theorem gives a method for finding an optimal solution of an integer programming problem in the standard form. First of all, compute a Gröbner basis G of the toric ideal I_A with respect to the desired weighted order. Then find a feasible solution $(x_1, \ldots, x_n)$ and compute the remainder of $(w_1^{x_1} w_2^{x_2} \ldots w_n^{x_n})$ upon division by G. If the remainder is $w_1^{\gamma_1} w_2^{\gamma_2} \ldots w_n^{\gamma_n}$, then $(\gamma_1, \ldots, \gamma_n)$ is an optimal solution. Let us illustrate this method with a couple of examples.

Example 4.2.4. *We will solve the following integer programming problem*

$$\textit{min} \qquad 2x_1 + 3x_2 + 5x_5 + 5x_6$$

subject to

$$\begin{aligned} 2x_1 + x_2 + x_3 &= 16 \\ x_1 + 3x_2 - x_4 + x_5 &= 20 \\ x_1 + x_2 + x_6 &= 10 \\ x_i \in \mathbb{Z}_{\geq}, \quad i &= 1, 2, \ldots, 6. \end{aligned}$$

The coefficient matrix of the problem is

$$A = \begin{bmatrix} 2 & 1 & 1 & 0 & 0 & 0 \\ 1 & 3 & 0 & -1 & 1 & 0 \\ 1 & 2 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

If we compute a generaring set for corresponding toric ideal using techniques given in the previous chapter, we have

$$I_A = < x_4x_5 - 1, x_2 - x_3x_5^3x_6, x_1 - x_3^2x_5x_6 > .$$

We found the reduced Gröbner basis of I_A with repect to wlex order where $w = (2, 3, 0, 0, 5, 5)$ in Example 2.4.3.

Hence

$$G = \{x_4x_5 - 1, x_2 - x_3x_5^3x_6, x_1 - x_3^2x_5x_6\}.$$

It is clear that $(0, 0, 16, 0, 20, 10)$ *is a feasible solution of the given integer programming problem. If we divide* $x_3^{16}x_5^{20}x_6^{10}$ *by G, then the remainder is* $x_1^5x_2^5x_3$*. Hence* $(5, 5, 1, 0, 0, 0)$ *is an optimal solution.*

Example 4.2.5. *Consider the following integer programming problem*

$$\textit{max} \quad 50x_1 + 30x_2$$

subject to

$$2x_1 + x_2 \leq 14$$

$$5x_1 + 5x_2 \leq 40$$

$$x_1 + 3x_2 \leq 18$$

$x_1, x_2 \geq 0$ *integers.*

If we convert this problem into standard form,

$$\textit{min} \quad -50x_1 - 30x_2$$

subject to

$$\begin{aligned} 2x_1 + x_2 + x_3 &= 14 \\ 5x_1 + 5x_2 + x_4 &= 40 \\ x_1 + 3x_2 + x_5 &= 18 \end{aligned}$$

$x_i \geq 0$ *integers.*

The coefficient matrix of the problem is

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

Using computation technique of toric ideals is

$$I_A = < x_2 - x_3x_4^5x_5^3, x_1 - x_3^2x_4^5x_5 > .$$

To obtain weight we must find a positive integer λ *such that*

$$-50 + 8\lambda > 0 \quad \textit{and} \quad -30 + 9\lambda > 0.$$

So we can take $\lambda = 7$. The weight vector is

$$w = (-50, -30, 0, 0, 0) + 7(8, 9, 1, 1, 1) = (6, 33, 7, 7, 7).$$

Then we compute reduced Gröbner basis of I_A with respect to wlex order which is equal

$$G = \{x_4x_5 - 1, x_2 - x_3x_5^3x_6, x_1 - x_3^2x_5x_6\}.$$

It is clear that $(0, 0, 14, 40, 18)$ is a feasible solution of the given integer programming problem. If we divide $x_3^{14}x_4^{40}x_5^{18}$ by G, then the remainder become $x_1^6x_2^2x_5^6$. Hence $(6, 2, 0, 0, 6)$ is an optimal solution of the integer programming problem. We did not give the details of the computations in previous examples, we made necessary computations of Gröbner bases using Mathematica.

Notice that in the last example we have negative values in the objective function but all entries of the coefficient matrix are non-negative. Hence we able to find a non-negative weight vector. If both objective function and the coefficient matrix have negative entries, Sturmfel's method does not work. In this case, the only known solution of the problem is the classical method of (Conti and Traverso 1991).

5. CONCLUSIONS AND RECOMMENDATIONS

Using Gröbner bases is very effective technique for solving integer programming problems. The most important part of this technique is the computation of saturation of the toric ideals. This can be done using Gröbner bases. Since Buchberger's algorithm for Gröbner basis computation is sensitive to the number of variables, the main goal is to compute Gröbner bases with less variable possible. We try to make a contribution in this direction.

There are special kind of integer programming problems such as transportation problems, scheduling problems and assignment problems. These problems have special kind of corresponding matrices. The computation of toric ideals and saturations for this special matrices can be also investigated in further studies.

REFERENCES

Adams W and Loustaunau P (1994) An Introduction to Gröbner Bases, American Mathematical Society, Graduate Studies in Mathematics, 3, Rhode Island.

Blanco V and Puerto J (2009) "Partial gröbner bases for multiobjective integer linear optimization", J. SIAM Discrete Math., 23:571–595.

Blanco V and Puerto J (2011) "Some algebraic methods for solving multiobjective polynomial integer programs", J. Symbolic Comput., 46:511–533.

Buchberger B (1965) "An algorithm for finding a basis for the residue class ring of a zero-dimensional polynomial ideal", Ph. D. thesis, Universitat Innsbruck, Austria, 41:475–511.

Conti P and Traverso C (1991) "Buchberger algorithm and integer programming", Lecture Notes in Computer Science, 539:130–139.

Cox D, Little J, and O'Shea D (1992) Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra, First Edition, Springer, New York.

Cox D, Little J, and O'Shea D (1998) Using Algebraic Geometry, First Edition, Springer, Newyork.

Hosten S and Sturmfels B (1995) "Grin: An implementation of gröbner bases for integer programming, in: Integer programming and combinatorial optimization", Lecture Notes in Computer Science, 920:207–276.

Li Q, Yi G, Darlington J, and Ida T (2001) "Minimised geometric buchberger algorithm for integer programming", Annals of Operations Research, 108:87–109.

Schrijver A (1986) Theory of linear and integer programming, John Wiley and Sons, New York.

Sturmfels C (1996) Gröbner Bases and Convex Polytopes, American Mathematical Society, University Lectures Series, 8, Rhode Island, USA.

Taha H (2011) Operations research, An introduction, Tenth Edition, Pearson, New York.

Thomas R R (1995) "A geometric buchberger algorithm for integer programming", Mathematics of Operations Research, 20:864–884.

Thomas R R and Weismantel R (1997) "Truncated gröbner bases for integer programming", Applicable Algebra in Engineering, Communication and Computing, 8:241–257.