Generalized reduction to compute toric ideals

Deepanjan Kesh ** and Shashank K. Mehta

Indian Institute of Technology, Kanpur - 208016, India,
{deepkesh,skmehta}@cse.iitk.ac.in

Abstract. Toric ideals have many applications including solving integer programs. Several algorithms for computing the toric ideal of an integer matrix are available in the literature. Since it is an NP hard problem the present approaches can only solve relatively small problems. We propose a new approach which improves upon a well known saturation technique.

1 Introduction

Let A be an integer matrix, and let $ker(A)$ be the lattice kernel of A, i.e., integer solutions of $Au = 0$. For any $u \in ker(A)$, let u_+ denote the vector such that $u_+[i] = u[i]$ if $u[i] > 0$ else $u_+[i] = 0$. Vector $u-$ is given by $u_+ - u$. For any positive integer vector $v \in \mathbb{N}^n$ monomial $x_1^{v[1]} x_2^{v[2]} \cdots x_n^{v[n]}$ is concisely denoted by x^v. The polynomial ideal generated by $\{x^{u+} - x^{u-} | u \in ker(A)\}$ is called the *toric ideal* of A and it is denoted by I_A. In this paper we address the problem of computing a generator of I_A, which we loosely call the problem of computing a toric ideal.

This problem has some useful applications including solving integer programs [1–3], computing primitive partition identities [4] chapters 6 and 7, and solving scheduling problem [5] among a few others.

There are several algorithm in the literature to compute I_A for a given $d \times n$ matrix A. Each of these algorithms requires the computations of one or more Gröbner bases. These include an algorithm, emerging as an application of Theorem 2 section 3.3 [6], which involves the computation of a Gröbner basis of an ideal in a polynomial ring over $n + d - 1$ variables. Urbanke [7] proposed an algorithm which involves of $O(n)$ Gröbner basis computations, all in an n-variable ring. The algorithm by Sturmfels, although very different in nature, involves similar computation and similar performance. Both these algorithms are significantly more efficient than the first algorithm since Büchberger's algorithm for computing Gröbner basis is very sensitive to the number of variables. Bigatti et.al. [8] improved Sturmfels' algorithm, but it appears that in the worst case their algorithm may not fare better than Sturmfels'. Recently Hemmecke and Malkin [9] have proposed a new approach project and lift which involves the computation of one Gröbner basis in a ring of j variables for $j = 1, 2, \ldots, n$. Their algorithm shows significant improvement over the prevailing best algorithms.

** This work was partly supported by Microsoft Research India through 2006 MSR India PhD Fellowship Program

We present an algorithm that also requires the computation of one Gröbner basis in $k[x_1, \cdots, x_i]$ for each i. Unlike *project and lift* we continue to perform saturation at each step by one variable as in Sturmfels' algorithm. Therefore we also require the computation of one Gröbner basis in each ring $k[x_1, \cdots, x_i]$. In our approach the basis is computed with respect to reverse lexicographic term order, which is known to be the most efficient among all term orders, [10, 11].

The remainder of the paper is organized as follows. The next section introduces essential background concepts and notations. Section 3 discusses surjective ring homomorphisms and some interesting properties. Section 4 gives a simple algorithm to compute saturation of a non-homogeneous ideal. In section 5 we give basic steps to perform Gröbner basis computation in a limited sense. We compute a basis which itself is not a Gröbner basis but its projection in a subspace is the Gröbner basis for the projected ideal. In the sixth section we describe the new algorithm to compute the saturation of a binomial ideal. Last section gives experimental results and conclusions.

2 Background

Suppose V is a lattice kernel basis, i.e., a basis of $ker(A)$ which generates the kernel vectors with integer coefficients. Let J_V be the ideal generated by $\{x^{u+} - x^{u-} | u \in V\}$. Then $I_A = J_V : (x_1 \dots x_n)^\infty$ where the r.h.s. denotes the ideal $\{f \in k[\mathbf{x}] | x^\alpha \cdot f \in J_V \ for\ some\ x^\alpha\}$ called the *saturation* of J. Thus the computation of a toric ideal has two steps: computation of lattice kernel basis and the saturation of J_V. The first step has a polynomial time solution by computing the Hermite normal form of A. Therefore the complex step is the saturation computation.

One of the most useful ideas in computational commutative algebra is Gröbner basis of an ideal. It has found many applications in computations related to ideal [12, 6]. The first and the best known algorithm to compute a Gröbner basis is due to Büchberger [13]. It turns out that it is useful in the computation of the saturation of an ideal.

One of the initial attempts to compute I_A was to compute the elimination ideal of the ideal generated by $\{t_0 t_1 \cdots t_d - 1, x_1 t^{a_1-} - t^{a_1+}, \dots, x_n t^{a_n-} - t^{a_n+}\}$, where a_i is the i-th row of A and t are new $d+1$ variables, by eliminating t's. Here A is $d \times n$. The elimination ideal can be computed by computing the Gröbner basis of the ideal under a specific term order. However, the algorithm is too slow owing to the fact that the complexity of the Büchberger's algorithm is sensitive to the number of variables involved. A wiser choice is to work in the n variables in which J lies.

An algorithm working, in n variables, for saturation is due to Urbanke [7]. It uses the fact that if the spanning set of the lattice $ker(A)$ contains a vector with all positive coordinates, then $J = I_A$. In general, this might not be true and to ensure this in A, the algorithm replaces a certain columns of A by their negation till this holds true. Let new matrix be A' and corresponding lattice kernel basis be V'. Then $I_{A'} = J_{V'}$. With this ideal as the starting point, it

reaches I_A by replacing the columns by their original form, one at a time, and each such step requires computing a Gröbner basis. This approach will require at most $n/2$ Gröbner bases of n variables each.

Another algorithm which also works in n variables is due to Sturmfels [14, 4]. Their algorithm makes use of theorem 3 (lemma 12.1, chapter 12 of [4]) and the fact that $(J : (x_1 x_2 \dots x_n)^\infty) = ((\dots(J : x_1^\infty)\dots) : x_n^\infty)$. The algorithm first computes the basis B of J_V from the Hermite Normal form of A. Then it performs n iterations of Gröbner basis computation starting with B. In the i-th iteration it computes the Gröbner basis with a graded reverse lexicographic term order with x_i least and removes the largest common factor of the form x_i^α from each member of the computed basis. It was reported in [14] that the performances of the two algorithms are generally same. Bigatti et.al. [8] improve Sturmfels' algorithm.

Hemmecke and Malkin [9] presented an entirely new approach called *project and lift* . Let I_j denote the ideal I_A projected on $x_1, \dots, x_j$, i.e., by setting $x_{j+1}, dots, x_n$ to 1. They begin by computing the Gröbner basis of I_1 and build their way up to I_A, one dimension at a time. In the i-th step they compute a certain grading vector r_i, and a Gröbner basis w.r.t. a term order based on r in the ring $k[x_1, \cdots, x_i]$. This approach exploits the fact that Gröbner basis computation is very sensitive to the number of variables in the polynomial ring containing the ideal.

A closer look at the algorithm due to Sturmfels as well as our algorithm would reveal that both of these algorithms do not explicitly exploit the primality of the toric ideal I_A and hence both of these algorithms can be applied to any binomial ideals. That is, both these algorithms compute the saturation of any binomial ideal (one that has at least one basis containing only binomials.)

Let $k[\mathbf{x}]$ denote the polynomial ring $k[x_1, \dots, x_n]$ over a field k. Also, let $B \subset k[\mathbf{x}]]$ be a set of polynomials. Then, we denote by $\langle B \rangle$, the ideal generated by the polynomials in B. A monomial $x_1^{\alpha_1} \dots x_n^{\alpha_n}$ would normally be compactly written as x^α.

A total ordering $\prec$ on the monomials of $k[\mathbf{x}]$ is called a term order if it satisfies following properties: (i) it is a well-ordering (Artinian) and (ii) $x^\alpha \prec x^\beta \Rightarrow x^{\alpha+\gamma} \prec x^{\beta+\gamma}$ for all α, β, γ. Given a term order $\prec$ on $k[\mathbf{x}]$, we denote the leading term of a polynomial f by $in_\prec(f)$. One particular term order is being repeatedly used in this paper - the *graded reverse lexicographic order*. This is frequently the most efficient ordering to compute Gröbner basis. Let $\boldsymbol{d}$ be a grading vector and $\prec_{\boldsymbol{d}}$ be the graded reverse lexicographic order. Then, given two monomials x^α and x^β, we say $x^\alpha \prec_{\boldsymbol{d}} x^\beta$ if $\alpha \cdot \boldsymbol{d} < \beta \cdot \boldsymbol{d}$ or if they are equal, then the last non-zero coordinate in $\alpha - \beta$ is negative. One more aspect of this term order that would now and again come to the fore is the least variable of the order. So, we would by $\prec_{\boldsymbol{d},i}$ represent a graded reverse lexicographic ordering with x_i as the least element.

Let I be an ideal in $k[\mathbf{x}]$ and $\prec$ be a term order. By $in_\prec(I)$ we denote the set $\{in_\prec(f) | f \in I\}$ which is called the initial ideal if I because it is a monomial ideal.

If $G = \{f_1, \ldots, f_m\}$ is a basis of I such that $in_\prec(I) = \langle in_\prec(f_1), \ldots, in_\prec(f_n)\rangle$, then G is called a Gröbner basis of I.

Let f be a polynomial and B be a set of polynomials in $k[\mathbf{x}]$. If there is $f_i \in B$ and a term $c \cdot x^\alpha$ in f such that $in_\prec(f_i)$ divides $c \cdot x^\alpha$ and if the quotient is $c' \cdot x^\beta$ then $f \rightarrow f - c' \cdot x^\beta * f_i$ is called a *reduction* step. If a sequence of reductions lead to f' by the polynomials of B such that f' is irreducible by any member of B, then we say that f is reduced to f' by B. If G is a Gröbner basis such that each $f \in G$ is irreducible by $G \setminus \{f\}$, then G is called *reduced* Gröbner basis. This basis is unique for a given ordering. We will denote it by $\mathcal{G}_\prec(I)$.

3 Surjective ring homomorphism

In this paper ϕ will denote a *surjective* ring homomorphism $k[\mathbf{x}] \rightarrow k[\mathbf{y}]$, where $k[\mathbf{x}]$ denotes $k[x_1 \ldots, x_n]$ and $k[\mathbf{y}]$ denotes $k[y_1, \ldots, y_m]$.

Definition 1. *Let $S \subseteq k[\mathbf{x}]$ be a set. Then, we define $\phi(S) = \{\phi(f) \mid f \in S\}$.*

Lemma 1. *Let $f_1, \ldots, f_s \in k[\mathbf{x}]$. Then, $\phi(\langle f_1, \ldots, f_s\rangle) = \langle \phi(f_1), \ldots, \phi(f_s)\rangle$.*

Proof. Let $f' \in \phi(\langle f_1, \ldots, f_s\rangle)$. Then, $\exists f \in \langle f_1, \ldots, f_s\rangle$ such that $\phi(f) = f'$. Then $\exists g_1, \ldots g_s \in k[\mathbf{x}]$ such that $f = \sum_i g_i f_i$. It implies $\phi(f) = \phi(\sum_i g_i f_i)$. Hence $f' = \sum_i \phi(g_i)\phi(f_i)$. Which implies $f' \in \langle \phi(f_1), \ldots, \phi(f_s)\rangle$.

Conversely, let $f' \in \langle \phi(f_1), \ldots, \phi(f_s)\rangle$. Then, $\exists g'_1, \ldots, g'_s \in k[\mathbf{y}]$ such that $f' = \sum_i g'_i \phi(f_i)$. From surjectivity, $\exists g_1, \ldots, g_s \in k[\mathbf{x}]$ such that $\phi(g_1) = g'_1, \ldots,$ $\phi(g_s) = g'_s$. So, $f' = \sum_i \phi(g_i)\phi(f_i) = \phi(\sum_i g_i f_i)$. Since $\sum_i g_i f_i \in \langle f_1, \cdots, f_s\rangle$, $f' \in \phi(\langle f_1, \cdots, f_s\rangle)$. ■

Definition 2. *Kernel of a homomorphism ϕ is $ker(\phi) = \{f \in k[\mathbf{x}] \mid \phi(f) = 0\}$.*

From the first theorem of isomorphism, $k[\mathbf{x}]/ker(\phi)$ is isomorphic to $k[\mathbf{y}]$. We shall denote this isomorphism by Φ.

Definition 3. *Let S be a subset of $k[\mathbf{x}]$. Then $\phi^{-1}(S) = \{f \in k[\mathbf{x}] | \phi(f) \in S\}$.*

Observation 1 *Let J be an ideal in $k[\mathbf{y}]$. Then $\phi^{-1}(J)$ is an ideal. Also J, $\Phi^{-1}(J)$, and $\phi^{-1}(J)/ker(\phi)$ are isomorphic.*

Projections are examples of surjective homomorphisms which will be used in algorithms discussed in this paper.

Definition 4. *Let V be a set of variables and $V' \subset V$. Then the map $\phi : k[V] \rightarrow k[V \setminus V']$ is a said to be a projection map if $\phi(f) = f|_{x=1 \forall x \in V'}$. We shall denote the projections $k[\mathbf{x}] \rightarrow k[x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n]$, $k[x_i, \cdots, x_{i+j}, z] \rightarrow k[x_i, \cdots, x_{i+j}]$, and $k[\mathbf{x}] \rightarrow k[x_{i+1}, \ldots, x_n]$ by π_i, π_z, and Π_i respectively.*

Observation 2 *π_i, π_y and Π_i are surjective ring homomorphisms.*

4 Homogeneous polynomials and saturation

4.1 Homogenization

Definition 5. *Let $f \in k[\mathbf{x}]$ and $\boldsymbol{d} \in \mathbb{N}^n$. We say f is homogeneous w.r.t. $\boldsymbol{d}$ if for all monomials $x^\alpha \in f$, $\boldsymbol{d} \cdot \alpha$ are equal. Vector $\boldsymbol{d}$ is called the grading vector.*

Let $\boldsymbol{d} \in \mathbb{N}^{n+1}$ be a 0/1 vector such that $d_{n+1} = 1$. Now we define a mapping $h_{\boldsymbol{d}} : k[\mathbf{x}] \to k[\mathbf{x}, z]$ such that $h_{\boldsymbol{d}}(f)$ is homogeneous w.r.t. $\boldsymbol{d}$ for any $f \in k[\mathbf{x}]$. Let $f = \sum_i c_i x^{\alpha_i} \in k[\mathbf{x}]$. Let $\alpha'_i \in \mathbb{N}^{n+1}$ such that its 1 to n components coincide with those of α_i and its $n+1$ component is 0. Let $c_{i_0} x^{\alpha_{i_0}}$ be a term in f such that $\alpha'_{i_0} \cdot \boldsymbol{d} \geq \alpha'_i \cdot \boldsymbol{d}$ for all i. Let $\delta_i = (\alpha'_{i_0} - \alpha'_i) \cdot \boldsymbol{d}$. Define $h_{\boldsymbol{d}}(f) = \sum_i x^{\alpha_i} z^{\delta_i}$. We shall denote $h_{\boldsymbol{d}}(f)$ by $\tilde{f}$ when $\boldsymbol{d}$ is known from the context. Observe that $\pi_z(\tilde{f}) = f$. If $B = \{f_i\}_i$ is a set of polynomials of $k[\mathbf{x}]$, then by homogenization of B we would mean the set $\tilde{B}$ given by $\{\tilde{f}_i\}_i$.

4.2 Colon Ideals

Definition 6. *Let $J \subseteq k[\mathbf{x}]$ be an ideal. Then $J : x_i^\infty$ denotes the ideal $\{f \in k[\mathbf{x}] \mid x_i^a f \in J \text{ for some } a\}$. $(\ldots(J : x_1^\infty)\ldots) : x_i^\infty$ is equal to $J : (x_1 \cdots x_i)^\infty$ which is given by $\{f \in k[\mathbf{x}] \mid x^\alpha f \in J \text{ for some } \alpha \in \mathbb{N}^n\}$.*

In general the computation of $J : x_i^\infty$ is expensive, see section 4 in chapter 4 of [6]. But in a special case when J is a homogeneous ideal an efficient method to compute $J : x_i^\infty$ is known as described in the following theorem.

Notation Let f be a polynomial and a be the largest integer such that x_j^a divides f, then we denote the quotient of the division by $f \div x_j^\infty$. If B be a set of polynomials, then $B : x_i^\infty$ denotes the set $\{f \div x_i^\infty | f \in B\}$.

Theorem 3 (lemma 12.1, [4]). *Let $J \subseteq k[\mathbf{x}]$ be a homogeneous ideal w.r.t. the grading vector $\boldsymbol{d}$. Also let $\mathcal{G}_{\prec_{\boldsymbol{d}},j}(J) = \{f_i\}_i$. Then $\{f_i \div x_j^\infty\}_i$ is a Gröbner basis of $J : x_j^\infty$.*

A trivial lemma follows.

Lemma 2. *Let $J \subseteq k[\mathbf{x}]$ be any ideal. Then $\pi_i(J : x_j^\infty) = \pi_i(J) : x_j^\infty$.*

Using these results, Algorithm 2 computes $J : x_j^\infty$ when ideal J is not homogeneous.

5 Shadow algorithms under a surjective homomorphism

Let I be an ideal in $k[\mathbf{x}]$. Lemma 1 shows that $\phi(I)$ is an ideal in $k[\mathbf{y}]$. In this section we show how to compute a basis B of I such that $\phi(B)$ is a Gröbner basis of $\phi(I)$.

Let α and β be two vectors in $\mathbb{N}^n$, and let $\alpha[i]$ and $\beta[i]$ denotes their i^{th} components. Then, $\alpha \vee \beta$ is the vector given by $(\alpha \vee \beta)[i] = max\{\alpha[i],\ \beta[i]\}$.

Data: A generating set, B, of an ideal $J \subseteq k[\mathbf{x}]$; An index i; A grading vector $\boldsymbol{d} \in \mathbb{N}^{n+1}$ be the vector with all components one.
Result: The Gröbner basis of $\langle B \rangle : x_i^\infty$

1 $\tilde{B} := \{\tilde{f} | f \in B\}$;
2 Compute $\mathcal{G}_{\prec_{\boldsymbol{d},i}}(\langle \tilde{B} \rangle)$;
3 Compute $B' = \{\ f \div x_i^\infty \mid f \in \mathcal{G}_{\prec_{\boldsymbol{d},i}}(\langle \tilde{B} \rangle)\ \}$;
4 **return** $\pi_y(B')$.

Algorithm 2: Computation of $\langle B \rangle : x_i^\infty$

In this and the next section we will assume the existence of an oracle which computes any one member h of $\phi^{-1}(m)$ for any monomial $m \in k[\mathbf{y}]$. With an abuse of the notation we shall denote this by $h := \phi^{-1}(m)$ as a step in the algorithms given below.

Let $\prec$ denote a term order in $k[\mathbf{y}]$. Consider any $h_1, h_2 \in k[\mathbf{y}]$. Let $c_1 y^{\alpha_1} = in_\prec(h_1)$ and $c_2 y^{\alpha_2} = in_\prec(h_2)$. Also let $\beta_1 = (\alpha_1 \vee \alpha_2) - \alpha_1$ and $\beta_2 = (\alpha_1 \vee \alpha_2) - \alpha_2$. Then the S-polynomial of these polynomials is given by $S(h_1, h_2) = c_2 y^{\beta_1} h_1 - c_1 y^{\beta_2} h_2$. Observe that if $in_\prec(h_2)$ divides $in_\prec(h_1)$, then $S(h_1, h_2)$ is the reduction of h_1 by h_2. Algorithm 3 computes $g_1, g_2 \in k[\mathbf{x}]$ for given $f_1, f_2 \in k[\mathbf{x}]$ such that $\phi(g_1)\phi(f_1) - \phi(g_2)\phi(f_2) = S(\phi(f_1), \phi(f_2))$.

Data: $f_1, f_2 \in k[\mathbf{x}]$, a surjective ring homomorphism $\phi : k[\mathbf{x}] \to k[\mathbf{y}]$, a term order $\prec$ over $k[\mathbf{y}]$, an oracle that computes any one member of $\phi^{-1}(m)$ for any monomial m of $k[\mathbf{y}]$
Result: Two polynomials $g_1, g_2 \in k[\mathbf{x}]$ such that $\phi(f_1 g_1 - f_2 g_2) = S(\phi(f_1), \phi(f_2))$

1 Let $c_1 \cdot y^{\alpha_1} = in_\prec(\phi(f_1)),\ c_2 \cdot y^{\alpha_2} = in_\prec(\phi(f_2))$;
2 $\beta_1 := (\alpha_1 \vee \alpha_2) - \alpha_1$;
3 $\beta_2 := (\alpha_1 \vee \alpha_2) - \alpha_2$;
4 **return** $g_1 := \phi^{-1}(c_2 y^{\beta_1}); g_2 := \phi^{-1}(c_1 y^{\beta_2})$;

Algorithm 3: $\mathcal{A}(f_1, f_2, \phi, \prec)$: computation of g_1, g_2 for given f_1, f_2

Observation 4 $(g_1, g_2) = \mathcal{A}(f_1, f_2, \phi, \prec) \Rightarrow \phi(g_1 f_1 - g_2 f_2) = S(\phi(f_1), \phi(f_2))$.

5.1 Generalized division algorithm

Let $g, g_1, \cdots, g_s \in k[\mathbf{y}]$ and $\prec$ be a term order in $k[\mathbf{y}]$. Then $g = \sum_i q_i g_i + r$ is said to be a *standard* expression for g if (i) $in_\prec(q_i g_i) \preceq in_\prec(g)\ \forall i$ and (ii) no monomial of r is divisible by $in_\prec(g_i)$ for any i, i.e., no monomial of r belongs to $\langle \{in_\prec(g_i) | 1 \leq i \leq s\} \rangle$. Standard expression generalizes the concept of division of a polynomial by another polynomial to the division of a polynomial by a set of polynomials. Here r is called the remainder and q_i are called the quotients of the division of g by $\{g_1, \cdots, g_s\}$. The algorithm to perform such a division is well known, see section 3 in chapter 2 of [6]. Let $f, f_1, \cdots, f_s \in k[\mathbf{x}]$. In Algorithm 4

we present a pseudo-division algorithm for f by $f_1, \cdots, f_s$ such that its image in $k[\mathbf{y}]$ gives a standard expression for $\phi(f)$ w.r.t. $\phi(f_1), \cdots, \phi(f_s)$.

Data: $f \in k[\mathbf{x}]$; $\{f_1, \ldots, f_s\} \subset k[\mathbf{x}]$; a surjective ring homomorphism, $\phi : k[\mathbf{x}] \to k[\mathbf{y}]$; a term order $\prec$ over $k[\mathbf{y}]$; an oracle to compute one member of $\phi^{-1}(m)$ for any monomial m of $k[\mathbf{y}]$.
Result: $\bar{f}, q_1, \ldots, q_s, r \in k[\mathbf{x}]$ such that $\bar{f}f = \sum_j q_j f_j + r$ and $\phi(\bar{f})\phi(f) = \sum_j \phi(q_j)\phi(f_j) + \phi(r)$ is a standard expression for $\phi(f)$ under $\prec$, where $\phi(\bar{f})$ is a constant

```
1  f̄ := 1; q_1 := 0, ..., q_s := 0; r := 0;
2  p := f;
3  while φ(p) ≠ 0 do
4  |  if ∃ i s.t. in_≺(φ(f_i)) divides in_≺(φ(p)) then
5  |  |  (g_1, g_2) := A(p, f_i, φ, ≺);
   |  |  /* φ(g_1) is a constant                                   */
6  |  |  f̄ := f̄ * g_1; q_1 := q_1 * g_1, ..., q_s := q_s * g_1; r := r * g_1;
7  |  |  p := p * g_1 − f_i * g_2;
8  |  |  q_i := q_i + g_2
9  |  end
10 |  else
11 |  |  g_1 := φ^{-1}(in_≺(φ(p)));
12 |  |  r := r + g_1;
13 |  |  p := p − g_1
14 |  end
15 end
   /* φ(p) = 0                                                     */
16 r := r + p.
```

Algorithm 4: SHADOW_DIV($f, \{f_1, \ldots, f_s\}, \phi, \prec$)

Observe that the leading term of $\phi(p)$ strictly decreases after each pass of the while loop. Combining with this fact that $\prec$ is a well-ordering we observe that the algorithm terminates. Also observe that $\bar{f} \cdot f = \sum_j q_j f_j + r + p$ is an invariant of the loop. Thus we have the following claim.

Lemma 3. *Algorithm 4, SHADOW_DIV($f, f_1, \ldots, f_s, \phi, \prec$), terminates to give $\bar{f} \cdot f = \sum_j q_j \cdot f_j + r$ and $\phi(f) = (1/\phi(\bar{f}))(\sum_j \phi(q_j)\phi(f_j) + \phi(r))$ is a standard expression for $\phi(f)$ under $\prec$, where $\phi(\bar{f})$ is a non-zero constant.*

5.2 Büchberger's Algorithm with generalized division

Now we present Algorithm 5 to compute a basis of any ideal in $k[\mathbf{x}]$ such that the image of the basis under ϕ is a Gröbner basis of the image of the ideal.

Lemma 4. *Algorithm 5 terminates.*

Data: $B = \{ f_1, \ldots, f_s \} \subseteq k[\mathbf{x}]$; a surjective ring homomorphism $\phi : k[\mathbf{x}] \to k[\mathbf{y}]$, a term order, $\prec$, in $k[\mathbf{y}]$

Result: A subset $C_1 \subset k[\mathbf{x}]$ such that $\langle C_1 \rangle = \langle B \rangle$ and $\phi(C_1)$ is a Gröbner basis of $\phi(\langle B \rangle)$, a subset $C_2 \subset k[\mathbf{x}]$ such that $\phi(C_2)$ is the reduced Gröbner basis of $\phi(\langle B \rangle)$ and for each $f \in \langle B \rangle$ there is $h \in \phi^{-1}(1)$ such that $h \cdot f \in \langle C_2 \rangle$

```
1  B_new := B;
2  repeat
3  |   B_old := B_new;
4  |   for each pair f_1, f_2 ∈ B_new s.t. f_1 ≠ f_2 and φ(f_1) ≠ 0, φ(f_2) ≠ 0 do
5  |   |   (g_1, g_2) := A(f_1, f_2, φ, ≺);
6  |   |   compute SHADOW_DIV(g_1 f_1 − g_2 f_2, B_new, φ, ≺);
7  |   |   if r ≠ 0 then
8  |   |   |   B_new := B_new ∪ {r}
9  |   |   end
10 |   end
11 until B_new = B_old;
12 C_1 := B_new;
13 B_old := B_new;
14 for each f ∈ B_old do
15 |   compute SHADOW_DIV(f, B_new \ {f}, φ, ≺);
16 |   B_new := B_new \ {f};
17 |   if r ≠ 0 then
18 |   |   B_new := B_new ∪ {r};
19 |   end
20 end
21 C_2 := B_new;
```

Algorithm 5: SHADOW_BÜCH$(B, \phi, \prec)$

Proof. We first consider the computation of C_1.

The algorithm iterates only if we detect that $B_{new} \neq B_{old}$. Let B_i denote the basis after the i-th iteration. So there must have been $f, g \in B_{i-1}$ such that the remainder, r, of division of $g_1 f - g_2 g$ by B_{i-1} is non-zero.

Then from Lemma 3, $in_\prec(r) \notin \langle in_\prec(\phi(B_{i-1})) \rangle$. Thus, $\langle in_\prec(\phi(B_0)) \rangle \subsetneq \langle in_\prec(\phi(B_1)) \rangle \subsetneq \langle in_\prec(\phi(B_2)) \rangle \subsetneq \cdots$. $k[\mathbf{y}]$ is Noetherian hence this chain must be finite and consequently the algorithm must stop after finitely many iterations.

The termination of the second part is obvious. ■

In the first part of the algorithm the remainder r is appended in the successive bases. But $r = g_1 f - g_2 g - \sum_i q_i f_i$ so it is already in the ideal before division, so appending it to the basis does not expand the ideal.

Lemma 5. $\langle B \rangle = \langle C_1 \rangle$.

Lemma 6. *$\phi(C_1)$ is the Gröbner basis of $\langle \phi(B) \rangle$. Further, $\phi(C_2)$ is the reduced Gröbner basis for $\langle \phi(B) \rangle$.*

Proof. It was pointed out after Algorithm 3 that $\phi(g_1 f_1 - g_2 f_2) = S(\phi(f_1), \phi(f_2))$. Upon termination, $\phi(r) = 0$ for all $f_1, f_2 \in B_{new}$. So from Lemma 3 the standard expression for the S-polynomial is $S(\phi(f_1), \phi(f_2)) = \sum_j \phi(q_j)\phi(f_j)$ for all pairs $\phi(f_1), \phi(f_2) \in \phi(B_{new})$. From Büchberger's criterion $\phi(B_{new})$ is a Gröbner basis, see [6] section 7 of chapter 2.

In the second part $\phi(r)$ is the result of the reduction of $\phi(f)$ by $\phi(B_{new})\backslash\{f\})$. So upon termination, no polynomial in $\phi(B_{new})$ is reducible by the rest of the polynomials. Thus $\phi(B_{new})$ is the reduced Gröbner basis. ■

Lemma 7. *For every $f \in \langle B \rangle$ there exists $h \in \phi^{-1}(1)$ such that $h \cdot f \in \langle C_2 \rangle$.*

Proof. In the second part of SHADOW_BÜCH, let the successive bases after each reduction be $C_1 = B_0, B_1, \cdots, B_k = C_2$ such that $B_{i+1} = (B_i \cup \{r\}) \setminus \{f\}$ where r is the result of reduction of $f \in B_i$ by $B_i \setminus \{f\}$. We will show that for each j, if $g \in B_0$, then there exists $h \in \phi^{-1}(1)$such that $h \cdot g \in B_j$.

The claim is trivially true for $j = 1$. In order to prove the claim by induction let us assume that it holds for $j \leq i$.

We have $B_i \setminus \{f\} = B_{i+1} \setminus \{r\}$. From SHADOW_DIV algorithm we know that for some constant c there exists $\bar{f} \in \phi^{-1}(c)$ such that $\bar{f} \cdot f \in B_{i+1}$. Let $g \in \langle B_0 \rangle$ then from induction hypothesis $\exists \bar{g} \in \phi^{-1}(1)$ such that $\bar{g} \cdot g \in \langle B_i \rangle$. Thus $\bar{g} \cdot g = \sum_k q_k f_k$ where $f_k \in B_i$. If f does not occur in the sum then $\bar{g} \cdot g \in \langle B_{i+1} \rangle$. Otherwise, suppose $f_1 = f$. So $(1/c)\bar{f} \cdot \bar{g} \cdot g = (1/c) \sum_k q_k (\bar{f} f_k) \in B_{i+1}$. The desired $h = (1/c)\bar{f} \cdot \bar{g}$. ■

Remark If we assume that the computation of ϕ and ϕ^{-1} of a polynomial takes constant time, then the time complexity of SHADOW_BÜCH is same as that of Büchberger's algorithm on input $\phi(B)$.

5.3 Projection homomorphism and binomial ideal

From now onwards we shall restrict our consideration to only projection homomorphisms. In the following we shall use z to denote those x-variables which are set to 1 by the projection homomorphism and the remaining variables will be denoted by symbol y. For example, if we are considering $\phi = \Pi_i$ then $x_1, \cdots, x_i$ will be denoted by $z_1, \cdots, z_i$ and $x_{i+1}, \cdots, x_n$ will be denoted by $y_1, \cdots, y_{n-i}$ The steps computing $\phi^{-1}()$ in $\mathcal{A}$ (algorithm 3) and SHADOW_DIV (algorithm 5) are described as follows.

In algorithm $\mathcal{A}$ for $j = 1$ and 2, f_j must contain a sub-polynomial of the form $h_j(z).y^{\alpha_j}$ such that $\phi(h_j(z)) = c_j$ and $in_\prec(\phi(f'_j))$ is strictly less than $in_\prec(\phi(f))$ where $f'_j = f_j - h_j(z).y^{\alpha_j}$. We define steps 4 and 5 as $g_1 := h_2(z)y^{\beta_1}$ and $g_2 := h_1(z)y^{\beta_2}$.

In algorithm *SHADOW_DIV* there exists a sub-polynomial $h(z)y^\alpha$ in $p(x)$ such that $in_\prec(\phi(p - h(z)y^\alpha))$ is strictly less than $in_\prec(\phi(p))$. We define step 10 as $g_1 := h(z)y^\alpha$.

Observation 5 *If ϕ is a projection homomorphism and f_1, f_2 are homogeneous w.r.t. $\boldsymbol{d}$ and $(g_1, g_2) = \mathcal{A}(f_1, f_2, \phi, \prec)$, then $g_1 f_1 - g_2 f_2$ is also homogeneous w.r.t. $\boldsymbol{d}$.*

We further restrict our discussion to binomial ideals. If a binomial $f = x^{\alpha_1} - x^{\alpha_2}$ is such that $\phi(f)$ is non-zero, then $\phi(x^{\alpha_1}) \neq \phi(x^{\alpha_2})$. Hence g_1, g_2 in steps 4,5 in $\mathcal{A}$ and g_1 in step 10 in SHADOW_BÜCH are all monomials.

Observation 6 *If ϕ is a projection, f_1 and f_2 are binomials and $(g_1, g_2) = \mathcal{A}(f_1, f_2, \phi, \prec)$, then $f_1 g_1 - f_2 g_2$ is the S-polynomial of f_1, f_2, hence it is also a binomial.*

Observation 7 *If ϕ is a projection and B is a set of binomials, then $\bar{f}$ computed by $SHADOW\_DIV(f, B, \phi, \prec)$ is a monomial. Additionally, if f and each member of B is homogeneous, then so is the remainder r.*

In the notation for the variables in this section, $\bar{f}$ computed by SHADOW_DIV is z^α for some α. Using this fact in the proof of Lemma 7 we get the following lemma which is at the heart of algorithm proposed in the next section.

Lemma 8. *If ϕ is a projection, B is a set of binomials, and C_2 is computed by SHADOW_*
$BÜCH(B, \phi, \prec)$, then for each binomial $f \in \langle B \rangle$ there exists a monomial z^α such that $z^\alpha f \in \langle C_2 \rangle$.

6 A fast algorithm for toric ideals

Let B be a finite set of binomials from $k[\mathbf{x}]$. In Algorithm 6 we present a new algorithm to compute $\langle B \rangle : (x_1 \dots x_n)^\infty$ which is the key step in the computation of (generator set of) a toric ideal I_A given a matrix A.

Data: $B \subset k[\mathbf{x}]$, a finite set of binomials
Result: A generating set of $\langle B \rangle : (x_1 \dots x_n)^\infty$
1 **for** $i = n - 1$ *to* 0 **do**
2 $\quad \boldsymbol{d} := (0_1, \dots, 0_i, 1_{i+1}, \dots, 1_n, 1_y)$;
3 $\quad \tilde{B} = \{\tilde{f} | f \in B\}$;
4 $\quad \tilde{C_2} :=$ SHADOW_BÜCH$(\tilde{B}, \Pi_i, \prec_{\boldsymbol{d}, i+1})$;
5 $\quad B := \pi_y(\tilde{C_2} \div x_{i+1}^\infty)$;
6 **end**
7 **return** B;
Algorithm 6: Computation of $I : (x_1 \cdots x_n)^\infty$ for a binomial ideal I

To prove the correctness of Algorithm 6 let us assume that at the start of an iteration the value of B is B_{old} and at its end it is B_{new}. From Lemma 8 $\langle \tilde{C_2} \rangle : (x_1 \cdots x_i)^\infty = \langle \tilde{B} \rangle : (x_1 \cdots x_i)^\infty$ and from Lemma 6 $\Pi_i(\tilde{C_2})$ is the reduced Gröbner basis of $\langle \Pi(\tilde{B}) \rangle$. From Theorem 3 $\langle \tilde{C_2} \div x_{i+1}^\infty \rangle = \langle \tilde{C_2} \rangle : x_{i+1}^\infty$. Hence $\langle \tilde{C_2} \div x_{i+1}^\infty \rangle : (x_1 \cdots x_i)^\infty = (\langle \tilde{C_2} \rangle : (x_1 \cdots x_i)^\infty) : x_{i+1}^\infty = (\langle \tilde{B} \rangle : (x_1 \cdots x_i)^\infty) : x_{i+1}^\infty = (\langle \tilde{B} \rangle : (x_1 \cdots x_{i+1})^\infty)$. Taking the projection π_y we get $\langle B_{new} \rangle : (x_1 \cdots x_i)^\infty = \langle B_{old} \rangle : (x_1 \cdots x_{i+1})^\infty$. Thus we have the correctness theorem.

Theorem 8. *Algorithm 6 correctly computes* $\langle B \rangle : (x_1 \cdots x_n)^\infty$.

The advantage of the new algorithm as follows. In this algorithm the dimension of the y-space is 1 in the first iteration, 2 in the second iteration, so on. Symbolically let $t(i)$ denote the time complexity of the Büchberger's algorithm in i variable problem. Then, as remarked after Lemma 7, the cost of the proposed algorithm is $\sum_{i=1}^{n} t(i)$ against the Sturmfels' algorithm's cost $n \cdot t(n)$.

7 Experimental Results

In this section we present the results of performance of the new algorithm and compare it with the existing algorithm by Sturmfels [14]. Due to time constraint we could not perform comparative study with *project and lift* by Hemmecke and Malkin [9].

In these experiments we randomly generate binomials and compute $J_V : (x_1 \ldots x_n)^\infty$. The programs were written in C. There are cases where one can ignore certain S-polynomial reduction in the Büchberger algorithm for Gröbner basis computation. There is a significant literature on criteria to select such S-polynomials. We only applied one such criterion, referred as *criterion tail* in Proposition 3.15 of [15] in the implementation of the new algorithm as well as to Sturmfels algorithm. Since every such criterion can be applied to both algorithms, we believe the performance gains shown here will remain same after the implementations are fully optimized.

Table 1 shows performances of the two algorithms. Although only a few cases are shown in the table we ran an extensive experiment and in each and every case, unless the overall time was very very small, the proposed algorithm was faster. Also, as expected the performance ratio improves as the number of variables increase.

Table 1.

Number of variables	Size of basis		Time taken (in sec.)		Speedup
	Initial	Final	Sturmfels	Proposed	
6	2	5	0.0	0.0	-
	4	51	0.001	0.0	-
8	4	186	0.20	0.02	10
	6	597	10.0	1.17	8.6
10	6	729	27.70	0.74	37.4
	8	357	5.48	0.87	6.3
12	6	423	5.80	0.32	18.1
	8	2695	1147.07	32.98	34.8
14	10	751	601.0	20.52	29.3

We observed an unanticipated downside of the proposed algorithm. In this algorithm reduction ignores the invisible variables (lower indexed). During the reduction (recall $\bar{f}$ in SHADOW_DIV) the exponents of these variables are found to grow to significantly large values. This may lead to numbers exceeding the representation range of the machine in large problems.

References

1. Conti, P., Traverso, C.: Buchberger algorithm and integer programming. In: AAECC. (1991) 130–139
2. Urbaniak, R., Weismantel, R., Ziegler, G.M.: A variant of the buchberger algorithm for integer programming. SIAM J. Discret. Math. **10** (1997) 96–108
3. Thomas, R.R., Weismantel, R.: Truncated grbner bases for integer programming. In: in Engineering, Communication and Computing, Kluwer Academic Publishers (1995) 241–257
4. Sturmfels, B.: Grobner Bases and Convex Polytopes. Volume 8 of University Lecture Series. American Mathematical Society (1995)
5. Tayur, S.R., Thomas, R.R., Natraj, N.R.: An algebraic geometry algorithm for scheduling in the presence of setups and correlated demands. Mathematical Programming **69** (1995) 369–401
6. Cox, D.A., Little, J., O'Shea, D.: Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra, 3/e (Undergraduate Texts in Mathematics). Springer-Verlag New York, Inc., Secaucus, NJ, USA (2007)
7. Biase, F.D., Urbanke, R.: An algorithm to calculate the kernel of certain polynomial ring homomorphisms. Experimental Mathematics **4** (1995) 227–234
8. Bigatti, A.M., Scala, R., Robbiano, L.: Computing toric ideals. J. Symb. Comput. **27** (1999) 351–365
9. Hemmecke, R., Malkin, P.N.: Computing generating sets of lattice ideals and markov bases of lattices. Journal of Symbolic Computation **44** (2009) 1463 – 1476
10. Bayer, D., Stillman, M.: On the complexity of computing syzygies. j-J-SYMBOLIC-COMP **6** (1988) 135–148 (or 135–147??) Computational aspects of commutative algebra.
11. Bayer, D., Stillman, M.: A theorem on refining division orders by the reverse lexicographic order. Duke Mathematical Journal **55** (1987) 321–328
12. Adams, W.W., Loustaunau, P.: An Introduction to Gröbner Bases. Volume 3 of Graduate Studies in Mathematics. American Mathematical Society, Rhode Island (1994)
13. Buchberger, B.: A theoretical basis for the reduction of polynomials to canonical forms. SIGSAM Bull. **10** (1976) 19–29
14. Hosten, S., Sturmfels, B.: Grin: An implementation of grbner bases for integer programming (1995)
15. Bigatti, A.M., Scala, R., Robbiano, L.: Computing toric ideals. J. Symb. Comput. **27** (1999) 351–365