Master-thesis
Department of Mathematical Sciences
University of Aarhus
Denmark

June 2002

Computing Gröbner fans of toric ideals

Anders Nedergaard Jensen

Advisor: Niels Lauritzen

Abstract

This thesis is about the toric ideals in a polynomial ring over a field. Basic properties of toric ideals and methods for computing Gröbner bases for them are presented. This includes the saturating Buchberger algorithm taking advantage of the structure of toric ideals. Associated to a positive-homogeneous ideal is its state polytope and its Gröbner fan. Graph traversal algorithms for the edge graph of the state polytope of a toric ideal are given and implemented. Using these we can compute all reduced Gröbner bases of a toric ideal and thereby also the Gröbner fan. Combining various ideas the final program becomes faster than the original one by Huber and Thomas allowing us to compute even larger Gröbner fans than possible before.

Contents

1 Toric ideals **6**
- 1.1 The group ring . . . 6
- 1.2 The polynomial ring and the Laurent polynomial ring . . . 7
- 1.3 Toric ideals . . . 8
- 1.4 Prime ideals generated by binomials . . . 10

2 Computing generators for a toric ideal **13**
- 2.1 Computing generators for a toric ideal . . . 13
- 2.2 Representing binomials . . . 14
- 2.3 The saturating division algorithm . . . 14
- 2.4 The saturating Buchberger algorithm . . . 18

3 Saturating ideals **21**
- 3.1 Homogeneous ideals . . . 21
- 3.2 Saturating ideals . . . 22

4 Lattices **26**
- 4.1 Lattices . . . 26
- 4.2 A reduced basis in $\mathbb{R}^2$ . . . 26
- 4.3 A reduced basis in $\mathbb{R}^n$ . . . 27
- 4.4 Lenstra Lenstra Lovász algorithm . . . 28
- 4.5 Dependent integer vectors . . . 31

5 Convex polyhedral sets **36**
- 5.1 Convex polyhedral sets . . . 36
- 5.2 Faces . . . 36
- 5.3 Normal fans . . . 38

6 The Gröbner fan **40**
- 6.1 Lemmas in Sturmfels' . . . 40
- 6.2 Initial ideals . . . 41
- 6.3 Homogeneous ideals . . . 43
- 6.4 Reduced Gröbner bases and monomial initial ideals . . . 43
- 6.5 The Gröbner fan . . . 45
- 6.6 Facets of toric Gröbner cones . . . 47
- 6.7 The state polytope . . . 50

7 Computing the Gröbner fan **52**
- 7.1 Algorithm flip . . . 52
  - 7.1.1 Implementation details . . . 53
- 7.2 Finding the facets of a Gröbner cone . . . 54
- 7.3 Traversing the graph . . . 55

7.4 The toric Gröbner walk 56
7.5 The reverse search tree method 59
7.6 Example 61

8 Computational experience **63**
8.1 Timing examples 63
8.2 Computing a single reduced Gröbner basis 65
8.3 Computing all reduced Gröbner bases 67
8.4 Compared to TiGERS 68
8.5 Where is the time spent? 69
8.6 Reliability 70

9 Postscript **71**

A Sources **72**

B Notation **73**

C A tiny user manual **74**

Preface

My interest has always been the interplay between mathematics and computer science so Gröbner bases were obviously a possible subject for my master thesis. Taking a course on Gröbner bases some years ago my fellow student Jesper Petersen and I implemented Buchberger's algorithm for the polynomial ring over the field $\mathbb{Q}$. An unpleasant property of this field is that the fraction representation of the polynomial coefficients usually grow very large as the Gröbner basis computation proceeds. The first part of this master thesis is about the so-called toric ideals whose reduced Gröbner bases can be computed without doing non-trivial calculations in the field. Toric ideals are interesting for the reason among others that they can be used for solving certain combinatorial problems. Among these is the integer programming problem known to be NP-complete (see [Sturmfels] and [Papadimitriou 1994] for details). For that reason we cannot hope for a good theoretical time complexity in general nor is a complexity analysis a subject of this thesis.

For the second part of the thesis my advisor Niels Lauritzen suggested the paper "Computing Gröbner Fans of Toric Ideals" by Birkett Huber and Rekha Thomas. My task has been to understand this paper and the underlying theory, fill out some gaps and implement its algorithms. The idea is to start with one reduced Gröbner basis and vary the term order to get an other. In this way we compute all reduced Gröbner bases of a toric ideal and thereby we also compute the so-called Gröbner fan of the ideal. Again the properties of toric ideals make the computations easier. The algorithms presented have all been implemented giving a program considerably faster than the original one by [Huber,..] allowing us to compute Gröbner fans with more elements than before. Experiments are presented in the last section.

The reader should be familiar with basic Gröbner basis theory. For students in Århus this knowledge can be obtained by reading the "Algebra 1" course notes by Niels Lauritzen. Other people could read [Cox, Little and O'Shea]. Appendix A contains a list of sources for the various sections and appendix B describes the notation used. At some points I have left out central proofs. These are the proofs of theorems from polyhedral set theory connecting the Gröbner fan and the state polytope of an ideal. Proving these theorems in detail could easily add another 15 pages to the thesis. That is why they were left out. Proofs for these theorems are given in [Sturmfels].

1 Toric ideals

In this section we will define the toric ideals using the group ring construction of the polynomial rings over a field. Some basic properties of the toric ideals will be proved. To become more familiar with toric ideals we will end this section by proving that the toric ideals actually are the prime ideals generated by binomials (see definition 1.10).

1.1 The group ring

Let k be a field and G a monoid. The group ring is an algebra $k[G]$ consisting of all functions $f : G \to k$ with finite support ($f(a) \neq 0$ for only finitely many $a \in G$). The addition is given by addition in k:

$$(f + g)(\sigma) = f(\sigma) + g(\sigma)$$

for $f, g \in k[G]$ and $\sigma \in G$. The zero function is the zero element in $k[G]$ and the product of two elements $f, g \in k[G]$ is given by the convolution:

$$(g * f)(\sigma) = \sum_{\lambda+\mu=\sigma} f(\lambda) * g(\mu)$$

where $\sigma \in G$ and (λ, μ) runs through $G \times G$ so that $\lambda + \mu = \sigma$. Since f and g have finite support, the sum is finite. The new function $(g * f)$ has finite support and is therefore an element in $k[G]$. The 1-element is the map mapping zero to 1 and the rest of G to zero. Scaling an element $f \in k[G]$ by an element $a \in k$ is done by:

$$(af)(\sigma) = a(f(\sigma))$$

for $\sigma \in G$.

Proposition 1.1 *Let k be a field and G a monoid. $k[G]$ is an algebra over k.*

The proof is left to the reader.

Let G and H be monoids and let $A : G \to H$ be a monoid homomorphism. A induces an algebra homomorphism $\pi_A : k[G] \to k[H]$ by

$$\pi_A(f)(\sigma) = \sum_{A\lambda=\sigma} f(\lambda)$$

for $f \in k[G], \sigma \in H$ and λ running through G so that $A\lambda = \sigma$. $\pi_A(f)$ has finite support.

Proposition 1.2 *π_A is an algebra homomorphism.*

Proof. Let $\pi = \pi_A$. We check the equalities:

$$\pi(f+g)(\sigma) = \sum_{A\lambda=\sigma} (f+g)(\lambda) = \sum_{A\lambda=\sigma} f(\lambda) + \sum_{A\lambda=\sigma} g(\lambda) = \pi(f)(\sigma) + \pi(g)(\sigma)$$

$$= (\pi(f) + \pi(g))(\sigma)$$

$$\pi(f*g)(\sigma) = \sum_{A\lambda=\sigma} (f*g)(\lambda) = \sum_{A\lambda=\sigma} \sum_{\alpha+\beta=\lambda} f(\alpha)*g(\beta) = \sum_{A(\alpha+\beta)=\sigma} f(\alpha)*g(\beta)$$

$$= \sum_{a+b=\sigma} \sum_{A\alpha=a, A\beta=b} f(\alpha)*g(\beta)$$

$$= \sum_{a+b=\sigma} (\sum_{A\alpha=a} f(\alpha)) * (\sum_{A\beta=b} g(\beta)) = \sum_{a+b=\sigma} \pi(f)(a) * \pi(g)(b) = (\pi(f)*\pi(g))(\sigma)$$

$$\pi(c*f)(\sigma) = \sum_{A\lambda=\sigma} (c*f)(\lambda) = \sum_{A\lambda=\sigma} c*(f(\lambda)) = c*\sum_{A\lambda=\sigma} (f(\lambda)) = c*(\pi(f)(\sigma))$$

$$\pi(1)(\sigma) = 1(\sigma)$$

□

1.2 The polynomial ring and the Laurent polynomial ring

The group ring construction gives us an alternative way of constructing the polynomial ring over a field k in several variables. Let $n \in \mathbb{N}$. $\mathbb{N}^n$ is a monoid by vector addition. $k[\mathbb{N}^n]$ is isomorphic to $k[\mathbf{x}] := k[x_1, \ldots, x_n]$. One isomorphism is given by: x_i maps to the map taking the ith standard basis vector to 1 and the rest of $\mathbb{N}^n$ to zero for $i = 1, \ldots, n$. It is left to the reader to show that this is in fact an isomorphism. We will soon forget that $k[\mathbf{x}]$ and $k[\mathbb{N}^n]$ are two different things and identify $k[\mathbf{x}]$ with $k[\mathbb{N}^n]$ using the isomorphism above.

Let $d \in \mathbb{N}$. The Laurent polynomial ring is a polynomial ring where we allow the exponents to be negative. We denote the Laurent polynomial ring in d variables by $k[\mathbf{t}^{\pm 1}]$. To be more precise we identify $k[\mathbf{t}^{\pm 1}]$ with $k[\mathbb{Z}^d]$.

Lemma 1.3 *$k[\mathbb{N}^n]$ and $k[\mathbb{Z}^d]$ are domains.*

Proof. $k[\mathbb{N}^n] = k[\mathbf{x}]$ is already known to be a domain. To show that $k[\mathbb{Z}^d]$ is a domain let $a, b \in k[\mathbb{Z}^d]$ and $ab = 0$. Multiply this element by a monomial $c \in k[\mathbb{N}^d]$ so that $ca, cb \in k[\mathbb{N}^d]$. $(ca)(cb) = 0$ and we must have either $ca = 0$ or $cb = 0$. In both cases we conclude that either $a = 0$ or $b = 0$. □

1.3 Toric ideals

Let $d, n \in \mathbb{N}$ and $A \in mat_{dn}(\mathbb{Z})$ with columns $a_1, \ldots, a_n \in \mathbb{Z}^d$. A defines a monoid homomorphism in the usual way: $A : \mathbb{N}^n \to \mathbb{Z}^d$, thereby inducing an algebra homomorphism $\pi := \pi_A : k[\mathbb{N}^n] \to k[\mathbb{Z}^d]$.

Remark 1.4 π maps x_i to t^{a_i}. Since π is a homomorphism this defines $\pi(f)$ for all $f \in k[\mathbf{x}]$. Especially, $\pi(x^v) = t^{Av}$ for $v \in \mathbb{N}^n$. Another way of saying this is that π does the substitution $\pi(f) = f(t^{a_1}, \ldots, t^{a_n})$ and cancel terms as allowed in $k[\mathbf{t}^{\pm 1}]$.

The kernel of π is an ideal and we denote it I_A.

Definition 1.5 *An ideal of this form is called a toric ideal.*

We will assume from now that the field k has characteristic $char(k) \neq 2$. This will not be important right now but it will be in the sections to come. Especially it is important in theorem 6.17.

Lemma 1.6 *A toric ideal is a prime ideal.*

Proof. Let $\alpha\beta \in I_A$. $\pi(\alpha)\pi(\beta) = \pi(\alpha\beta) = 0$ implying that $\pi(\alpha) = 0 \vee \pi(\beta) = 0$ since the Laurent polynomial ring is a domain (lemma 1.3). α or β must be in I_A. □

Lemma 1.7 *A binomial $x^u - x^v \in I_A$ if and only if $Au = Av$ where $u, v \in \mathbb{N}^n$*

Proof. If $u, v \in \mathbb{N}^n$ satisfy $Au = Av$ then clearly $\pi(x^u - x^v) = \pi(x^u) - \pi(x^v) = t^{Au} - t^{Av} = 0$. On the other hand if $\pi(x^u - x^v) = \pi(x^u) - \pi(x^v) = t^{Au} - t^{Av} = 0$ the two terms cancel, implying $Au = Av$. □

Proposition 1.8 *As a k-vector space I_A is spanned by*

$$\{x^u - x^v | u, v \in \mathbb{N}^n \wedge Au = Av\}.$$

Proof. By lemma 1.7 these binomials are in I_A. Let $\prec$ be a term order. Suppose that I_A is not spanned by the binomials. We may choose a polynomial $p \in I$ which is not in the span such that $in_\prec(p) = 1x^u$ is minimal with respect to $\prec$. $p \in \ker(\pi)$. Hence, x^u cancel with at least one other term ax^v when substituting using π, $a \in k$. This term is less than x^u with respect to $\prec$ since it is not initial in p. $\pi(x^u) = \pi(x^v)$. Define $q = x^u - x^v$. $\pi(q) = 0$. $\pi(p - q) = 0$ and $p \neq q$. $in_\prec(p-q) \prec in_\prec(p)$. Hence, by assumption it is possible to write $p-q$ as a linear combination of the binomials. $p = p - q + x^u - x^v$ can be written as a linear combination as well and this is a contradiction. □

Corollary 1.9 *I_A is generated by finitely many binomials.*

Proof. The set in proposition 1.8 spans I_A. Hilbert's basis theorem tells us that I_A is generated by finitely many polynomials. Each of these polynomials is a linear combination of finitely many binomials from the set. By taking all these binomials we have a finite set generating I_A. □

It is useful to redefine the term "binomial":

Definition 1.10 *By a binomial we mean a monomial difference $x^u - x^v$ where $u, v \in \mathbb{N}^n$. A pure binomial is a binomial where x^u and x^v have no common factor, that is, for all i: $v_i = 0 \vee u_i = 0$.*

By this definition $0 = 1 - 1$ is a pure binomial. For a vector $u \in \mathbb{Z}^n$ we introduce the notation p_u for the pure binomial $x^{u^+} - x^{u^-}$.

Corollary 1.11 *A reduced Gröbner basis for a toric ideal I_A with respect to a term order $\prec$ consists of binomials.*

Proof. We observe that reducing a binomial by a set of binomials using the division algorithm gives a new binomial (may be zero). The S-polynomial of two binomials is a binomial (may be zero). Hence, starting with the set of binomials in corollary 1.9, using Buchberger's Algorithm we produce a reduced Gröbner basis consisting only of binomials. □

Remark 1.12 It is worth noticing that the binomials in a reduced Gröbner basis for a toric ideal are pure: Suppose that $x^\alpha - x^\beta$ was an element in the reduced Gröbner basis of I_A with respect to a term order $\prec$ and that the terms in $x^\alpha - x^\beta$ had a common factor x_i. $x^\alpha - x^\beta = x_i(\frac{x^\alpha}{x_i} - \frac{x^\beta}{x_i}) \in I_A$. Since x_i maps to t^{a_i} by π_A, $x_i \notin I_A$. I_A is prime implying $\frac{x^\alpha}{x_i} - \frac{x^\beta}{x_i} \in I_A$. The initial term of $\frac{x^\alpha}{x_i} - \frac{x^\beta}{x_i}$ with respect to $\prec$ belongs to $in_\prec(I_A)$ and divides x^α or x^β non-trivially, contradicting that $x^\alpha - x^\beta$ is an element in the reduced Gröbner basis of I_A with respect to $\prec$.

We will use the notation $\mathcal{G}_\prec(I)$ to denote the reduced Gröbner basis of I with respect to $\prec$. An example of reduced Gröbner bases for a toric ideal is given in section 7.6.

We have proved that a toric ideal is prime and generated by binomials. The converse is also true:

Proposition 1.13 *A prime ideal in $k[\mathbf{x}]$ generated by binomials is toric.*

Proof. See section 1.4. □

Example 1.14 Not all binomial ideals are toric. The ideal $I := \langle x^2 - y^2 \rangle$ is not prime since $(x - y)(x + y) \in I$ but $x - y, x + y \notin I$.

1.4 Prime ideals generated by binomials

In this section it will be proven that a prime ideal generated by binomials is toric. First we do some observations:

Proposition 1.15 *Let $I \subset k[\mathbf{x}]$ be an ideal generated by binomials. For a polynomial $p = \sum_{v \in \mathbb{N}^n} a_v x^v \in I$ with $a_v \in k$ the sum of its coefficients $\sum_{v \in \mathbb{N}^n} a_v$ is zero.*

Corollary 1.16 *An ideal $I \subset k[\mathbf{x}]$ generated by binomials contains no monomials.*

Corollary 1.17 *If $I \subset k[\mathbf{x}]$ is a prime ideal generated by binomials then I is generated by pure binomials.*

Lemma 1.18 *Let $I \subset k[\mathbf{x}]$ be an ideal and $v \in \mathbb{Z}^n$. If $p_v \in I$ then $p_{-v} \in I$.*

Proof. $p_{-v} = -p_v \in I$. □

Lemma 1.19 *Let $I \subset k[\mathbf{x}]$ be a prime ideal generated by binomials and let $u, v \in \mathbb{Z}^n$. If $p_u, p_v \in I$ then $p_{v-u} \in I$.*

Proof. In section 2.4 a little computation will prove that

$$\frac{x^{u^+ \vee v^+}}{x^{v^+}} p_v - \frac{x^{u^+ \vee v^+}}{x^{u^+}} p_u = x^w p_{v-u}$$

for some $w \in \mathbb{N}^n$. x^w cannot be in I according to corollary 1.16. The left hand side is in I and I is prime. Hence, p_{v-u} is in I. □

Lemma 1.20 *Let $I \subset k[\mathbf{x}]$ be a prime ideal generated by binomials, let $u, v \in \mathbb{Z}^n$ such that $p_v, p_u \in I$ and let $c \in \mathbb{N}$. The binomials p_{v+u} and p_{cu} are in I.*

Proof. Follows from the two lemmas above by induction. □

The hard part is the following:

Lemma 1.21 *Let $I \subset k[\mathbf{x}]$ be a prime ideal generated by binomials. If $u \in \mathbb{Z}^n$ such that $p_{cu} \in I$ for some $c \in \mathbb{N} \backslash \{0\}$ then $p_u \in I$.*

Proof.

$$p_{cu} = x^{cu^+} - x^{cu^-} = (x^{u^+} - x^{u^-})(x^{(c-1)u^+} + x^{(c-2)u^+} x^{u^-} + \ldots + x^{(c-1)u^-}) \in I$$

If $char(k) = 0$ the proof is simple: The second factor is not in I by proposition 1.15. Since I is prime, the first factor must be in I. $p_u = x^{u^+} - x^{u^-} \in I$.

In the general case the proof is harder. Let $b \in \mathbb{N}\backslash\{0\}$ be the smallest number such that $p_{bu} \in I$. Suppose $b \geq 2$. We write:

$$p_{bu} = x^{bu^+} - x^{bu^-} = (x^{u^+} - x^{u^-})(x^{(b-1)u^+} + x^{(b-2)u^+}x^{u^-} + \ldots + x^{(b-1)u^-}) \in I$$

with b terms in the second factor of the product. By the assumption the first factor cannot be in I. Since I is prime the second factor has to be in I. Using the arguments in proposition 1.11 and remark 1.12 we see that a reduced Gröbner basis of I consists of pure binomials. The second factor reduces to 0 modulo a reduced Gröbner basis $\mathcal{G}$ of I. This implies that at least two monomials $x^{au^+ + (b-1-a)u^-}$ and $x^{a'u^+ + (b-1-a')u^-}$ reduce to the same monomial by a sequence of reductions with respect to binomials $p_{\gamma_1}, \ldots, p_{\gamma_e}$ and $p_{\gamma'_1}, \ldots, p_{\gamma'_{e'}}$ in $\mathcal{G}$ (making it possible for terms to cancel). $a, a', e, e' \in \mathbb{N}$. We may assume that $a > a'$ and since there are b terms in the second factor $a - a' < b$. From the division algorithm we get the equality: $x^{au^+ + (b-1-a)u^- - \gamma_1 - \ldots - \gamma_e} = x^{a'u^+ + (b-1-a')u^- - \gamma'_1 - \ldots - \gamma'_{e'}} \Rightarrow$ $w := au^+ + (b-1-a)u^- - a'u^+ - (b-1-a')u^- = \gamma_1 + \ldots + \gamma_e - \gamma'_1 - \ldots - \gamma'_{e'}$. Using lemma 1.20 $p_w \in I$. $w = (a-a')u^+ - (a-a')u^- = (a-a')u$ contradicting that b was the smallest number such that $p_{bu} \in I$. Hence, $b = 1$. □

Proposition 1.22 *Let $I \subset k[\mathbf{x}]$ be a prime ideal generated by binomials and let $u_1, \ldots, u_k \in \mathbb{Z}^n$ such that $p_{u_i} \in I$ for all i. Let $a_1, \ldots, a_k \in \mathbb{Q}$. If $\sum_i a_i u_i \in \mathbb{Z}^n$ then $p_{\sum_i a_i u_i} \in I$.*

Proof. Let $a \in \mathbb{Z}$ be the product of the denominators of $a_1, \ldots, a_k$. The linear combination $v := a\sum_i a_i u_i$ is an integer linear combination of $u_1, \ldots, u_k$, so by lemma 1.20 $p_v \in I$. Since I is prime and generated by binomials and $\frac{1}{a}v \in \mathbb{Z}^n$ lemma 1.21 tells us that $p_{\sum_i a_i u_i} = p_{\frac{1}{a}v} \in I$. □

We now return to the proof of proposition 1.13.

Proof. Let $I \subset k[\mathbf{x}]$ be the prime ideal generated by binomials. According to corollary 1.17 we may assume that these binomials are pure. Using Hilbert's basis theorem as we did in corollary 1.9 we get that I is generated by finitely many pure binomials. Denote them by $p_{v_1}, \ldots, p_{v_k}$ where $v_i \in \mathbb{Z}^n$ for all i. We wish to construct a homomorphism $\pi : k[\mathbf{x}] \to k[\mathbf{t}^{\pm 1}]$ with $ker(\pi) = I$. Furthermore, π must be induced by a semigroup homomorphism $A : \mathbb{N}^n \to \mathbb{Z}^d$ for some $d \in \mathbb{N}$. A extends to the linear map $A : \mathbb{Q}^n \to \mathbb{Q}^d$. Since p_{v_i} must be in $ker(\pi)$, v_i must be in $ker(A)$ for all i.

Construction: Select an $l \in \mathbb{N}$ and a subset $\{u_1, \ldots, u_l\}$ of $\{v_1, \ldots, v_k\}$ so that $\{u_1, \ldots, u_l\}$ is a $\mathbb{Q}$-basis for $S = span_{\mathbb{Q}}(v_1, \ldots, v_k)$. Let $d := n - l$. Extend the basis to a basis $\{u_1, \ldots, u_{l+d}\}$ for $\mathbb{Q}^n$. We now define the linear map $A' : \mathbb{Q}^n \to \mathbb{Q}^d$ by:

$$u_i \mapsto 0 \text{ for } i \in \{1, \ldots, l\}$$

$$u_{l+i} \mapsto e_i \text{ for } i \in \{1, \ldots, d\}$$

where $\{e_1, \ldots, e_d\}$ is the standard basis of $\mathbb{Q}^d$. A' is given by some matrix $A' \in mat_{dn}(\mathbb{Q})$. Instead of looking at A' we look at the matrix $A \in mat_{dn}(\mathbb{Z})$ defined by $A := sA'$ where $s \in \mathbb{Z}$ is the product of all denominators in A'. A defines a map $A : \mathbb{N}^n \to \mathbb{Z}^d$. This map induces an algebra homomorphism $\pi : k[\mathbf{x}] \to k[\mathbf{t}^{\pm 1}]$.

Now, we just have to show that with this construction $ker(\pi) = \langle p_{v_1}, \ldots, p_{v_k} \rangle$.
$\supset$: Clearly, since π is a homomorphism and $v_i \in ker(A)$ implying $p_{v_i} \in ker(\pi)$.
$\subset$: The left hand side is toric and using remark 1.12 we see that $I_A = ker(\pi)$ is generated by pure binomials. By lemma 1.7 it suffices to verify $Aa = 0 \Rightarrow p_a \in I$ for all $a \in \mathbb{Z}^n$. Let $a \in \mathbb{Z}^n$ and $Aa = 0$. $a \in ker(A)$ which is generated by $\{u_1, \ldots, u_l\}$. Hence, $a = \sum_{i=1}^{l} a_i u_i$ for some $a_i \in \mathbb{Q}$. For $i \leq l$ $p_{u_i} \in I$ which is prime and generated by binomials. Using proposition 1.22 we see that $p_a \in I$ and we are done. $\square$

2 Computing generators for a toric ideal

We begin this section by giving a method for computing a generating set for a toric ideal I_A where A is a matrix with non-negative entries. This is done using Buchberger's algorithm. The properties of toric ideals allow us to delete common factors in binomials. Common factors are easily deleted when binomials $x^u - x^v$ are represented by $u - v$. This leads us to "the saturating division algorithm" and "the saturating Buchberger algorithm". In fact, these algorithms do all their computations on vectors. Reduction and S-polynomial computation turn out to be simple vector operations. Another nice property when implementing these algorithms is that we do not have to do any computations in the field k.

2.1 Computing generators for a toric ideal

The algorithm we are about to present is based on the following proposition:

Proposition 2.1 *Let $A \in mat_{dn}(\mathbb{N})$ be a matrix with non-negative entries.*

$$\ker(\pi_A) = J \cap k[\mathbf{x}]$$

where J is the ideal $\langle x_1 - t^{a_1}, \ldots, x_n - t^{a_n} \rangle$ in $k[\mathbf{x}][t_1, \ldots, t_d] = k[\mathbb{N}^{n+d}]$.

Proof.
$\supset$: π_A extends to an algebra homomorphism $\pi_B : k[\mathbf{x}][t_1, \ldots, t_d] \to k[t_1, \ldots, t_d]$ by $t^u x^v \mapsto t^u t^{Av}$ for $u \in \mathbb{N}^d, v \in \mathbb{N}^n$. π_B is induced by the monoid homomorphism $B : \mathbb{N}^{n+d} \to \mathbb{N}^d$ given by $(v, u) \mapsto u + Av$. The generators for J are in $\ker(\pi_B)$, implying $J \cap k[\mathbf{x}] \subset ker(\pi_B) \cap k[\mathbf{x}] = ker(\pi_A)$ since π_A and π_B are equal on $k[\mathbf{x}]$.
$\subset$: Let $p \in \ker(\pi_A)$. $p = \sum_i m_i$ for some monomials $m_i \in k[\mathbf{x}]$. Since $x_i \sim t^{a_i}$ modulo J each of these monomials m_i is equivalent to $\pi_A(m_i) \pmod J$, implying $p = \sum_i m_i \sim \sum_i \pi_A(m_i) = \pi_A(\sum_i m_i) = \pi_A(p) = 0 \pmod J$. Hence, $p \in J$. □

Algorithm 2.2
Input: *A matrix $A \in mat_{dn}(\mathbb{N})$ with non-negative entries.*
Output: *Generators for I_A.*
Introduce the new variables $t_1, \ldots, t_d$. Compute a Gröbner basis G for $J = \langle x_1 - t^{a_1}, \ldots, x_n - t^{a_n} \rangle$ in $k[\mathbf{x}][t_1, \ldots, t_d]$ with respect to an elimination term order with $x_i \prec t_j$. $G \cap k[\mathbf{x}]$ is a generating set for I_A (it is in fact a Gröbner basis).

Proof. The algorithm follows from the proposition above and the well known method for computing elimination ideals [Lauritzen] theorem 6.8.1. □

Remark 2.3 Recall that an elimination term order with $x_i \prec t_j$ means that if some $t_j | m_1$ and $m_1 \prec m_2$ where m_1 and m_2 are monomials then $t_i | m_2$ for some i. The term order mentioned in the algorithm could be the lexicographic order

with $x_1 \prec \ldots \prec x_n \prec t_1 \prec \ldots \prec t_d$. Another possible term order is $\prec$ defined by $y^\alpha \prec y^\beta \Leftrightarrow \alpha_{n+1} + \ldots + \alpha_{n+d} < \beta_{n+1} + \ldots + \beta_{n+d} \vee (\alpha_{n+1} + \ldots + \alpha_{n+d} = \beta_{n+1} + \ldots + \beta_{n+d} \wedge y^\alpha \prec' y^\beta)$ where $\alpha, \beta \in \mathbb{N}^{n+d}$ and $\prec'$ can be any term order on $k[\mathbb{N}^{n+d}]$.

2.2 Representing binomials

Later we will see that some of the algorithms only need to present polynomials which are binomials. Furthermore, each of these binomials is pure and can be represented by a vector $v \in \mathbb{Z}^n : p_v := x^{v^+} - x^{v^-}$. We have already seen in corollary 1.11 that the Buchberger algorithm produces binomials if it is given binomials as input. These binomials are not necessarily pure.

However, if the given ideal I is toric and thereby prime we conclude that $x^{u+w} - x^{v+w} \in I$ if and only if $x^v - x^u \in I$ for $u, v, w \in \mathbb{N}^n$. (By definition $x_i \notin ker(\pi)$, we may factor out one common variable at a time and the conclusion follows.) For a binomial $x^u - x^v$ we introduce the notation $sat(x^u - x^v)$ for the pure (saturated) binomial p_{u-v}. With this notation we rewrite the observation above: $x^u - x^v \in I \Leftrightarrow sat(x^v - x^u) \in I$. Making this factorisation during the computations in the division algorithm and continuing with the binomial factor gives us the saturating division algorithm 2.5 which of course does not produce the same output. Later we will see that in this way we only have to represent pure binomials when computing toric ideals.

Observe that for $v \in \mathbb{Z}^n$ $p_v = -p_{-v}$, so p_v and p_{-v} generate the same ideal. In a sense the sign is not necessary. Given a term order $\prec$ we make the following useful decision: a pair of pure binomials p and $-p$ is represented by the vectors v and $-v \in \mathbb{Z}^n$ in the sense $p = \pm p_{\pm v}$ *but we choose* the vector $u \in \{v, -v\}$ with the property $\pm in_\prec(p) = \pm in_\prec(-p) = x^{u^+}$ as the representative for the pair when computing. Given a term order $\prec$ we say that a pure binomial $p_v \neq 0$ is *ordered* if $in_\prec(p_v) = x^{v^+}$. If p_v is a pure non-zero binomial then either p_v or p_{-v} is ordered. Always using the chosen representative u implies that p_u is ordered.

Example 2.4 $p_{(-1,1)^T} = x_2 - x_1$ is not ordered with respect to the lexicographic order with $x_2 \prec x_1$. We will use the vector $(1,-1)^T$ to represent $\pm(x_2 - x_1)$ when computing since $p_{(1,-1)^T} = x_1 - x_2$ is ordered with respect to the term order.

2.3 The saturating division algorithm

In this subsection we will see a new version of the division algorithm doing all its operations on vectors representing binomials. The changes done in the algorithm are the ones described in the previous section which are:

- Deleting common factors in the polynomial being reduced.
- Forgetting the sign of the polynomial being reduced.

Having made the above decision on the representatives u and v in $\mathbb{Z}^n$ (p_v and p_u are ordered) we observe:

- Testing for divisibility $in_\prec(p_v)|in_\prec(p_u)$ is done by checking that ${v^+}_i \leq {u^+}_i$ for all i.
- Given that the divisibility condition is satisfied reducing p_u by p_v is done by: $p_{u'} := p_u - \frac{in_\prec(p_u)}{in_\prec(p_v)}p_v$. Clearly, this is a binomial, but is it pure? Let's compute the monomial in $p_{v'}$ with positive sign. That is the monomial given by $-\frac{in_\prec(p_u)}{in_\prec(p_v)}tail_\prec(p_v) = -\frac{x^{u^+}}{x^{v^+}}(-x^{v^-}) = x^{u^+-v^++v^-}$. The monomial with negative sign is $-x^{u^-}$. If the two monomials are relatively prime the binomial is pure and a representative is $(u^+ - v^+ + v^-) - u^- = (u^+ - u^-) - (v^+ - v^-) = u - v$ (or $v - u$). If the monomials do have a common divisor then we want to factorise the binomial. Let the greatest common divisor be x^w for some $w \in \mathbb{N}^n$. $x^{u^+-v^++v^-} - x^{v^-} = x^w(x^{u^+-v^++v^--w} - x^{v^--w}) = x^w p_{u-v}$ since $x^{u^+-v^++v^--w}$ and x^{u^--w} are relatively prime. This means that if we want to continue the computation in the division algorithm with the pure binomial factor $\pm p_{u-v}$ it is also sufficient to use $u' = u - v$ or $u' = -(u - v)$ depending on which vector is the right representative according to the term order (making $p_{u'}$ an ordered binomial).

We now present the modified (nondeterministic) division algorithm appearing when using the binomial factor after a reduction and using the observations above. We call it "the saturating division algorithm" because we only do the computations on the pure binomial factors. The reader is encouraged to compare this to the ordinary division algorithm ([Lauritzen] or other) since the ordinary division algorithm is more abstract and involves terms like "remainder" which we are going to use in the upcoming arguments, whereas this description is more "low-level".

Algorithm 2.5 *The saturating division algorithm*
Let $\prec$ be a term order.
Input: *Ordered and pure generators for an ideal $I = \langle p_{b_1}, \ldots, p_{b_k} \rangle$ and an ordered pure binomial p_v. ($b_i \neq 0$ for all i.)*
Output: *A pure binomial p_w*
while($p_v \neq 0$ and there is an i such that $x^{b_i^+}|x^{v^+}$)
{
    $v := v - b_i$;
    if(p_v is not ordered)$v := -v$;
}
if($p_v \neq 0$)
while(there is an i such that $x^{b_i^+}|x^{v^-}$)
{

$v := v + b_i$;
}
$w := v$;

Remark 2.6 To see that this is actually the algorithm arising we observe:

It is only possible to transfer a monomial to the remainder two times. After the first loop we transfer x^{v^+} to the remainder and continue reducing $-x^{v^-}$. Letting u denote v^- we see that reduction by b_i is done by $u := u - b_i$. The new u still has non-negative entries. This means that $-x^{v^-}$ is substituted by $-x^{v^- - b_i}$. That is we want to continue the division algorithm with $-x^{v^- - b_i}$. Before we do that we saturate $-x^{v^- - b_i}$ with the remainder obtained so far: $sat(x^{v^+} - x^{v^- - b_i}) = sat(x^\omega p_{v+b_i}) = p_{v+b_i}$ with $x^w = x^{v^+ \wedge (v^- - b_i)}$. After saturating we continue the reduction of the new $-x^{v^-}$. This explains the line $v := v + b_i$.

The essence of this remark is that in the second loop we allow saturation of the current monomial with the remainder.

Remark 2.7 The algorithm terminates. The first loop terminates because the order of the $in_\prec(p_v)$ decreases with respect to $\prec$ when we reduce and also when we saturate. The same is true for x^{v^-} in the second loop. For the ordinary division algorithm the argument for termination was the same as for this one.

Some important properties of the saturating division algorithm follow immediately:

Proposition 2.8 *If the ideal I in the saturating division algorithm is contained in a toric ideal I_A (for example if $I = I_A$)the output p_w satisfies*

$$p_v \in I_A \Leftrightarrow p_w \in I_A$$

$$\forall i : in_\prec(p_{b_i}) \not| in_\prec(p_w) \quad \text{when} \quad w \neq 0$$

$$\forall i : in_\prec(p_{b_i}) \not| tail_\prec(p_w) \quad \text{when} \quad w \neq 0$$

We will now concentrate on the connection between the saturating division algorithm and the ordinary one.

Lemma 2.9 *If a monomial is transfered to the remainder during the saturating division algorithm the remainder can never be zero again and the binomial cannot reduce to zero in this run of the saturating division algorithm.*

Proof. Assume that we have transfered a monomial to the remainder. At any time let $-x^u$ denote the part of the binomial that we continue reducing and let x^v be the remainder. $(u, v \in \mathbb{N}^n)$. By the time that x^v is transfered to the remainder $x^u \prec x^v$ (sharp) since x^v is the leading term. As in the ordinary division

algorithm every time we reduce by a polynomial $-x^u$ decreases with respect to $\prec$ and therefore still $x^u \prec x^v$ (sharp). In case of saturation we remove a common factor but this does not change the relation $x^u \prec x^v$ (sharp). Especially, when x^u is transfered to the remainder at the end of the computation $x^v - x^u$ is not zero since $x^u \prec x^v$ (sharp). □

Notice, transfers of monomials to the remainder are avoided if and only if the binomial reduces to 0 during the first loop.

Proposition 2.10 *If a binomial reduces to 0 by some run of the saturating division algorithm it also reduces to 0 by some run of the division algorithm.*

Proof. By the lemma during the saturating division algorithm we never transfer a monomial to the remainder. This implies that intermediate computation can be described as follows: Let p_1 be the polynomial we wish to reduce. We define p_{i+1} inductively for $i > 0$. Either p_i is zero or there exists an index a_i such that $in_\prec(p_{b_{a_i}})$ divides $in_\prec(p_i)$. If p_i is not zero define $p'_{i+1} = p_i - x^{v_i} p_{b_{a_i}}$ where $x^{v_i} = in_\prec(p_i)/in_\prec(p_{b_{a_i}})$. Define $p_{i+1} = \pm sat(p'_{i+1}) = \pm x^{-u_i} p'_{i+1}$ for some $u_i \in \mathbb{N}^n$. The "$\pm$" was written because we might have to change the sign to order the binomial. Eventually $p_j = 0$ for some j by the assumption.

The ordinary and the saturating algorithm are both nondeterministic. By making the same choices $a_1, \ldots, a_{j-1}$ in the ordinary algorithm we see that it is possible to reduce the polynomial to zero: Let q_i denote the polynomial at iteration i of the division algorithm. By induction we prove that $q_i = \pm x^{w_i} p_i$ for some $w_i \in \mathbb{N}^n$. Clearly, $x^{w_1} = 1$ satisfies this. Induction step: Assume the claim is true for i. $in_\prec(p_{b_{a_i}})$ divides $in_\prec(p_i)$ and therefore also $in_\prec(q_i)$. We let

$$q_{i+1} = q_i \pm x^{w_i} x^{v_i} p_{b_{a_i}} = \pm x^{w_i}(p_i - x^{v_i} p_{b_{a_i}})$$

$$= \pm x^{w_i} p'_{i+1} = \pm x^{w_i} x^{u_i} sat(p'_{i+1}) = \pm x^{w_i} x^{u_i} p_{i+1}$$

By letting $x^{w_{i+1}} = x^{w_i} x^{u_i}$ the claim is true for $i+1$. We should notice that the "$\pm$" does not interfere with our argument. By induction the claim is always true also at the end of the algorithm. This means that if p_j is zero then also q_j is zero and we have shown that q_1 reduces to zero by some run of the division algorithm. □

Example 2.11 The converse of proposition 2.10 is false. With the lexicographic order $e \prec d \prec c \prec b \prec a$ the polynomial $ac - bd$ reduces to 0 by the division algorithm modulo $\{a - b, bc - e, bd - e\}$ but the polynomial reduces to $c - d$ by any run of the saturating division algorithm.

Instead of proving proposition 2.10 we could have shown that if the saturating division algorithm reduces a binomial to zero then the binomial "reduces to zero modulo $\{p_{b_1}, \ldots, p_{b_k}\}$" in the ordinary sense [Lauritzen]. This is actually just what we need when applying Buchberger's S-criterion in section 2.4. In this way we would get rid of the randomness in the division algorithm.

2.4 The saturating Buchberger algorithm

The vector representation also turns out to be useful when computing the S-polynomial of two pure binomials. Let $u, v \in \mathbb{Z}^n$ both non-zero and p_u and p_v be ordered with respect to $\prec$. According to the definition:

$$S(p_v, p_u) = \frac{x^{u^+ \vee v^+}}{in_\prec(p_v)} p_v - \frac{x^{u^+ \vee v^+}}{in_\prec(p_u)} p_u =$$

$$-\frac{x^{u^+ \vee v^+}}{x^{v^+}} x^{v^-} + \frac{x^{u^+ \vee v^+}}{x^{u^+}} x^{u^-} = x^{(u^+ \vee v^+) - u^+ + u^-} - x^{(u^+ \vee v^+) - v^+ + v^-}$$

$$= x^{(u^+ \vee v^+) - u} - x^{(u^+ \vee v^+) - v} = x^w (x^{(u^+ \vee v^+) - u - w} - x^{(u^+ \vee v^+) - v - w})$$

$$= x^w p_{(u^+ \vee v^+) - u - w - ((u^+ \vee v^+) - v - w)} = x^w p_{v-u}$$

where x^w is the greatest common divisor. Hence, $(u^+ \vee v^+) - u - w$ and $(u^+ \vee v^+) - v - w$ are non-negative and have disjoint support, proving the equality.

Now the idea is to forget the factor x^w and use $sat(S(p_v, p_u))$ as S-polynomial instead. For example in Buchberger's algorithm on toric ideals this seems like a good idea since the saturated S-polynomial must be in the ideal too.

Definition 2.12 *Let p_u, p_v be ordered pure binomials. The saturated S-polynomial is defined to be: $S_{sat}(p_v, p_u) = sat(S(p_v, p_u)) = p_{v-u}$*

The saturating Buchberger algorithm is the ordinary Buchberger algorithm where we compute saturated S-polynomials instead of S-polynomials and use the saturating division algorithm instead of the ordinary one.

Algorithm 2.13 *The saturating Buchberger algorithm*
Input: *A term order $\prec$ and a set of ordered pure binomials S generating an ideal an ideal J contained in a toric ideal I.*
Output: *A reduced Gröbner basis $\mathcal{G}$ with respect to $\prec$ for an ideal J' satisfying $J \subset J' \subset I$.*

```
G := ∅;
while(S ≠ ∅)
{
    choose p ∈ S;S := S\{p};
    q:= remainder of p mod G using the saturating division algorithm w.r.t. ≺;
    if(q ≠ 0)
    {
        for(r ∈ G) S := S ∪ {ordered S_sat(q, r)};
        G := G ∪ {q};
```

}
}
minimise and reduce $\mathcal{G}$*;*

Usually we will run this algorithm on a toric ideal, that is, with $J = I$. In this case we should notice that every time we loop $\mathcal{G} \cup S$ generates $I = J$. If $J \neq I$ we notice that every time we loop the set $\mathcal{G} \cup S$ generates an ideal that is containing J and is contained in I.

If the algorithm terminates it is clear that all saturated S-polynomials reduce to zero by some run of the saturating division algorithm modulo $\mathcal{G}$. By proposition 2.10 all saturated S-polynomials reduce to 0 by the division algorithm and therefore also the S-polynomials. By Buchberger's S-criterion this means that the new set of generators is a Gröbner basis. Termination of the ordinary Buchberger algorithm is shown by observing that the monomial ideal generated by the initial terms of the elements in $\mathcal{G}$ increases every time we add a new polynomial. The same argument is valid for the saturating Buchberger algorithm.

Minimising $\mathcal{G}$ is just a matter of deleting elements. The reduction of $\mathcal{G}$ can be done by running the division algorithm on each element $p \in \mathcal{G}$ modulo $\mathcal{G}\backslash\{p\}$ since this only changes the tail of p. When J is toric (and $J = I$) this process only produces pure binomials - if not the pure binomial factor would be in J and its initial term would divide the initial term of p non-trivially contradicting that p is an element in a minimal Gröbner basis with respect to $\prec$. That the produced binomials are pure allows us to use the saturating division algorithm instead of the ordinary one for the reduction of $\mathcal{G}$. This gives us a reduced Gröbner basis of $J = I$ consisting of pure binomials. When the ideal J is not toric it is not clear if using the division algorithm for reducing $\mathcal{G}$ will produce pure binomials so we should not apply the saturating division algorithm instead of the ordinary one in this case.

This seems fine. We may use the saturating Buchberger algorithm on generators for toric ideals. But how do we use this algorithm to compute generators for a toric ideal I_A? If A has positive entries algorithm 2.2 tells us how. We just have to prove that the ideal J in the algorithm is toric in order to use the saturating Buchberger algorithm directly.

Proposition 2.14 *Let* $A \in mat_{dn}(\mathbb{N})$ *be a matrix with non-negative entries. The ideal* $J = \langle x_1 - t^{a_1}, \ldots, x_n - t^{a_n} \rangle$ *in* $k[\mathbf{x}][t_1, \ldots, t_d]$ *is toric.*

Proof. Continuing the proof of proposition 2.1 it remains to be shown that $J = ker(\pi_B)$.
$\subset$: $x_i - t^{a_i} \in ker(\pi_B)$.
$\supset$: By lemma 1.9 $ker(\pi_B)$ is generated by binomials of the form $x^{u^+}t^{v^+} - x^{u^-}t^{v^-}$ where $u \in \mathbb{Z}^n, v \in \mathbb{Z}^d$ and $B(u \oplus v) = 0$ meaning $Au + v = 0$. We want to prove that each such binomial is in J. That $x \sim t^{a_i}$ modulo J implies $x^w \sim t^{Aw}$ for $w \in \mathbb{N}^n$. Hence, computing modulo J we get: $x^{u^+}t^{v^+} - x^{u^-}t^{v^-} \sim t^{Au^+}t^{v^+} - t^{Au^-}t^{v^-}$

$= t^{Au^+ + v^+} - t^{Au^- + v^-}$. Since $Au + v = 0$ we get $Au^+ + v^+ - Au^- - v^- = 0$, so $Au^+ + v^+ = Au^- + v^-$. Hence, $x^{u^+}t^{v^+} - x^{u^-}t^{v^-} \in J$. □

There are two main reasons for using the saturating Buchberger algorithm:

- Saturating binomials during the computation increases the ideal generated by the initial terms of the generators faster and hopefully thereby the algorithm will terminate earlier.
- The data structures for vector representation are simpler.

3 Saturating ideals

Algorithm 2.2 introduced the new variables $t_1, \ldots, t_d$ before applying Buchberger's algorithm. The running time of Buchberger's algorithm depends heavily on the number of variables. Even when using the faster saturating Buchberger algorithm the method in algorithm 2.2 is slow. In this section we will study an alternative method for computing generators for a toric ideal given a matrix defining it. We start by defining the homogeneous ideals.

3.1 Homogeneous ideals

We define what it means to be homogeneous with respect to a grading:

Definition 3.1 *A polynomial $p = \sum_{v \in \mathbb{N}^n} c_v x^v \in k[\mathbf{x}]$ ($c_v \in k$) is homogeneous with respect to a vector $d = (d_1, \ldots, d_n) \in \mathbb{Z}^n$ if $v \cdot d$ is constant for all $c_v \neq 0$. The vector d is called a grading and the constant number $v \cdot d$ is called the d-degree of p.*

Sometimes we will use the words "d-homogeneous" and "d-degree" even when $d \in \mathbb{R}^n$.

Definition 3.2 *An ideal $I \subset k[\mathbf{x}]$ is homogeneous with respect to a grading $d = (d_1, \ldots, d_n) \in \mathbb{Z}^n$ if I is generated by a set of polynomials each homogeneous with respect to d.*

Definition 3.3 *Let $I \subset k[\mathbf{x}]$ be an ideal, $d = (d_1, \ldots, d_n) \in \mathbb{Z}^n$ be a grading and $m \in \mathbb{Z}$. The vector space of polynomials in I of d-degree m together with the zero polynomial is denoted I_m.*

Proposition 3.4 *Let $I \subset k[\mathbf{x}]$ be a homogeneous ideal with respect to a non-negative grading $d = (d_1, \ldots, d_n) \in \mathbb{N}^n$. As a vector space $I = \bigoplus_{i=0}^{\infty} I_i$.*

Proof. Let $f \in I$. Since I is homogeneous there exist d-homogeneous generators for I meaning that there exist polynomials $h_i \in I$ and $a_i \in k[\mathbf{x}]$ such that $f = \sum_{i=1}^{k} a_i h_i$. h_i has non-negative d-degree for all i. We can split each a_i into monomials. Hence, we can write $f = \sum_{i=1}^{k'} h_i'$ with $h_i' \in I_i$. (We may assume that the h's have different d-degrees, if not we just add the polynomials with the same degree). The uniqueness of this sum is clear since no terms can cancel in such a sum. □

If an ideal I is d-homogeneous it is generated by d-homogeneous polynomials: Hilbert's basis theorem tells us that we need only a finite number of elements to generate I. Each of these polynomials can be written as $f = g_1 h_1 + \ldots + g_k h_k$ where $h_1, \ldots, h_k$ are some of the d-homogeneous generators of I. Consequently

we need only a finite number of d-homogeneous generators to generate I. Furthermore, having this finite set of d-homogeneous generators we can compute a Gröbner basis for I with respect to a term order using the Buchberger algorithm. Notice that this process only produces d-homogeneous polynomials. Especially, this means that any reduced Gröbner basis for I consists of d-homogeneous elements.

By proposition 1.8 a toric ideal I_A is homogeneous with respect to any row vector in A. Hence, it is homogeneous with respect to any vector in the row-space of A.

3.2 Saturating ideals

We now define the ideal quotients:

Definition 3.5 *Let R be a commutative ring, $I \subset R$ an ideal and $f \in R$. We define*

$$(I : f) = \{g \in R | gf \in I\}$$
$$(I : f^\infty) = \{g \in R | gf^n \in I \text{ for some } n \in \mathbb{N}\}$$

.

This definition makes $(I : f)$ and $(I : f^\infty)$ sets containing I. These sets are in fact ideals in R:

Proposition 3.6 *$(I : f)$ and $(I : f^\infty)$ are ideals in I.*

Proof. Case $(I : f^\infty)$:Let $g, g' \in (I : f^\infty)$. Then $gf^n \in I$ and $g'f^{n'} \in I$ for some $n, n' \in \mathbb{N}$. This implies that $gf^{n''}, g'f^{n''} \in I$ for $n'' = max(n, n')$ and we conclude $(g + g')f^{n''} \in I$ and $g + g' \in (I : f^\infty)$. Clearly, multiplication by an element from $(I : f^\infty)$ gives a new element in $(I : f^\infty)$. □

The following lemma taken from [Sturmfels] tells us how to compute ideal quotients in $k[\mathbf{x}]$ with respect to one of the variables:

Proposition 3.7 *Let $I \subset k[\mathbf{x}]$ be a homogeneous ideal with respect to some grading $v \in \mathbb{N}^n_{>0}$. Let $\prec$ be a term order satisfying (for all v-homogeneous elements $f \in k[\mathbf{x}]$):*

$$x_n | in_\prec(f) \Rightarrow x_n | f$$

If G is a Gröbner basis for $I \subset k[\mathbf{x}]$ with respect to $\prec$ consisting of v-homogeneous elements then

$$G' = \{f \in G : x_n \not| f\} \cup \{f/x_n : f \in G, x_n | f\}$$

is a Gröbner basis for $(I : x_n)$ with respect to $\prec$ and

$$G'' = \{f/x_n^i : f \in G, x_n^i | f, x_n \not| f/x_n^i\}$$

is a Gröbner basis for $(I : x_n^\infty)$ with respect to $\prec$.

Proof. We will prove and use only the last claim. Clearly, $\langle G'' \rangle \subset (I : x_n^\infty)$. To prove $in_\prec(I : x_n^\infty) \subset in_\prec(G'')$ let $g \in (I : x_n^\infty)$. We want to show that $in_\prec(g) \in in_\prec(G'')$. There exists an r such that $gx_n^r \in I$. Since G is a Gröbner basis for I and $gx_n^r \in I$ there must be an $f \in G$ such that $in_\prec(f)|in_\prec(gx_n^r) = in_\prec(g)x_n^r$. Let R be the number of times that x_n divides f. By the choice of term order this is also the number of times that x_n divides $in_\prec(f)$ since f is v-homogeneous. $in_\prec(f/x_n^R)x_n^R|in_\prec(g)x_n^r$. $in_\prec(f/x_n^R)$ does not contain any x_n. Hence, $in_\prec(f/x_n^R)|in_\prec(g)$. And we are done since $f/x_n^R \in G''$. □

Remark 3.8 Clearly, the condition on the term order $\prec$ in proposition 3.7 cannot be satisfied for all polynomials in $k[\mathbf{x}]$: x divides $in_\prec(x-1)$ but not $x-1$. Given a weight vector $v \in \mathbb{R}^n_{>0}$ the graded reverse lexicographic term order $\prec_{grevlex}$ is defined by:

$$x^a \prec_{grevlex} x^b \Leftrightarrow a \cdot v < b \cdot v$$

$$\vee(a \cdot v = b \cdot v \wedge (\exists i : a_i > b_i \wedge a_j = b_j \text{ for } j = i+1, \ldots, n))$$

Observe, for v-homogeneous f: $x_n|in_{\prec_{grevlex}}(f) \Rightarrow x_n|f$. So $\prec_{grevlex}$ can be used in the proposition.

We introduce the concept of saturated ideals and show some basic properties:

Definition 3.9 *Let $f \in k[\mathbf{x}]$. An ideal $I \subset k[\mathbf{x}]$ is f-saturated if $(I : f^\infty) = I$*

Lemma 3.10 *Let I be an ideal in $k[\mathbf{x}]$ and $f, g \in k[\mathbf{x}]$ then*

$$(I : (fg)^\infty) = ((I : f^\infty) : g^\infty)$$

Proof.
$\subset$: Let $h \in (I : (fg)^\infty) \Rightarrow h(fg)^n \in I \Rightarrow (hg^n)f^n \in I \Rightarrow hg^n \in (I : f^\infty) \Rightarrow h \in ((I : f^\infty) : g^\infty)$ for some n.
$\supset$: Let $h \in ((I : f^\infty) : g^\infty) \Rightarrow hg^n \in (I : f^\infty) \Rightarrow hg^n f^m \in I \Rightarrow h(gf)^{max(n,m)} \in I \Rightarrow h \in (I : (gf)^\infty)$ for some n, m. □

Corollary 3.11 *An ideal $I \subset k[\mathbf{x}]$ is (fg)-saturated if it is f-saturated and g-saturated.*

Proof. We know that $(I : f^\infty) = I = (I : g^\infty)$. Hence, $I = (I : g^\infty) = ((I : f^\infty) : g^\infty) = (I : (fg)^\infty)$. □

Remark 3.12 If an ideal $I \subset k[\mathbf{x}]$ is fg-saturated then it is f-saturated. This can be seen by using the lemma and the definition to get the inclusions:

$$(I : f^\infty) \supset I = (I : (fg)^\infty) = ((I : f^\infty) : g^\infty) \supset (I : f^\infty)$$

Lemma 3.13 *If I and J are ideals in $k[\mathbf{x}]$ satisfying $I \subset J \subset (I : f^\infty)$ then $(J : f^\infty) = (I : f^\infty)$*

Proof. The inclusion $\supset$ follows from $I \subset J$. To prove the other inclusion let $h \in (J : f^\infty)$ then $hf^n \in J \subset (I : f^\infty)$ and h times f to some power is indeed in I. □

The connection between toric ideals and saturation becomes clear in the following lemma and proposition:

Lemma 3.14 *A toric ideal $I_A \subset k[\mathbf{x}]$ is x_i saturated.*

Proof. We must show that $I_A \supset (I_A : x_i^\infty)$. Let $q \in (I_A : x_i^\infty)$. That is, there exists an $n \in \mathbb{N}$ such that $qx_i^n \in I_A$. I_A is prime and $x_i \notin I_A = ker(\pi_A)$ since a monomial maps to a monomial by the definition of π_A. Hence, $q \in I_A$. □

Definition 3.15 *The lattice kernel of a matrix $A \in mat_{dn}(\mathbb{Z})$ is the set $ker(A) \cap \mathbb{Z}^n$.*

Definition 3.16 *If $C \subset \mathbb{Z}^n$ then we define*

$$J_C = \langle p_v | v \in C \rangle$$

Proposition 3.17 *C spans the lattice kernel of A ($span_{\mathbb{Z}}(C) = \mathbb{Z}^n \cap ker(A)$) if and only if $(J_C : (x_1 x_2 \dots x_n)^\infty) = I_A$.*

Proof. We will prove (and use) only the only-if direction. We must show that $(J_C : (x_1 x_2 \dots x_n)^\infty) = I_A$.
$\supset$: By proposition 1.8 I_A is generated by binomials p_u where $u \in \mathbb{Z}^n$ and $Au = 0$. Since C spans the lattice $ker(A)$ this means that u can be written $u = \sum_{i=1}^r \lambda_i v_i$ with $v_i \in C$ and $\lambda_i \in \mathbb{Z}$. We now observe: $p_a \in (J_C : (x_1 \dots x_n)^\infty) \Rightarrow p_{-a} \in (J_C : (x_1 x_2 \dots x_n)^\infty)$ for $a \in \mathbb{Z}^n$. Moreover, $p_a, p_b \in (J_C : (x_1 x_2 \dots x_n)^\infty) \Rightarrow p_{a-b} \in (J_C : (x_1 x_2 \dots x_n)^\infty)$ for $a, b \in \mathbb{Z}^n$ using the identity in the proof of lemma 1.19. By induction $p_u \in (J_C : (x_1 x_2 \dots x_n)^\infty)$.
$\subset$: $J_C \subset I_A \subset (J_C : (x_1 x_2 \dots x_n)^\infty)$. Using lemma 3.13 we see that $(J_C : (x_1 x_2 \dots x_n)^\infty) = (I_A : (x_1 x_2 \dots x_n)^\infty)$. Lemma 3.14 tells us that $(I_A : (x_1 x_2 \dots x_n)^\infty) = I_A$ and the lemma follows. □

The lemma gives us the following algorithm for computing generators for a toric ideal:

Algorithm 3.18
Input: *A matrix $A \in mat_{dn}(\mathbb{Z})$ with a strictly positive vector in its row-space.*
Output: *Generators for I_A.*
Compute a set $C \subset \mathbb{Z}^n$ spanning the lattice kernel of A. Compute generators for $(J_C : (x_1 x_2 \dots x_n)^\infty)$. These generators are generators for I_A.

Remark 3.19 Lemma 3.10 tells us that we may saturate J_C with respect to one variable at a time, successively computing

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

Proposition 3.7 gives an algorithm for doing this using the Buchberger algorithm. Saturating the ideal is the most time consuming part of algorithm 3.18. Since the time for the Buchberger algorithm depends on the exponents of the generators it would be good to use sets of "small" or "reduced" generators. Such sets (Lenstra Lenstra Lovász reduced bases) are described in section 4.1 where there also is an algorithm for finding a such reduced basis. A by-product of that algorithm is an algorithm for computing generators for the lattice kernel of A. Example 4.19 shows how this works.

Remark 3.20 It is possible to use the saturating Buchberger algorithm in algorithm 3.18 instead of the ordinary Buchberger algorithm. Using the saturating Buchberger algorithm we compute generators for ideals (see remark 3.19): $I_0, I_1, \ldots, I_n$ each satisfying $I_i \subset I_A$ and $(J_C : (x_1 x_2 \ldots x_i)^\infty) \subset I_i$ for $i = 0, \ldots, n$ with $I_0 = J_C$. The first inclusion is satisfied since all generators at the beginning are in I_A and each new generator is computed by subtracting vector representatives of existing generators which gives a new element in I_A by lemma 1.19. The second inclusion is proved by induction. For $i = 0$ the left hand side equals the right hand side. Assume the inclusion is true for $i - 1$. We first observe that $I_{i-1} \subset I_i$ and secondly that I_i is x_i-saturated: $(I_i : x_i^\infty) = I_i$. Thereby, $(I_{i-1} : x_i^\infty) \subset (I_i : x_i^\infty) = I_i$. The induction hypothesis tells us that $(J_C : (x_1 x_2 \ldots x_{i-1})^\infty) \subset I_{i-1}$ implying $(J_C : (x_1 x_2 \ldots x_i)^\infty) \subset (I_{i-1} : x_i^\infty) \subset I_i$. At the end $i = n$ and $(J_C : (x_1 x_2 \ldots x_n)^\infty) \subset I_n \subset I_A$. Consequently, the computed generators for I_n are generators for $I_A = (J_C : (x_1 x_2 \ldots x_n)^\infty)$.

4 Lattices

In this section we will study lattices. The purpose is to find a set of small generators for a lattice suitable for computations. A by-product of this section is a method for computing generators for the lattice kernel of a matrix $A \in mat_{dn}(\mathbb{Z})$.

4.1 Lattices

Definition 4.1 *Let $n \in \mathbb{N}$ and let $B = \{b_1, \ldots, b_n\}$ be a basis of $\mathbb{R}^n$. The subset $L \subset \mathbb{R}^n$ defined by $L = \sum_{i=1}^{n} \mathbb{Z}b_i = \{a_1b_1 + \ldots + a_nb_n | a_1, \ldots, a_n \in \mathbb{Z}\}$ is called a lattice of rank n. The set $\{b_1, \ldots, b_n\}$ is called a lattice basis of L.*

A lattice L has many bases (if $n > 1$). The purpose of this section is to find a basis that is suitable for computations, meaning that the entries of its vectors are small.

Lemma 4.2 *Let $L \subset \mathbb{R}^n$ be a lattice of rank n. Let $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_n\}$ be two bases for L. Then $|det(a_1, \ldots, a_n)| = |det(b_1, \ldots, b_n)|$.*

Proof. A and B are both subsets of L. Hence, a_i and b_i can be written:

$$a_i = \lambda_{i1}b_1 + \ldots + \lambda_{in}b_n$$

$$b_i = \mu_{i1}a_1 + \ldots + \mu_{in}a_n$$

for $i = 1, \ldots, n$ for some $M_1 = (\lambda_{ij}) \in mat_{nn}(\mathbb{Z})$ and $M_2 = (\mu_{ij}) \in mat_{nn}(\mathbb{Z})$. We have $M_1M_2 = I$ and therefore $det(M_1)det(M_2) = det(M_1M_2) = det(I) = 1$. Since the determinants of M_1 and M_2 are integers they must be $+1$ or -1. M_1 changes B to A implying $det(a_1, \ldots, a_n) = det(M_1)det(b_1, \ldots, b_n)$ and the conclusion follows. □

The lemma tells us that we cannot choose a basis of arbitrary small vectors since the absolute value of the determinant must be constant. What is a "small" basis (with the same |determinant|)? If we were not looking for a lattice basis but for a basis of $\mathbb{R}^n$ the answer would be an orthogonal basis. An almost orthogonal basis could be suitable for a lattice.

4.2 A reduced basis in $\mathbb{R}^2$

Let $B = \{b_1, b_2\}$ be a lattice basis of a lattice $L \subset \mathbb{R}^2$. If we wanted to make B an orthogonal basis of $\mathbb{R}^2$ using the Gram-Schmidt process we would subtract the projection of b_2 onto b_1 from b_2: $b_2^* = b_2 - \mu b_1$, $\mu = \frac{b_1 \cdot b_2}{b_1 \cdot b_1} \in \mathbb{R}$. However, to make B an almost orthogonal lattice basis of L we subtract b_1 from b_2 $k \in \mathbb{Z}$ times, where k is one of the integers nearest to μ. $\{b_1, b_2\}$ is still a basis for the lattice and hopefully the new b_2 is shorter than the old one. This leads us to the first condition for a lattice basis $\{b_1, b_2\} \subset \mathbb{R}^2$ to be reduced:

- $|\frac{b_1 \cdot b_2}{b_1 \cdot b_1}| \leq \frac{1}{2}$

It can be satisfied simply by making the adjustment above. An other condition could be $|\frac{b_2 \cdot b_1}{b_2 \cdot b_2}| \leq \frac{1}{2}$. However, we do not require this. Instead we choose the second condition to be

- $||b_2||^2 \geq \frac{3}{4}||b_1||^2$

This condition is known as the Lovász condition. Have a look at the figure below. If the first condition is satisfied then the second condition almost always (or always almost) implies $|\frac{b_2 \cdot b_1}{b_2 \cdot b_2}| \leq \frac{1}{2}$. When computing a reduced basis if the second condition is not satisfied we swap b_1 and b_2 and try to satisfy the first condition. Later we will see that repeating this process eventually leads to a reduced basis.

Given the vector b_1 the figure shows the regions b_2 must be situated in to satisfy the 3 conditions.

- The rectangular region: $|\frac{b_1 \cdot b_2}{b_1 \cdot b_1}| \leq \frac{1}{2}$.
- The region outside the two circles: $|\frac{b_2 \cdot b_1}{b_2 \cdot b_2}| \leq \frac{1}{2}$.
- The region outside the dotted circle: $||b_2||^2 \geq \frac{3}{4}||b_1||^2$.

4.3 A reduced basis in $\mathbb{R}^n$

We now generalise to $\mathbb{R}^n$. Let $B = \{b_1, \ldots, b_n\}$ be a basis of $\mathbb{R}^n$ spanning a lattice L. We wish to change B so that the elements of B become almost orthogonal. We recall the Gram-Schmidt process:

$$b_1^* := b_1$$

$$b_2^* := b_2 - \mu_{21} b_1^*$$

$$\ldots$$

$$b_n^* := b_n - \mu_{n1} b_1^* - \ldots - \mu_{n(n-1)} b_{n-1}^*$$

where $\mu_{ij} = \frac{b_j^* \cdot b_i}{b_j^* \cdot b_j^*}$ for $1 \leq j < i \leq n$. b_i^* is the projection of b_i onto $span_{\mathbb{R}}\{b_1, \ldots, b_{i-1}\}^{\perp}$. A reasonable condition for B to be called reduced is that it is not possible to "orthogonalise" it further in a way suggested by the Gram-Schmidt process. That is by letting $b_i := b_i - \lceil \mu_{ij} \rfloor b_j$. This leads us to the condition:

- $|\mu_{ji}| = |\frac{b_i^* \cdot b_j}{b_i^* \cdot b_i^*}| \leq \frac{1}{2}$ for $1 \leq i < j \leq n$

We now translate the Lovász condition from the 2-dimensional case to the n-dimensional case. For $i \in \{2, \ldots, n\}$ we will require something like $||b_i||^2 > \frac{3}{4}||b_{i-1}||^2$. But instead of taking the lengths in $\mathbb{R}^n$ we require that the projections of the vectors to $span_{\mathbb{R}}\{b_1, \ldots, b_{i-2}\}^{\perp}$ satisfy the inequality. This gives us the following definition:

Definition 4.3 *A set of linearly independent vectors $B = \{b_1, \ldots, b_n\}$ spanning a lattice L is called an LLL-reduced basis of L if:*

- $|\frac{b_i^* \cdot b_j}{b_i^* \cdot b_i^*}| = |\mu_{ji}| \leq \frac{1}{2}$ *for* $1 \leq i < j \leq n$
- $||b_i^* + \mu_{i(i-1)} b_{i-1}^*||^2 \geq \frac{3}{4}||b_{i-1}^*||^2$ *for* $i \in \{2, \ldots, n\}$

where b_i^* and μ_{ij} are given by the Gram-Schmidt process.

The second condition is called the Lovász condition.

Remark 4.4 A priori it is not clear that an LLL-reduced basis exists. Its existence will be shown in the next section.

Remark 4.5 For $n > 0$ an LLL-reduced basis is not unique.

Remark 4.6 The constant $\frac{3}{4}$ in the definition does not have to be $\frac{3}{4}$. It can be chosen anywhere between $(\frac{1}{2})^2$ and 1. The constant plays an important role when showing termination of the LLL-algorithm in the following section. The proof requires the constant to be less than 1. The other limit is important when proving properties of reduced bases. In particular we will use it in section 4.5.

For properties of LLL-reduced bases, I refer to [Lenstra,..].

4.4 Lenstra Lenstra Lovász algorithm

Let $B = \{b_1, \ldots, b_n\} \subset \mathbb{R}^n$ be a lattice basis of L. In this section we will prove that there exists a reduced lattice basis of L and we will provide an algorithm for finding it.

In the following let at any time μ_{ij} denote the coefficients from the Gram-Schmidt process run on $\{b_1, \ldots, b_n\}$ and let $\{b_1^*, \ldots, b_n^*\}$ be the resulting orthogonal basis.

The algorithm has two rules for changing the lattice basis B:

1. If for a vector b_k and an index $i < k$ we have that $|\mu_{ki}| > \frac{1}{2}$ we modify b_k by subtracting each vector b_j $(1 \leq j < k)$ an integral number of times ending up with a b_k for which $|\mu_{kj}| \leq \frac{1}{2}$ for all $j \in \{1, \ldots, k-1\}$.
2. If for some i the Lovász condition is not satisfied we exchange b_i and b_{i-1}. We immediately achieve that their projections to $span_{\mathbb{R}}\{b_1, \ldots, b_{i-2}\}^{\perp}$ satisfy the inequality in the Lovász condition.

The first rule needs some comments. At first we are tempted to modify b_k in this way:

$$b_k := b_k - \lceil \mu_{k1} \rfloor b_1 - \ldots - \lceil \mu_{k(k-1)} \rfloor b_{k-1};$$

However, the b_j's are not necessarily orthogonal to the b_j^*'s, so subtracting b_i from b_k might affect other μ_k's than μ_{ki}. But it cannot affect $\mu_{kj} = \frac{b_j^* \cdot b_k}{b_j^* \cdot b_j^*}$ for $j > i$ since b_i is orthogonal to b_j^*. As a consequence the wanted conditions on the μ_k's can be obtained by starting with the largest indices first. The following loop illustrates this:

for$(i = k-1, k-2, \ldots, 1)$
{
    compute the μ's;
    $b_k := b_k - \lceil \mu_{ki} \rfloor b_i$;
}

Using the two rules in the right order it is possible to compute a reduced lattice basis:

Algorithm 4.7 *Lenstra Lenstra Lovász*
Input: *A lattice basis* $B = \{b_1, \ldots, b_n\} \subset \mathbb{R}^n$ *for a lattice* $L \subset \mathbb{R}^n$
Output: *An LLL-reduced lattice basis* $B = \{b_1, \ldots, b_n\} \subset \mathbb{R}^n$ *for* L
$k := 2$;
while$(k \leq n)$
{
    $\mu :=$*coefficients from the Gram-Schmidt process run on* $\{b_1, \ldots, b_k\}$;
    $\{b_1^*, \ldots, b_k^*\} :=$ *result of the Gram-Schmidt process;*
    change b_k *by rule 1;*
    recompute μ*'s and* $\{b_1^*, \ldots, b_k^*\}$;
    if$(||b_k^* + \mu_{k(k-1)} b_{k-1}^*||^2 < \frac{3}{4}||b_{k-1}^*||^2)$
    {
        $(b_{k-1}, b_k) := (b_k, b_{k-1})$;
        $k := k - 1$;
        if $(k = 1) k := 2$;
    }
    else $k := k + 1$;
}

Remark 4.8 We notice that swapping elements is done in an insertion sort fashion. In fact, if B is an orthogonal basis and for all $i \neq j$: $||b_i||^2 \geq 2||b_j||^2$ or $||b_j||^2 \geq 2||b_i||^2$ the algorithm is just a sorting algorithm sorting the vectors by their lengths.

Proof. Correctness: The following is an invariant which is true every time we loop (after calculating μ and $\{b_1^*, \ldots, b_k^*\}$):

- For $1 \leq i < j < k$: $|\mu_{ji}| \leq \frac{1}{2}$
- For $1 < i < k$:$||b_i^* + \mu_{i(i-1)} b_{i-1}^*||^2 \geq \frac{3}{4}||b_{i-1}^*||^2$
- $\{b_1, \ldots, b_n\}$ is a lattice basis of L

The last claim is true because we only change a vector in $\{b_1, \ldots, b_n\}$ by adding an other vector from the same set an integral number of times.

When $k = 2$ the invariant is obviously true. And the first claim in the invariant remains true at each iteration, either because we decrease k by one and have not changed the vectors $\{b_1, \ldots, b_{k_{new}-1}\}$ or because we adjust b_k allowing us to increase k.

The second claim is true at each iteration:

- In the case where we decrease k the claim involves only the vectors $\{b_1, \ldots, b_{k_{old}-2}\}$ which we have not changed and which we already know satisfy the claim.
- In the case where we increase k we first observe that we only change the vector $b_{k_{new}-1}$ which only is involved in one inequality and this is explicitly verified by the if-statement.

If the algorithm terminates we must have $k = n + 1$ and the invariant now tells us that we actually have found an LLL-reduced basis.

Termination: We define $d_i = det((b_j \cdot b_l)_{1 \leq j,l \leq i})$. This determinant is the same as the determinant of the product of C and $C^T \in mat_{nn}(\mathbb{R})$, where the first i rows of C are the vectors $b_1, \ldots, b_i$ and the remaining rows are $V = \{v_{i+1}, \ldots, v_n\}$, where V is an orthonormal basis of $span_{\mathbb{R}}(b_1, \ldots, b_i)^{\perp}$. Using Gram Schmidt C can be transformed into a matrix C' by simple row operations without changing the determinant, where the first i rows of C' are $b_1^*, \ldots, b_i^*$ and the rest are V. Since the rows of C' are orthogonal, $\det(C) = \det(C') = \prod_{j=1}^{i} ||b_j^*||^2$. We conclude that $d_i = \prod_{j=1}^{i} ||b_j^*||^2$.

Define $D = \prod_{i=1}^{n-1} d_i$ which is positive. We will now see what happens to D at each iteration:

- The adjustment of b_k does not change the vectors $b_1^*, \ldots, b_n^*$. Hence, D does not change either.
- The only d_i that is affected by the swapping of b_{k-1} and b_k is d_{k-1}, since for the other d_i's d_i is a determinant of the same matrix as before or the same matrix as before with just two rows and two columns exchanged. In the product "$d_{k-1} = \prod_{j=1}^{k-1} ||b_j^*||^2$" b_{k-1}^* is substituted by $b_k^* + \mu_{k(k-1)} b_{k-1}^*$ whose length is less than $\sqrt{\frac{3}{4}}$ of the length of b_{k-1}^*. Hence, the new D must be $< \frac{3}{4}$ times the old D. That is D is decreased every time we decrease k.

If we can show that D is larger than a positive constant which only depends on L, we have shown that k can only be decreased finitely many times. Since $1 \leq k \leq n+1$ we also would know that k can be increased only finitely many times. At each iteration we change k. This implies that we cannot iterate for ever. Hence, the algorithm terminates.

To prove that D is larger than a positive constant it suffices to show that d_i is for all i. If the input vectors are in $\mathbb{Z}^n$ the proof is simple. The number d_i is positive and an integer and we must have $d_i \geq 1$. For the general case I refer to [Lenstra,..]. Their proof depends heavily on the fact that the set $\{||x||^2 | x \in L \backslash \{0\}\}$ has a smallest element. This completes the proof. □

Remark 4.9 If $L \subset \mathbb{Z}^n$ using the proof for termination it is easy to give an upper limit for the number of iterations. Given the input vectors, let $B \geq ||b_i^*||^2$ for $i = 1, 2, \ldots, n$. D is positive and computed from integers. Hence, $D \geq 1$ always. At the beginning $d_i \leq B^i$ and consequently $D \leq B^{\frac{1}{2}(n-1)n}$. Decreasing D by a factor $\frac{3}{4}$ is only possible $O(log(D))$ times. We conclude that the main loop of the algorithm terminates in $O(n^2 \log(B))$ iterations. (This is not an estimate for the total running time.)

Remark 4.10 The given algorithm differs from the original algorithm given by [Lenstra,..]. The structure of their loop is slightly different. It does not update b_k completely but only with respect to b_{k-1} in the case where we swap. This is sufficient because b_k has to be updated with respect to $b_{k-2}, \ldots, b_1$, in the next iteration(s) anyway.

Moreover, they observe that calculating μ_{ij} at each iteration is not necessary. Instead, they carefully update some of the μ's each time they modify one of the basis vectors.

Both differences are due to optimisation. I have left out these optimisations in order to keep the algorithm clear.

Remark 4.11 Because of the divisions the numbers computed in the algorithm are not necessarily integer. When implementing the algorithm we have to either use floating point arithmetics or do the calculations as fractions. However, observing that $d_i \mu_{ji} \in \mathbb{Z}$ when the input is integer vectors [Lenstra,..] modify the algorithm to compute with integers only. This is a part of their complexity analysis. We will not go into details here. The algorithm I implemented uses floating point calculations and works fine for our purpose.

4.5 Dependent integer vectors

Definition 4.12 *Let $n, l \in \mathbb{N}$ and $l \leq n$. Let $\{b_1, \ldots, b_l\}$ be a set of independent vectors in $\mathbb{R}^n$. The set L defined by $L = \sum_{i=1}^{l} \mathbb{Z} b_i = \{a_1 b_1 + \ldots + a_l b_l | a_1, \ldots, a_l \in \mathbb{Z}\}$ is called a lattice of rank l. The set $\{b_1, \ldots, b_l\}$ is called a lattice basis of L.*

Definition 4.3 also applies to a lattice of rank l (substituting n by l in the definition).

In this section we see an other version of the LLL-algorithm. The input must be n integer vectors which we allow to be dependent. The algorithm shows that these vectors generate a lattice in $\mathbb{R}^n$ of rank $\leq n$. The algorithm finds an independent set of generators for this lattice and finds dependencies between input vectors. Let $A \in mat_{nn}(\mathbb{Z})$ be the matrix with columns $b_1, \ldots, b_n$. The algorithm computes a basis of the kernel of A. In fact the computed basis is a lattice basis of the lattice $\ker(A) \cap \mathbb{Z}^n$.

Let us see what happens when we try to run the ordinary algorithm with the new type of input (n vectors $b_1, \ldots, b_n \in \mathbb{Z}^n$ that may be dependent). Since the input vectors may be dependent we might have a problem when using the Gram-Schmidt process. $\mu_{ij} = \frac{b_j^* \cdot b_i}{b_j^* \cdot b_j^*}$ cannot be computed when b_j^* is zero. Instead we define $\mu_{ij} = 0$ when b_j^* is zero. This is the only change we make in the LLL-algorithm. The set generated by integer linear combinations of the input vectors is denoted L. A priori we do not know that L is a lattice.

For the new LLL-algorithm we have to consider:

- Is the invariant still valid? Is the output an LLL-reduced basis?
- Does the algorithm terminate?

The last claim of the invariant of course needs to be changed to: $\{b_1, \ldots, b_n\}$ generates L. This is true since we only change a generator in $\{b_1, \ldots, b_n\}$ by adding an other generator an integral number of times. The rest of the invariant is still valid and can be shown in the same way as before.

Termination:

Let at any time $S = \{i \in \{1, \ldots, n\} | b_i^* = 0\}$. $|S|$ is constant since the set $\{b_i^* | b_i^* \neq 0\}$ is a basis of $span_{\mathbb{R}}\{b_1, \ldots, b_n\}$. Notice that S does not change when we update by rule 1. Swapping b_{k-1} with b_k only changes b_{k-1}^* and b_k^*. Notice that if $k-1 \in S$ we cannot swap b_{k-1} with b_k since the corresponding Lovász condition is true. This implies that a "zero" vector can only be swapped "upwards". Therefore the set S can only be changed a finite number of times. (S is not changed if $k-1, k \in S$.)

For each configuration of S we now show that the algorithm cannot loop forever. Define $d_i = \prod_{1 \leq j \leq i, j \notin S} ||b_j^*||^2$. By an argument analogous to the argument we used in the ordinary LLL-algorithm we see that $d_i = det(b_j \cdot b_l)_{1 \leq j,l \leq i \wedge j,l \notin S}$. Define $D = \prod_{i=1}^n d_i$.

As before: updating by rule 1 does not change D since it does not change $\{b_1^*, \ldots, b_n^*\}$. When swapping b_{k-1}^* is not zero ($k-1 \notin S$). If $k \notin S$ d_{k-1} is the only d_i affected since for $i \neq k-1$ d_i is the determinant of the same matrix as before or the same matrix with two rows and two columns exchanged. b_{k-1}^* is substituted by $b_k^* - \mu_{k(k-1)} b_{k-1}^*$ which makes the new d_{k-1} less than $\frac{3}{4}$ times the

old. In the case $k \in S$ d_l is only affected for $l \geq k-1$ assuming that S does not change. But all of these are also decreased by at least a factor $\frac{3}{4}$. In both cases D is decreased by at least a factor $\frac{3}{4}$. Since the input vectors are integer we conclude that the algorithm terminates by the same argument we used for the ordinary LLL-algorithm.

We have seen that the algorithm satisfies the invariant and that it terminates. It is now time to examine its output.

Let $b_1, \ldots, b_n$ be the output. If for some $i \in \{2, 3, \ldots, n\}$ we have that $b_i^* = 0$ the Lovász condition tells us:

$$||b_i^* + \mu_{i(i-1)} b_{i-1}^*||^2 \geq \frac{3}{4}||b_{i-1}^*||^2 \Rightarrow \mu_{i(i-1)}^2 ||b_{i-1}^*||^2 \geq \frac{3}{4}||b_{i-1}^*||^2 \Rightarrow$$

$$|\mu_{i(i-1)}| \geq \sqrt{\frac{3}{4}} \vee b_{i-1}^* = 0$$

The first cannot be true and we have that $b_{i-1}^* = 0$. (Here we needed that the constant $\frac{3}{4}$ had been chosen to be larger than $\frac{1}{4}$.) By induction $b_1^* = \ldots = b_i^* = 0$. Since $b_1 = b_1^*$ we also see that $b_1 = \ldots = b_i = 0$. The number of vectors among $\{b_1^*, \ldots, b_n^*\}$ which are zero is $m := n - \dim_{\mathbb{R}}(span_{\mathbb{R}}(b_1, \ldots, b_n))$. We have shown that the first m vectors of the output are zero and the remaining ones generate L. Hence, the remaining $n - m$ vectors must be independent and generate L. Therefore, L is a lattice of rank $n - m$ with the lattice basis $\{b_{m+1}, \ldots, b_n\}$.

Now let $B = (b_1, \ldots, b_n)$ be the input vectors and $B' = (b_1', \ldots, b_n')$ the output. Let $M \in mat_{nn}(\mathbb{Z})$ describe the relation between the B' and the B.

$$(b_1', \ldots, b_n')^T = M(b_1, \ldots, b_n)^T$$

M is computed from I by simple row operations which only change the sign of the determinant. Consequently, M is invertible and by Cramer's rule $M^{-1} \in mat_{nn}(\mathbb{Z})$. The first m rows of M must be in $\ker(A)$ where A is the matrix with columns $b_1, \ldots, b_n$. The dimension of $ker(A)$ is $dim_{\mathbb{R}}(\mathbb{R}^n) - dim_{\mathbb{R}}(im(A)) = n - dim_{\mathbb{R}}(span_{\mathbb{R}}(b_1, \ldots, b_n)) = m$ and we conclude that $span(m$ first rows of $M) = ker(A)$. This means that we have a method for finding the kernel of A. Since $M^{-1} \in mat_{nn}(\mathbb{Z})$ it is not just a basis we have found but a lattice basis of $\ker(A) \cap \mathbb{Z}^n$ (and we have shown that this is a lattice).

We have shown:

Proposition 4.13 *Let $n \in \mathbb{N}$ and $b_1, \ldots, b_n \in \mathbb{Z}^n$. The set $L = \{a_1 b_1 + \ldots + a_n b_n | a_1, \ldots, a_n \in \mathbb{Z}\}$ is a lattice of dimension $dim_{\mathbb{R}}(span_{\mathbb{R}}(b_1, \ldots, b_n))$.*

Proposition 4.14 *Let $n \in \mathbb{N}$ and $A \in mat_{nn}(\mathbb{Z})$. A defines a linear map $A : \mathbb{R}^n \to \mathbb{R}^n$. $ker(A) \cap \mathbb{Z}^n$ is a lattice of dimension $n - rank(A)$.*

And we have constructed the algorithm:

Algorithm 4.15
Input: *A set of vectors* $B = (b_1, \ldots, b_n)$ *in* $\mathbb{Z}^n$ *generating a lattice* L *of rank* k
Output: *A set of vectors* $B' = (b'_1, \ldots, b'_n)$ *and a matrix* $M \in mat_{nn}(\mathbb{Z})$ *which describes* B' *in terms of* B*:* $(b'_1, \ldots, b'_n)^T = M(b_1, \ldots, b_n)^T$*.* $\{b'_1, \ldots, b'_{n-k}\}$ *are zero.* $\{b'_{n-k+1}, \ldots, b'_n\}$ *is an LLL-reduced lattice basis of* L*. The* $n-k$ *first rows of* M *is a lattice basis of* $ker(A) \cap \mathbb{Z}^n$ *where* A *is the linear map given by the matrix with columns* B*.*

Example 4.16 Let $n = 2, b_1 = (r, 0)^T, b_2 = (s, 0)^T$ for $r, s \in \mathbb{N}\backslash\{0\}$. Running the new LLL-algorithm on this input we compute:

- $\pm gcd(r, s)$
- a relation $\pm gcd(r, s) = M_{21}r + M_{22}s$ where $M \in mat_{22}(\mathbb{Z})$

Almost like the output from the Euclidian algorithm. The only difference from the Euclidean algorithm is in this case that we allow results and intermediate results to be negative.

Example 4.17 Again, let $n = 2$. What happens if we run the algorithm on the vectors $b_1 = (1, 0)^T$ and $b_2 = (\sqrt{2}, 0)^T$? b_2 is not integer as required. If the algorithm terminates we have found an integer dependence between the vectors and hence shown that $\sqrt{2}$ is a rational number. Therefore, the algorithm does not terminate. As mentioned earlier, [Lenstra,..]'s argument for termination of the original algorithm depended on the fact that $\{||x||^2 | x \in L\backslash\{0\}\}$ had a minimal element. However, this is not the case when $b_1 = (1, 0)^T$ and $b_2 = (\sqrt{2}, 0)^T$, so at least this part of the argument does not apply.

Remark 4.18 The versions of the LLL-algorithm presented here need the number of given vectors to be equal to the dimension of $\mathbb{R}^n$. However, it is easy to see that this is an unnecessary restriction and after doing a few adjustments we may use the algorithms even though the number of vectors is not n.

Example 4.19 Let us see how we use the LLL-algorithm when computing a generating set for a toric ideal using algorithm 3.18.Let $A = \begin{pmatrix} 7 & 9 & 3 & 4 \\ 8 & 7 & 7 & 4 \\ 15 & 16 & 10 & 8 \end{pmatrix}$
We run the LLL-algorithm 4.15 on the columns of A. We get the identity:

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ -1 & 0 & -1 \\ 1 & -1 & 0 \end{pmatrix} = \begin{pmatrix} 42 & -25 & -23 & 0 \\ -88 & 52 & 48 & 1 \\ -7 & 4 & 4 & 0 \\ -8 & 5 & 4 & 0 \end{pmatrix} \cdot \begin{pmatrix} 7 & 8 & 15 \\ 9 & 7 & 16 \\ 3 & 7 & 10 \\ 4 & 4 & 8 \end{pmatrix}$$

where the rows in the matrix to the right are the input and the rows in the matrix to the left are the output. The matrix in the middle describes their relation. The

dimension of $ker(A)$ is 2. A lattice basis for the lattice kernel of A is the rows of $\begin{pmatrix} 42 & -25 & -23 & 0 \\ -88 & 52 & 48 & 1 \end{pmatrix}$. Running LLL once again we find an LLL-reduced lattice basis for the lattice kernel of A. It is the rows of: $\begin{pmatrix} -4 & 2 & 2 & 1 \\ -2 & -3 & -1 & 11 \end{pmatrix}$ corresponding to the pure binomials: $b^2c^2d - a^4$ and $d^{11} - a^2b^3c$ in $k[a, b, c, d]$. We now do the four iterations in algorithm 3.18 by four Gröbner basis computations using the saturating Buchberger algorithm. The resulting Gröbner basis for I_A consists of 3 elements: $\{b^5c^3 - a^2d^{10}, a^2b^3c - d^{11}, a^4 - b^2c^2d\}$.

5 Convex polyhedral sets

This section is a short introduction to the theory of convex polyhedral sets. Some proofs have been left out.

5.1 Convex polyhedral sets

We start by defining the affine hyperplanes, closed half spaces and polyhedral sets:

Definition 5.1 *An affine hyperplane in* $\mathbb{R}^n$ *is a set of the form* $H_{a\alpha} = \{x \in \mathbb{R}^n | a \cdot x = \alpha\}$ *where* $a \in \mathbb{R}^n \backslash \{0\}$ *and* $\alpha \in \mathbb{R}$.

Definition 5.2 *A closed half space of* $\mathbb{R}^n$ *is a subset of* $\mathbb{R}^n$ *of the form* $H^{\geq}_{a\alpha} = \{x \in \mathbb{R}^n : a \cdot x \geq \alpha\}$ *where* $a \in \mathbb{R}^n \backslash \{0\}$ *and* $\alpha \in \mathbb{R}$.

Definition 5.3 *A subset* K *of* $\mathbb{R}^n$ *is called a polyhedral set if it can be written as a finite intersection of closed half spaces.* $K = \bigcap_{i=1}^{d} H^{\geq}_{a_i \alpha_i}$.

We use $H^{>}_{a\alpha}$ to denote the open half-space $H^{\geq}_{a\alpha} \backslash H_{a\alpha}$.

The polytopes are a special kind of convex polyhedral sets. We have two ways of characterising them:

Definition 5.4 *A bounded polyhedral set is called a polytope.*

Theorem 5.5 *A set is a polytope if and only if it can be written as the convex hull of a finite set of vectors.*

A proof is given in [Grünbaum].

The cones are another interesting kind of convex polyhedral sets:

Definition 5.6 *A polyhedral set* K *is called a cone if it can be written as* $K = \bigcap_{i=1}^{d} H^{\geq}_{a_i 0}$ *for some* $a_i \in \mathbb{R}^n \backslash \{0\}$.

5.2 Faces

Definition 5.7 *Let* K *be a polyhedral set and let* $H^{\geq}_{a\alpha}$ *be a closed half space containing* K. $K \subset H^{\geq}_{a\alpha}$. *The intersection* $K \cap H_{a\alpha}$ *is called a face of* K. $\emptyset$ *and* K *are also called faces of* K. *The set of faces of* K *is denoted by* $\mathcal{F}(K)$.

Notice that a face of a polyhedral set is a polyhedral set. Another way of thinking of a face is as the points in the polyhedral set where a linear function is maximised. This explains the notation:

Definition 5.8 *Let* K *be a polyhedral set in* $\mathbb{R}^n$ *and let* $\omega \in \mathbb{R}$. *We define*

$$face_\omega(K) = \{x \in K | \omega \cdot x \geq \omega \cdot y \ \forall y \in K\}.$$

Proposition 5.9 *$face_\omega(K)$ is a face of K for all $\omega \in \mathbb{R}^n$. If a face F of K is non-empty it can be written as $F = face_\omega(K)$ for some $\omega \in \mathbb{R}^n$.*

Proof. We want to prove that $face_\omega(K)$ is a face of K. If $face_\omega(K) = \emptyset$ this is true by definition. If $\omega = 0$, $face_\omega(K) = K \in \mathcal{F}(K)$. Assume that $face_\omega(K) \neq \emptyset$ and $\omega \neq 0$. This means that ω assumes its maximum on K. Let α be this maximum. We claim that $face_\omega(K) = K \cap H_{\omega\alpha} = K \cap H_{-\omega-\alpha}$. The inclusion $\subset$ is satisfied since ω is constant on $face_\omega(K)$. The other inclusion is satisfied since ω assumes its maximum α on $H_{\omega\alpha}$. $K \subset H^{\geq}_{-\omega-\alpha}$. Hence, $face_\omega(K)$ is a face of K.

Let $F \subset K$ be a non empty face of K. If $K = F$ we see that $face_0(K) = F$. If $F \neq K$ there exist ω and α such that $F = K \cap H_{\omega\alpha}$ and $K \subset H^{\geq}_{\omega\alpha}$. We claim that $face_{-\omega}(K) = F$. Since $f \neq \emptyset$ $-\omega$ assumes its maximum $-\alpha$ on K and both $face_{-\omega}(K)$ and $F = K \cap H_{\omega\alpha}$ are precisely the elements in K assuming this maximum. □

The faces of a polytope are described in a simple way:

Lemma 5.10 *Let $F = face_\omega(K)$ be a non-empty face of a polytope K. By theorem 5.5 may write this as $F \in \mathcal{F}(conv(V)) \subset \mathbb{R}^n$ with $V = \{v_1, \ldots, v_t\} \subset \mathbb{R}^n$ and $K = conv(V)$. Let $U \subset V$ be the vertices of V in which ω is maximised. $F = conv(U)$.*

Proof. Since F is non empty ω assumes its maximum on K in some point p. This maximum is also assumed in at least one point of V since we are unable write p as a convex combination of the V's and get a larger value of ω in p than in V. We claim that $face_\omega(K) = conv(U)$. The inclusion $\supset$ is clear. For the other inclusion let q be a point in $face_\omega(K)$. q can be written as a convex combination of the $V's$. Suppose that one of the coefficients in front of the $V \backslash U$'s was different from 0 then by linearity of ω the value of ω in p had to be less than the maximum. This is a contradiction and q can be written as a convex combination of the U's. □

The faces are categorised by their dimension and three types of faces are of particular interest:

Definition 5.11 *Let $S \neq \emptyset$ be a subset of $\mathbb{R}^n$. The dimension $dim(S)$ of S is the smallest number $d \in \mathbb{N}$ such that S is contained in a d-dimensional affine subspace of $\mathbb{R}^n$.*

Definition 5.12 *Let $K \subset \mathbb{R}^n$ be a polyhedral set of dimension m. Faces of K of dimension 0 are called vertices of K, faces of dimension 1 are called edges and faces of dimension $m-1$ are called facets of K.*

Proposition 5.13 *Let $K = \bigcap_{i=1}^{t} H_{\alpha_i 0}^{\geq}$ be an n-dimensional polyhedral set in $\mathbb{R}^n$. Let $F = K \cap H_{\beta 0}$ be a facet of K with $K \subset H_{\beta 0}^{\geq}$. There exists an index i such that $H_{\beta 0} = H_{\alpha_i 0}$.*

Proof. We start by noticing that a point $p \in F$ has to be in at least one of the affine hyper planes $H_{\alpha_j 0}$. Otherwise there would be an open ball around $p \in \bigcap_{i=1}^{t} H_{a_i 0}^{>} \subset K$ contained in K contradicting that β assumes its maximum in K at p.

Suppose that the claimed index i did not exist. For each $j = 1, \ldots, t$ we get $F \not\subset H_{\alpha_j 0}$ since $F \subset H_{\alpha_j 0}$ would imply that $H_{\beta 0} = H_{\alpha_j 0}$ as they contain the same $n-1$-dimensional subset. Choose $p_j \in F \backslash H_{\alpha_j 0}$. Define $p := \frac{1}{t}(p_1 + \ldots + p_t)$. $p \in F$ and p is in none of the planes $H_{\alpha_j 0}$ for any j. This contradicts the first observation. □

Given a polytope K we associate a graph called the edge graph of K to it. The vertices of the edge graph are the vertices of K and two vertices in the edge graph are connected by an edge if and only if the corresponding vertices in K are contained in an edge of K.

5.3 Normal fans

Definition 5.14 *A polyhedral complex Δ is a finite collection of polyhedral sets of $\mathbb{R}^n$ such that:*

- $K \in \Delta \wedge F \in \mathcal{F}(K) \Rightarrow F \in \Delta$
- $K, K' \in \Delta \Rightarrow K \cap K' \in \Delta$

Definition 5.15 *A polyhedral complex Δ is a fan if all its nonempty elements are cones.*

To a polytope K we associate a fan called the normal fan of K. It is defined as follows:

Definition 5.16 *Let $K \subset \mathbb{R}^n$ be a polytope and let $F \in \mathcal{F}(K) \backslash \{\emptyset\}, K \neq \emptyset$. The normal cone of F at K is $\mathcal{N}_K(F) = \{\omega \in \mathbb{R}^n | face_\omega(K) = F\}$*

Proposition 5.17 *$\overline{\mathcal{N}_F(K)}$ the closure of a normal cone is a cone.*

Proof. By theorem 5.5 K is the convex hull of some set $V = \{v_1, \ldots, v_t\} \subset \mathbb{R}^n$. By lemma 5.10 F is the convex hull of the subset $U \subset V$. Again by lemma 5.10 a vector ω satisfies $face_\omega(K) = F$ if and only if ω selects the set U in V. This gives us the following conditions on ω:

$$\omega \cdot u \geq \omega \cdot v \quad \forall (u, v) \in U \times V \quad \text{and} \quad \omega \cdot u > \omega \cdot v \quad \forall (u, v) \in U \times (V \backslash U)$$

This corresponds to an intersection S of open and closed half spaces. $S = \mathcal{N}_F(K)$. We know that $S \neq \emptyset$ so by lemma 5.18 the closure of the intersection is a cone. □

Lemma 5.18 *Let* $M = \{u \in \mathbb{R}^n | u \cdot \alpha_i = 0 \wedge u \cdot \beta_j \leq 0 \wedge u \cdot \gamma_k < 0$ *for* $i = 1, \ldots, a, j = 1, \ldots, b, k = 1, \ldots, c\}$ *and* $K = \{u \in \mathbb{R}^n | u \cdot \alpha_i = 0 \wedge u \cdot \beta_j \leq 0 \wedge u \cdot \gamma_k \leq 0$ *for* $i = 1, \ldots, a, j = 1, \ldots, b, k = 1, \ldots, c\}$. *If* $M \neq \emptyset$ *then* $\overline{M} = K$.

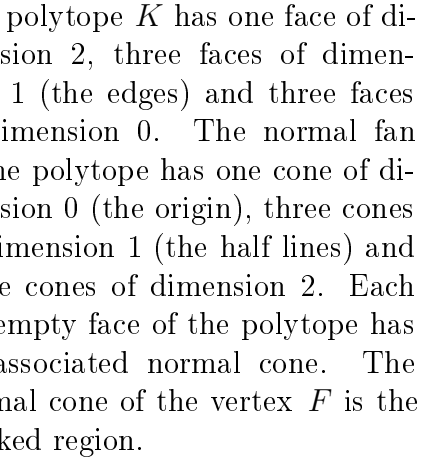
Proof. Let $v \in M$. Clearly, K is closed and contains M implying $\overline{M} \subset K$. Let $u \in K$. We just have to show that $u \in \overline{M}$. Define for $t = 1, 2, 3, \ldots$ the sequence $u_t = u + \frac{1}{t}(v - u) \to u$ for $t \to \infty$. u satisfies all equations and inequalities defining K and v satisfies all equations and equalities defining M. Hence, $u_t = \frac{1}{t}v + (1 - \frac{1}{t})u$ satisfies the equations defining M and u must be in the closure of M. □

Definition 5.19 *Let* $K \subset \mathbb{R}^n$ *be a polytope. The set of the closures of all normal cones at* K *together with the empty set is called the normal fan of* K.

Proposition 5.20 *The normal fan is a fan.*

We will not prove this.

Example 5.21



The polytope K has one face of dimension 2, three faces of dimension 1 (the edges) and three faces of dimension 0. The normal fan of the polytope has one cone of dimension 0 (the origin), three cones of dimension 1 (the half lines) and three cones of dimension 2. Each nonempty face of the polytope has an associated normal cone. The normal cone of the vertex F is the marked region.

The example suggests that there is a relation between the dimension of a face F and the dimension of its normal cone $\mathcal{N}_K(F)$. The last theorem we will need states this:

Theorem 5.22 *Let* $K \subset \mathbb{R}^n$ *be a polytope and let* M *be the map taking nonempty faces of* K *to the closed normal cones of* K*:*

$$M(F) = \overline{\mathcal{N}_K(F)} \quad \text{for} \quad F \in \mathcal{F}(K) \backslash \{\emptyset\}$$

M *maps* r *dimensional faces to* $n - r$ *dimensional cones. Two vertices are connected by an edge in the edge graph of* K *if and only if they map to closed normal cones sharing a facet. Furthermore, the edge maps to the common facet by* M.

The proof is left out. A first step is to show that given $u, v \in \mathbb{R}^n$ we have the identity $face_u face_v(K) = face_{v+\epsilon u}(K)$ for $\epsilon > 0$ sufficiently small. This will give the relation between the dimensions.

6 The Gröbner fan

In this section we will define the Gröbner fan for any polynomial ideal homogeneous with respect to a positive grading $d = (d_1, \ldots, d_n) \in \mathbb{N}^n_>$. But before we do that we have to extend the concept of initial ideals and then we will study the connection between reduced Gröbner bases and monomial initial ideals of a toric ideal homogeneous with respect to a positive grading.

We will see that when the ideal is toric the description of an n-dimensional cone in the Gröbner fan is very simple in terms of the corresponding reduced Gröbner basis. Finally, we will construct the state polytope whose normal fan is the Gröbner fan. The state polytope gives us more information on properties of the Gröbner fan which will become useful in section 7.

6.1 Lemmas in Sturmfels'

The arguments we are about to use are based on some lemmas. [Sturmfels] uses these lemmas many times. But he does not state all of them clearly. Most of them are corollaries of the following proposition based on the division algorithm and the existence of a Gröbner basis.

Definition 6.1 *Let $I \subset k[\mathbf{x}]$ be an ideal and let $\prec$ be a term order. By a standard monomial we mean a monomial in the set $std_\prec(I) := \{x^v | v \in \mathbb{N}^n \wedge x^v \notin in_\prec(I)\}$.*

Proposition 6.2 *Let $I \subset k[\mathbf{x}]$ be an ideal and let $\prec$ be a term order. We have the direct sum:*

$$k[\mathbf{x}] = I \oplus span_k(std_\prec(I))$$

meaning that (the equivalence classes of) the standard monomials form a basis for the k-vector space $k[\mathbf{x}]/I$.

Proof. Let $\mathcal{G}$ be a Gröbner basis for I with respect to $\prec$. Let $p \in k[\mathbf{x}]$. The existence of a sum $p = p_1 + p_2$ with $p_1 \in I$ and $p_2 \in span_k(std_\prec(I))$ follows from the division algorithm run on p modulo $\mathcal{G}$ with respect to $\prec$. To prove uniqueness suppose that $p_1 + p_2 = q_1 + q_2$ with $p_1, q_1 \in I$ and $p_2, q_2 \in span_k(std_\prec(I))$. Since $p_1 - q_1 \in I$ also $p_2 - q_2 \in I$. Suppose that $p_2 - q_2 \neq 0$ then the initial term of $p_2 - q_2$ must be in $in_\prec(I)$ contradicting that $p_2 - q_2 \in span_k(std_\prec(I))$. Hence, $p_2 = q_2$ and $p_1 = q_1$. □

Corollary 6.3 *Let $I \subset k[\mathbf{x}]$ be an ideal and let $\prec$ and $\prec'$ be term orders. If $in_\prec(I) \subset in_{\prec'}(I)$ then $in_\prec(I) = in_{\prec'}(I)$*

Proof. $in_\prec(I) \subset in_{\prec'}(I)$ implies $std_\prec(I) \supset std_{\prec'}(I)$. Both sets are vector space bases of $k[\mathbf{x}]/I$. Hence, $std_\prec(I) = std_{\prec'}(I)$ and $in_\prec(I) = in_{\prec'}(I)$. □

Corollary 6.4 *Let $I, J \subset k[\mathbf{x}]$ be ideals such that $I \subset J$ and let $\prec$ be a term order. If $I \neq J$ then $in_\prec(I) \neq in_\prec(J)$.*

Proof. We must prove that if $in_\prec(I) = in_\prec(J)$ then $I = J$. We know that $I \oplus span_k(std_\prec(I)) = k[\mathbf{x}] = J \oplus span_k(std_\prec(J))$. Furthermore, $std_\prec(I) = std_\prec(J)$. Let $p \in J$. Since $p \in k[\mathbf{x}]$ we can write $p = p_1 + p_2$ with $p_1 \in I$ and $p_2 \in span_k(std_\prec(I)) = span_k(std_\prec(J))$. Since $I \subset J$, $p_1 \in J$. Hence, we have written $p = p_1 + p_2$ with $p_1 \in J$ and $p_2 \in span_k(std_\prec(J))$. This can also be done by $p = p + 0$ with $p \in J$ and $0 \in span_k(std_\prec(J))$. Uniqueness implies $p = p_1 \in I$. □

Lemma 6.5 *If $J \subset k[\mathbf{x}]$ is a monomial ideal and $\prec$ is a term order then $in_\prec(J) = J$.*

One more proposition we will need is the following. A proof is given in [Sturmfels] (theorem 1.2).

Proposition 6.6 *Let $I \subset k[\mathbf{x}]$ be an ideal. The set $\{in_\prec(I) \mid \prec$ is a term order$\}$ is finite.*

6.2 Initial ideals

Earlier we have studied initial ideals with respect to term orders. Now, we extend the concept of initial ideals:

Definition 6.7 *Let $I \subset k[\mathbf{x}]$ be an ideal. Let $\omega \in \mathbb{R}^n$ and $p = \sum_{v \in \mathbb{N}^n} c_v x^v \in k[\mathbf{x}]$ where $c_v \in k$. We define $in_\omega(p)$ the initial form of p with respect to ω to be the sum of terms $c_v x^v$ with $c_v \neq 0$ for which $\omega \cdot v$ is maximal. The initial ideal of I with respect to ω is defined to be $in_\omega(I) = \langle in_\omega(q) | q \in I \rangle$.*

Definition 6.8 *A vector $\omega \in \mathbb{R}^n$ is said to be generic for an ideal I if $in_\omega(I)$ is a monomial ideal. Given I, a generic vector $\omega \in \mathbb{R}^n$ is said to represent a term order $\prec$ if $in_\prec(I) = in_\omega(I)$.*

Recall that given an ideal any term order can be represented by a vector in the following sense ([Sturmfels] proposition 1.11):

Proposition 6.9 *Let $\prec$ be a term order and $I \subset k[\mathbf{x}]$ an ideal. There exists a non-negative integer vector $\omega \in \mathbb{N}^n$ such that $in_\omega(I) = in_\prec(I)$.*

Let $\prec$ be a term order and $\omega \in \mathbb{R}^n_{\geq 0}$. Recall the term order $\prec_\omega$ defined by

$$x^v \prec_\omega x^u \Leftrightarrow \omega \cdot v < \omega \cdot u \vee (\omega \cdot v = \omega \cdot u \wedge x^v \prec x^u)$$

for $u, v \in \mathbb{N}^n$. Notice that for $f \in k[\mathbf{x}]$ we get that $in_{\prec_\omega}(f) = in_\prec(in_\omega(f))$.

The following proposition gives us a method for computing initial ideals. This method is not only of theoretical importance in proofs but will also be used in algorithm 7.1 .

Proposition 6.10 *Let $I \subset k[\mathbf{x}]$ be an ideal and let $\mathcal{G}$ be a Gröbner basis for I with respect to $\prec_u$ where $\prec$ is a term order and $u \in \mathbb{R}^n_{\geq 0}$. $\mathcal{G}' = \{in_u(g) | g \in \mathcal{G}\}$ is a Gröbner basis for $in_u(I)$ with respect to $\prec$.*

Proof. Since $\mathcal{G}$ is a Gröbner basis for I we have that $in_{\prec_u}(I) = in_{\prec_u}(\mathcal{G}) = in_{\prec}(\mathcal{G}') \subset in_{\prec} in_u(I)$. We must prove that $in_{\prec} in_u(I) \subset in_{\prec}(\mathcal{G}')$. To prove this we look at one of the generators for the left hand side $in_{\prec}(f)$ with $f \in in_u(I)$ meaning that f can be written as $f = \sum_{i=1}^{r} g_i in_u(h_i)$ with $g_i \in k[\mathbf{x}]$ and $h_i \in I$. Rewriting a couple of times we get $f = \sum_{i=1}^{m} in_u(f_i)$ with $f_i \in I$. Let $f_{i_1}, \ldots, f_{i_s}$ be the subset of polynomials such that $in_u(f_{i_1}), \ldots, in_u(f_{i_s})$ has the same u-degree as $in_{\prec}(f)$.

$$in_{\prec}(f) = in_{\prec}(\sum_{i=1}^{m} in_u(f_i)) = in_{\prec}(\sum_{j=1}^{s} in_u(f_{i_j}))$$

Applying the following lemma we continue

$$= in_{\prec} in_u(\sum_{j=1}^{s} f_{i_j}) = in_{\prec_u}(\sum_{j=1}^{s} f_{i_j})$$

Since $\sum_{j=1}^{s} f_{i_j}$ is in I, $in_{\prec}(f) \in in_{\prec_u}(I) = in_{\prec_u}(\mathcal{G}) = in_{\prec}(\mathcal{G}')$. □

Lemma 6.11 *Let $u \in \mathbb{R}^n$ and $h_1, \ldots, h_s \in k[\mathbf{x}]$ so that $in_u(h_1), \ldots, in_u(h_s)$ have the same u-degree $U \in \mathbb{R}$. If $\sum_{i=1}^{s} in_u(h_i) \neq 0$ then $\sum_{i=1}^{s} in_u(h_i) = in_u(\sum_{i=1}^{s} h_i)$.*

Proof. Summing the left hand side we get some monomials with u-degree U. Summing the sum on the right hand side the same monomials appear as a part of the sum since monomials only cancel if they have the same u-degree. The point is, the sum to the right is not 0 and there exist monomials of the maximal u-degree U. Hence, u selects exactly the terms on the left. □

Corollary 6.12 *For an ideal $I \subset k[\mathbf{x}]$, a term order $\prec$ and a vector $u \in \mathbb{R}^n_{\geq 0}$ we have $in_{\prec_u}(I) = in_{\prec}(in_u(I))$.*

Proof. Actually, we have just proven this. Let again $\mathcal{G}$ be the reduced Gröbner basis of I with respect to $\prec_u$ and $\mathcal{G}' = \{in_u(g) | g \in \mathcal{G}\}$, implying by the proposition: $in_{\prec} in_u(I) = in_{\prec}(\mathcal{G}')$. The right hand side is also equal to $in_{\prec_u}(\mathcal{G}) = in_{\prec_u}(I)$. □

Remark 6.13 In proposition 6.10 if $\mathcal{G}$ is reduced then so is $\mathcal{G}'$. $\mathcal{G}'$ is minimal because the initial terms in $\mathcal{G}'$ with respect to $\prec$ are exactly the initial terms in $\mathcal{G}$ with respect to $\prec_u$ which generates $in_{\prec_u}(I) = in_{\prec}(in_u(I))$ minimally. $\mathcal{G}'$ is reduced because the non initial terms in $\mathcal{G}'$ are non-initial terms in $\mathcal{G}$ which are not divisible by any of the minimal generators of $in_{\prec}(in_u(I))$since $\mathcal{G}$ is reduced. The coefficient of the initial terms in $\mathcal{G}'$ with respect to $\prec$ are 1 since this was the coefficient when they appeared in $\mathcal{G}$ as initial terms with respect to $\prec_u$.

6.3 Homogeneous ideals

We continue section 3.1 on homogeneous ideals.

If an ideal is homogeneous with respect to a grading then adding the grading does not change the initial ideal selected by a vector:

Proposition 6.14 *Let $I \in k[\mathbf{x}]$ be a homogeneous ideal with respect to a grading $d = (d_1, \ldots, d_n) \in \mathbb{Z}^n$ and let $\omega \in \mathbb{R}^n$ be any vector. Then $in_\omega(I) = in_{\omega+\lambda d}(I)$ for all $\lambda \in \mathbb{R}$.*

Proof. We only need to show that $in_\omega(I) \subset in_{\omega+\lambda d}(I)$ since the other inclusion follows from this inclusion by suitable choices of ω and λ. Let $f \in I$. Decompose f into its $d-$homogeneous components using lemma 3.4 $f = \sum_{i=0}^{m} f_i$ where $f_i \in I_i$. Since there is no cancellation $in_\omega(f) = \sum_{i \in J} in_\omega(f_i)$, where J is a suitable selection maximising ω. For all i, f_i is homogeneous with respect to d, implying that $in_\omega(f_i) = in_{\omega+\lambda d}(f_i) \in in_{\omega+\lambda d}(I)$. Hence, $in_\omega(f) \in in_{\omega+\lambda d}(I)$. We have shown that the generators of $in_\omega(I)$ are in $in_{\omega+\lambda d}(I)$. This completes the proof. □

Later we will see that the proposition gives us important information on the Gröbner fan which we are about to construct (example 6.20). But for now, we only need the corollary:

Corollary 6.15 *Let I be a homogeneous ideal with respect to a positive grading $d = (d_1, \ldots, d_n) \in \mathbb{N}^n_{>0}$ and let $\omega \in \mathbb{R}^n$ be a weight vector. There exists a positive vector $\omega' \in \mathbb{R}^n_{>0}$ such that $in_\omega(I) = in_{\omega'}(I)$.*

Proof. By the proposition $\omega' = \omega + \lambda d$ satisfies $in_\omega(I) = in_{\omega'}(I)$ for all $\lambda \in \mathbb{R}$. We just have to choose λ so that ω' becomes positive. This can be done since d is positive. □

6.4 Reduced Gröbner bases and monomial initial ideals

Let $I_A \subset k[\mathbf{x}]$ be a homogeneous toric ideal with respect to a positive grading $d \in \mathbb{N}^n_{>0}$. We will establish a bijection between the reduced Gröbner bases of I_A and the monomial initial ideals of I_A assuming that $char(k) \neq 2$. In the general case this is not possible as the following example shows.

Example 6.16 The ideal $\langle x + y \rangle \subset k[x, y]$ has only one reduced Gröbner basis $\{x + y\}$ but two monomial initial ideals $\langle x \rangle$ and $\langle y \rangle$. However, the ideal is not toric unless $char(k) = 2$.

The theorem we are going to prove is the following:

Theorem 6.17 *Let $I_A \subset k[\mathbf{x}]$ be a toric ideal homogeneous with respect to a positive grading. If $char(k) \neq 2$ there exists a bijection ϕ between the reduced Gröbner bases of I_A and the monomial initial ideals of I_A defined by*

$$\phi(\mathcal{G}_\prec(I_A)) = in_\prec(I_A)$$

for all term orders $\prec$.

We need the following lemma:

Lemma 6.18 *Let $I \subset k[\mathbf{x}]$ be an ideal and $\mathcal{G}_\prec(I)$ and $\mathcal{G}_{\prec'}(I)$ reduced Gröbner bases of I with respect to term orders $\prec$ and $\prec'$ respectively. If $\forall g \in \mathcal{G}_\prec(I)$: $in_\prec(g) = in_{\prec'}(g)$ then $\mathcal{G}_\prec(I) = \mathcal{G}_{\prec'}(I)$.*

Proof. We get the inclusion: $in_\prec(I) = in_\prec(\mathcal{G}_\prec(I)) = in_{\prec'}(\mathcal{G}_\prec(I)) \subset in_{\prec'}(I)$. Since an inclusion among monomial initial ideals cannot be strict (corollary 6.3) we get the equality: $in_{\prec'}(\mathcal{G}_\prec(I)) = in_{\prec'}(I)$ implying that $\mathcal{G}_\prec(I)$ is a Gröbner basis with respect to $\prec'$. If $\mathcal{G}_\prec(I)$ is reduced with respect to $\prec$ it is reduced with respect to $\prec'$. By uniqueness of reduced Gröbner bases $\mathcal{G}_\prec(I) = \mathcal{G}_{\prec'}(I)$. □

We return the proof of the theorem:

Proof. Why is ϕ well defined? $\mathcal{G}_\prec(I) = \{x^{\alpha_1} - x^{\beta_1}, \ldots, x^{\alpha_t} - x^{\beta_t}\}$ consists of pure binomials. In each binomial one term has coefficient 1 and the other term has coefficient -1. ($1 \neq -1$ if $char(k) \neq 2$.) By the definition of $\mathcal{G}_\prec(I)$ being reduced we know that the initial term of $x^{\alpha_i} - x^{\beta_i}$ with respect to $\prec$ has coefficient 1. Hence, $in_\prec(x^{\alpha_i} - x^{\beta_i}) = x^{\alpha_i}$ and ϕ is simply given by $\phi(\mathcal{G}_\prec(I)) = \phi(\{x^{\alpha_1} - x^{\beta_1}, \ldots, x^{\alpha_t} - x^{\beta_t}\}) = \langle x^{\alpha_1}, \ldots, x^{\alpha_t} \rangle$.

Surjective: Let $in_v(I_A)$ be a monomial initial ideal for $v \in \mathbb{R}^n$. Let $\prec$ be any term order. By corollary 6.15 we may assume that $v \in \mathbb{R}^n_{>0}$, implying that $\prec_v$ is a term order. $in_v(I_A) = in_\prec(in_v(I_A)) = in_{\prec_v}(I_A)$ (corollary 6.12). Hence, $\phi(\mathcal{G}_{\prec_v}(I_A)) = in_v(I_A)$.

Injective: Suppose $\phi(\mathcal{G}_\prec(I_A)) = \phi(\mathcal{G}_{\prec'}(I_A))$. That is $in_\prec(I_A) = in_{\prec'}(I_A)$. We claim that $\mathcal{G}_\prec(I_A) = \mathcal{G}_{\prec'}(I_A)$. To prove this we use lemma 6.18: Let $g \in \mathcal{G}_\prec(I_A)$. We know since $\mathcal{G}_\prec(I_A)$ is reduced that exactly one of the monomials (being $in_\prec(g)$) in g is in $in_\prec(I_A) = in_{\prec'}(I_A)$. $g \in I_A \Rightarrow in_{\prec'}(g) \in in_{\prec'}(I_A) = in_\prec(I_A)$. Hence, $in_\prec(g) = in_{\prec'}(g)$. And the lemma applies. □

Remark 6.19 Is it in general possible to construct ϕ if I is not homogeneous with respect to a positive grading?

No. The toric ideal $I = \langle x_1 x_2 - 1 \rangle \subset k[x_1, x_2]$ has three initial ideals. $\langle x_1 x_2 \rangle$ and $\langle 1 \rangle = k[x_1, x_2]$ are monomial initial ideals of I and $\langle x_1 x_2 - 1 \rangle$ is the third initial ideal. But the only reduced Gröbner basis for I is $\{x_1 x_2 - 1\}$. A matrix A defining $I = I_A$ is $A = (1 - 1) \in mat_{12}(\mathbb{Z})$.

6.5 The Gröbner fan

We now construct the Gröbner fan for an ideal I homogeneous with respect to a positive grading. $\mathbb{R}^n$ is divided into equivalence classes by the relation $r \sim s \Leftrightarrow in_r(I) = in_s(I)$ for $r, s \in \mathbb{R}^n$. The closure of an equivalence class is called a Gröbner cone. If $c \in \mathbb{R}^n$ is a vector we use $\mathcal{K}_c$ to denote the Gröbner cone that is the closure of the equivalence class of c. The Gröbner fan of I is the set of all Gröbner cones of I together with the empty set. $\mathcal{K}_c$ is called the Gröbner cone at c.

Example 6.20

The Gröbner fan for $I = \langle x_1 - x_2 \rangle \subset k[x_1, x_2]$ consists of three nonempty sets:

- $\{(x, y) \in \mathbb{R}^2 : x \geq y\}$
- $\{(x, y) \in \mathbb{R}^2 : x = y\}$
- $\{(x, y) \in \mathbb{R}^2 : x \leq y\}$

corresponding to the initial ideals $\langle x_1 \rangle$, $\langle x_1 - x_2 \rangle$ and $\langle x_2 \rangle$.

Since I is homogeneous with respect to the grading $(1, 1)$, the figure remains unchanged when translated by $(1, 1)$ by proposition 6.14. Given a matrix A, the toric ideal I_A is homogeneous with respect any vector from the row space of A. By proposition 6.14 the Gröbner fan remains unchanged when translated by a vector from the row space of A.

A priori, we do not know that the Gröbner cones are cones nor that the Gröbner fan is a fan.

Proposition 6.21 *A Gröbner cone is a cone.*

This proposition follows from the following ([Sturmfels] proposition 2.3):

Proposition 6.22 *Let $I \subset k[\mathbf{x}]$ be an ideal, $v \in \mathbb{R}^n_{>0}$ a vector, $\prec$ a term order and $\mathcal{G}_{\prec_v}(I)$ the reduced Gröbner basis corresponding to $\prec_v$. For $u \in \mathbb{R}^n_{>0}$:*

$$in_u(I) = in_v(I) \Leftrightarrow \forall g \in \mathcal{G}_{\prec_v}(I) : in_u(g) = in_v(g)$$

Proof.
$\Leftarrow$: We know that $in_{\prec_u}(g) = in_{\prec_v}(g)$ for all $g \in \mathcal{G}_{\prec_v}(I)$. Using lemma 6.18 we get $\mathcal{G}_{\prec_v}(I) = \mathcal{G}_{\prec_u}(I)$. Let us compute a Gröbner basis $\mathcal{G}$ for $in_u(I)$ with respect to $\prec$ using proposition 6.10:

$$\mathcal{G} = \{in_u(g) | g \in \mathcal{G}_{\prec_u}(I)\}$$

And now let us compute a Gröbner basis $\mathcal{G}'$ for $in_v(I)$ with respect to $\prec$:

$$\mathcal{G}' = \{in_v(g) | g \in \mathcal{G}_{\prec_v}(I)\}$$

We see that the $\mathcal{G} = \mathcal{G}'$ and therefore also that $in_u(I) = in_v(I)$.

$\Rightarrow$: $in_u(I) = in_v(I)$ implies $in_{\prec_u}(I) = in_{\prec_v}(I)$ (using corollary 6.12). Let $g \in \mathcal{G}_{\prec_v}(I)$. Since $\mathcal{G}_{\prec_v}(I)$ is reduced exactly one term (being $in_{\prec_v}(g)$) of g belongs to $in_{\prec_v}(I) = in_{\prec_u}(I)$. On the other hand $g \in I \Rightarrow in_{\prec_u}(g) \in in_{\prec_u}(I)$ so $in_{\prec_v}(g) = in_{\prec_u}(g)$. Applying lemma 6.18 we get $\mathcal{G}_{\prec_v}(I) = \mathcal{G}_{\prec_u}(I)$. Now let g be an element in the reduced Gröbner basis $\mathcal{G}_{\prec_v}(I) = \mathcal{G}_{\prec_u}(I)$. $in_{\prec_u}(g) = in_{\prec_v}(g)$. $in_u(g)$ consists of $in_{\prec_u}(g)$ and some terms h: $in_u(g) = in_{\prec_u}(g) + h$. $in_v(g)$ consists of $in_{\prec_v}(g)$ and some terms h': $in_v(g) = in_{\prec_v}(g) + h'$. We want to prove that $in_u(g) = in_v(g)$ or equivalently $h = h'$. Since the basis is reduced, no term in h lies in $in_{\prec_u}(I)$ and no term in h' lies in $in_{\prec_v}(I)$. $h - h' = in_u(g) - in_v(g) \in in_u(I) = in_v(I)$. Suppose for a moment that $h - h' \neq 0$. Then $in_{\prec}(h - h') \in in_{\prec} in_u(I) = in_{\prec_u}(I)$ (using corollary 6.12) contradicting that no terms of h and h' lies in $in_{\prec_u}(I) = in_{\prec_v}(I)$. Hence, $h = h'$ and $in_u(g) = in_v(g)$. □

The proposition does not require I to be a homogeneous ideal, but....

Remark 6.23 If I is homogeneous with respect to a positive grading the vector u in proposition 6.22 does not have to be positive but can be anywhere in $\mathbb{R}^n$. This is a consequence of proposition 6.14. Furthermore, even if $v \in \mathbb{R}^n \backslash \mathbb{R}^n_{>0}$ we can use the proposition with $\mathcal{G}_{\prec_{v'}}(I)$ instead of $\mathcal{G}_{\prec_v}(I)$ where v' is a positive vector equivalent to v.

Proposition 6.22 and the remark tell us that the equivalence classes of a vector v is described by a finite set of inequalities and equations. The inequalities are of the form $u \cdot \gamma_i < 0$ and the equations are of the form $u \cdot \alpha_i = 0$ where $\alpha_i, \gamma_i \in \mathbb{R}^n$. Since an equivalence class is non-empty, lemma 5.18 tells us that its closure is a cone. This completes the proof of proposition 6.21.

Remark 6.24 The monomial initial ideals correspond exactly to the cones of dimension n. If $in_v(I)$ is monomial the set $\mathcal{G}'$ in the proof of proposition 6.22 consists of monomials only by remark 6.13. This implies that only inequalities appear in the description of the equivalence class of v. Hence, the equivalence class is open in $\mathbb{R}^n$ and $\mathcal{K}_v$ must be of dimension n.

On the other hand if $\mathcal{K}_v$ is of dimension n then so is the equivalence class of v. All equalities appearing in the description of an equivalence class are nontrivial so we cannot have any equalities in the description of the equivalence class of v, implying that $\mathcal{G}'$ is a set of monomial generators for $in_v(I)$.

It still remains to be shown that the Gröbner fan is a fan. I will not prove this but refer to [Sturmfels].

Theorem 6.25 *The Gröbner fan is a fan.*

6.6 Facets of toric Gröbner cones

Even though we will not prove theorem 6.25 we will carefully study the Gröbner cones corresponding to monomial initial ideals of toric ideals. It turns out that these cones are very simple to describe in terms of the corresponding Gröbner basis.

Let us look at an n-dimensional Gröbner cone for a toric ideal I again homogeneous with respect to a positive grading and with $char(k) \neq 2$. By remark 6.24 we are looking at the closure of an equivalence class corresponding to a monomial initial ideal $in_v(I)$ with $v \in \mathbb{R}^n$. Using the bijection ϕ we have a reduced Gröbner basis $\mathcal{G}_\prec(I) = \{x^{\alpha_1} - x^{\beta_1}, \ldots, x^{\alpha_t} - x^{\beta_t}\}$ such that $in_\prec(I) = in_v(I)$. Which n-dimensional Gröbner cone we are looking at is determined by any of $v, \prec$ or $\mathcal{G}_\prec(I)$ alone.

By corollary 6.15 we may assume that $v \in \mathbb{R}^n_{>0}$ since I is homogeneous with respect to a positive vector. This means that $\prec_v$ is a term order. By proposition 6.22 we know that the equivalence class of v is given by:

$$\{\omega \in \mathbb{R}^n | in_\omega(g) = in_v(g) \text{ for all } g \in \mathcal{G}_{\prec_v}(I)\}$$

Observe $in_{\prec_v}(I) = in_\prec(in_v(I)) = in_\prec(in_\prec(I)) = in_\prec(I)$ so since ϕ is a bijection $\mathcal{G}_{\prec_v}(I) = \mathcal{G}_\prec(I)$. Furthermore, $in_v(x^{\alpha_i} - x^{\beta_i}) = x^{\alpha_i}$ since taking initial forms of the elements in $\mathcal{G}_{\prec_v}(I)$ will give the minimal generators of $in_v(I) = in_\prec(I)$ by remark 6.13. We may rewrite the equivalence class as:

$$= \{\omega \in \mathbb{R}^n | \omega \cdot \alpha > \omega \cdot \beta \text{ for all } x^\alpha - x^\beta \in \mathcal{G}_\prec(I)\}$$

Using lemma 5.18 we get

$$\mathcal{K}_v = \{\omega \in \mathbb{R}^n | \omega \cdot \alpha \geq \omega \cdot \beta \text{ for all } x^\alpha - x^\beta \in \mathcal{G}_\prec(I)\}$$

with each pure binomial in $\mathcal{G}_\prec(I)$ giving one inequality.

Some of the binomials are irrelevant in the sense that their inequalities can be left out in the description of $\mathcal{K}_v$. Other binomials are essential. The essential binomials are called facet binomials. That is, a binomial $x^{\alpha_i} - x^{\beta_i}$ is facet if

$$\neg(\omega \cdot \alpha \geq \omega \cdot \beta \text{ for all } x^\alpha - x^\beta \in \mathcal{G}_\prec(I) \backslash \{x^{\alpha_i} - x^{\beta_i}\} \Rightarrow \omega \cdot \alpha_i \geq \omega \cdot \beta_i)$$

The reason that we use the term "facet" for a binomial will become clear in the following. Let $x^{\alpha_i} - x^{\beta_i}$ be a facet binomial. The half space $H^{\geq}_{(\alpha_i - \beta_i)0}$ contains $\mathcal{K}_v$ implying that $H_{(\alpha_i - \beta_i)0} \cap \mathcal{K}_v$ is a face of $\mathcal{K}_v$ by definition. To see that it has dimension $n-1$ we observe that $x^{\alpha_i} - x^{\beta_i}$ being a facet binomial implies the existence of a vector $c' \in \mathbb{R}^n$ such that $c' \cdot \alpha_j \geq c' \cdot \beta_j$ for $j \neq i$ and $c' \cdot \alpha_i < c' \cdot \beta_i$. I is homogeneous with respect to a positive vector implying that the elements in the reduced Gröbner basis are homogeneous too with respect to this vector (section 3.1). Adding this positive vector to c' does not change the inequalities

so with out loss of generality we may assume that $c' \in \mathbb{R}^n_{>0}$. Combining v and c' we get a vector $c \in \mathbb{R}^n_{>0}$ satisfying

$$c \cdot \alpha_j > c \cdot \beta_j \text{ for } j \neq i \text{ and } c \cdot \alpha_i = c \cdot \beta_i$$

. The set

$$\{\omega \in \mathbb{R}^n | \omega \cdot \alpha > \omega \cdot \beta \text{ for all } x^\alpha - x^\beta \in \mathcal{G}_\prec(I) \backslash \{x^{\alpha_i} - x^{\beta_i}\}\}$$

is open implying the existence of a ball around c contained in the set. This ball intersected with the hyperplane $H_{(\alpha_i - \beta_i)0}$ is $n-1$-dimensional and contained in the face. Hence the face $H_{(\alpha_i-\beta_i)0} \cap \mathcal{K}_v$ is a facet.

Theorem 6.25 suggests that the facet of $\mathcal{K}_v$ itself is a Gröbner cone. We will prove this. By lemma 6.18 $\mathcal{G}_{\prec_c}(I) = \mathcal{G}_\prec(I)$. Using proposition 6.22 and lemma 5.18 we get that the closure of the equivalence class of c is given by

$$\mathcal{K}_c = \{\omega \in \mathbb{R}^n | \omega \cdot \alpha_j \geq \omega \cdot \beta_j \text{ for } j \neq i \text{ and } \omega \cdot \alpha_i = \omega \cdot \beta_i\}$$

This is exactly $H_{(\alpha_i-\beta_i)0} \cap \mathcal{K}_v$.

The projective drawing shows the locations of the points v, v', c and c'. In this example $\mathcal{G}_\prec(I)$ has three facet binomials. The forth plane corresponds to a facet binomial in $\mathcal{G}_{\prec'}(I)$.

$\mathcal{G}_{\prec_c}(I) = \mathcal{G}_\prec(I)$ implies $in_v(I) = in_\prec(I) = in_{\prec_c}(I) = in_\prec(in_c(I))$ so $in_v(I)$ is a monomial initial ideal of $in_c(I)$. Furthermore, a reduced Gröbner basis of $in_c(I)$ with respect to $\prec$ is $\{x^{\alpha_i} - x^{\beta_i}, x^{\alpha_1}, \ldots, x^{\alpha_{i-1}}, x^{\alpha_{i+1}}, \ldots, x^{\alpha_t}\}$ by proposition 6.10 and remark 6.13. Actually $in_c(I)$ has only two monomial initial ideals:

Proposition 6.26 *Let $\{x^{\alpha_i} - x^{\beta_i}, x^{\alpha_1}, \ldots, x^{\alpha_{i-1}}, x^{\alpha_{i+1}}, \ldots, x^{\alpha_t}\}$ be a reduced Gröbner basis with respect to a term order $<$ for an ideal J in $k[\mathbf{x}]$ homogeneous with respect to a positive vector (with $char(k) \neq 2$). J has exactly two reduced Gröbner bases and two monomial initial ideals. The other reduced Gröbner basis contains the binomial $x^{\beta_i} - x^{\alpha_i}$.*

Proof. J is not a monomial ideal since it has a reduced Gröbner basis containing a binomial. To prove that J has exactly two reduced Gröbner basis we observe that computing a Gröbner basis for J with respect to some term order starting with the basis given we will never introduce any new binomials but only monomials. S-polynomials are either 0 or a monomial. The same is true for an intermediate polynomial in the division algorithm. Only when reducing the basis we might have to change the sign of the binomial to make the coefficient of the initial term 1. This could introduce the binomial $x^{\beta_i} - x^{\alpha_i}$. The Gröbner basis computation is completely determined by the relation between x^{β_i} and x^{α_i} with respect to the term order. So there are at most two reduced Gröbner bases for J.

Since J is not a monomial ideal a reduced Gröbner basis of J must contain $x^{\alpha_i} - x^{\beta_i}$ or $x^{\beta_i} - x^{\alpha_i}$ but not both and not other binomials. Since $\alpha_i \neq \beta_i$ the vector $\beta_i - \alpha_i$ separates the two points α_i and β_i and since J is homogeneous with respect to a positive vector we get a vector $w \in \mathbb{R}^n_{>0}$ inducing a term order $<'_w$ with $x^{\alpha_i} <'_w x^{\beta_i}$ where $<'$ can be any term order. Hence, both reduced Gröbner bases exists. Computing initial ideals with respect to the two term orders $<$ and $<'_w$ by taking out initial terms of the Gröbner bases gives us the two possible monomial initial ideals of J. Only these two monomial initial ideals exist since any monomial initial ideal of J is represented by a vector and thereby by a positive vector and by a term order with either x^{α_i} larger than x^{β_i} or x^{β_i} larger than x^{α_i}. □

The second monomial initial ideal of $in_c(I)$ is $in_{\prec_{c'}} in_c(I)$ as $x^{\alpha_i} \prec_{c'} x^{\beta_i}$. $in_{\prec_{c'}} in_c(I) = in_{\prec_{c'c}}(I)$ and is also a monomial initial ideal of I. Let $\prec'$ be a term order such that $\mathcal{G}_{\prec'}(I)$ is the Gröbner basis corresponding to this monomial initial ideal and let v' be an equivalent vector. That is, $in_{\prec'}(I) = in_{v'}(I) = in_{\prec_{c'}} in_c(I)$. Notice that this initial ideal is independent of the choices of c, c' and $\prec'$ and so are $\mathcal{G}_{\prec'}(I)$ and $\mathcal{K}_{v'}$.

We will say that the Gröbner cone $\mathcal{K}_{v'}$ (or basis $\mathcal{G}_{\prec'}(I)$) is adjacent to the Gröbner cone $\mathcal{K}_v$ (or basis $\mathcal{G}_{\prec}(I)$) through the facet binomial $x^{\alpha_i} - x^{\beta_i}$. We will denote the basis adjacent to $\mathcal{G}_{\prec}(I)$ through $x^{\alpha_i} - x^{\beta_i}$ by $flip(\mathcal{G}_{\prec}(I), x^{\alpha_i} - x^{\beta_i})$. The relation "is adjacent to" is symmetric in the sense of the following proposition. This implies the existence of a graph called the graph of I with vertices being the reduced Gröbner bases of I and two bases being connected if they are adjacent through some facet binomial. An example of the graph of a toric ideal is given in section 7.6.

Proposition 6.27 *$x^{\beta_i} - x^{\alpha_i}$ is a facet binomial of $flip(\mathcal{G}_{\prec}(I), x^{\alpha_i} - x^{\beta_i})$ and $flip(flip(\mathcal{G}_{\prec}(I), x^{\alpha_i} - x^{\beta_i}), x^{\beta_i} - x^{\alpha_i}) = \mathcal{G}_{\prec}(I)$.*

Proof. $in_{\prec'_c}(I) = in_{\prec'}(in_c(I))$ is a monomial initial ideal of $in_c(I)$ with $x^{\alpha_i} \prec' x^{\beta_i}$ so it must be equal to $in_{\prec'}(I)$. By the bijection ϕ we get $\mathcal{G}_{\prec'}(I) = \mathcal{G}_{\prec'_c}(I)$. Another way of finding a reduced Gröbner basis for $in_c(I)$ with respect to $\prec'$ is

by taking initial forms with respect to c of the elements in $\mathcal{G}_{\prec'_c}(I)$. Since $x^{\beta_i} - x^{\alpha_i}$ appears in the resulting Gröbner basis it must be in $\mathcal{G}_{\prec'_c}(I) = \mathcal{G}_{\prec'}(I)$.

Since c selects only one binomial in $\mathcal{G}_{\prec'}(I)$ we get the inequalities $c \cdot \alpha > c \cdot \beta$ for all $x^\alpha - x^\beta \in \mathcal{G}_{\prec'}(I) \backslash \{x^{\beta_i} - x^{\alpha_i}\}$ and $c \cdot \alpha_i = c \cdot \beta_i$. Since v' satisfies $v' \cdot \alpha > v' \cdot \beta$ for all $x^\alpha - x^\beta \in \mathcal{G}_{\prec'}(I)$ we may select a sufficiently small $\delta > 0$ such that the vector $w := c - \delta v'$ satisfies $w \cdot \alpha > w \cdot \beta$ for all $x^\alpha - x^\beta \in \mathcal{G}_{\prec'}(I) \backslash \{x^{\beta_i} - x^{\alpha_i}\}$ and $w \cdot \alpha_i > w \cdot \beta_i$. Hence, $x^{\beta_i} - x^{\alpha_i}$ is a facet binomial in $\mathcal{G}_{\prec'}(I)$.

To find $flip(flip(\mathcal{G}_\prec(I), x^{\alpha_i} - x^{\beta_i}), x^{\beta_i} - x^{\alpha_i})$ we have to choose two vectors $c'_{new}, c_{new} \in \mathbb{R}^n$ according to the definition. The vectors $c'_{new} := w$ and $c_{new} := c$ satisfy the inequalities they are supposed to (see the paragraph above). So $flip(flip(\mathcal{G}_\prec(I), x^{\alpha_i} - x^{\beta_i}), x^{\beta_i} - x^{\alpha_i})$ is the reduced Gröbner basis of I corresponding to the other monomial initial ideal of $in_c(I)$ being $in_\prec(I)$. Hence $flip(flip(\mathcal{G}_\prec(I), x^{\alpha_i} - x^{\beta_i}), x^{\beta_i} - x^{\alpha_i})$ is in fact $\mathcal{G}_\prec(I)$. □

In section 7.4 we will see that any facet of $\mathcal{K}_v$ comes from a facet binomial (remark 7.9).

6.7 The state polytope

Let I be a homogeneous ideal with respect to the positive grading $(d_1, \ldots, d_n) \in \mathbb{N}^n_{>0}$. The purpose of this section is to construct a polytope whose normal fan is the Gröbner fan of I. This polytope is called the state polytope of I. We will not prove that its normal fan is the Gröbner fan. A proof is given in [Sturmfels].

Definition 6.28 *Let $K, K' \subset \mathbb{R}^n$ be two polytopes. We define the Minkowski sum of K and K' to be $K + K' := \{p + p' | p \in K, p' \in K'\}$.*

Using theorem 5.5 we get:

Proposition 6.29 *The Minkowski sum of two polytopes is a polytope.*

Let M be a monomial ideal and $d \in \mathbb{N}$. $\sum M_d$ denotes the sum of all vectors $a \in \mathbb{N}^n$ such that $x^a \in M$ and has degree d (with respect to the grading $(d_1, \ldots, d_n) \in \mathbb{N}^n$).

Define $state_d(I) = conv\{\sum in_\prec(I)_d\}_\prec$ where $\prec$ runs through all term orders. $state_d(I)$ is a polytope because there are only finitely many initial ideals. The state polytope is now defined to be the Minkowski sum $state(I) = \sum_{d=1}^{D} state_d(I)$ where D is the largest degree of the polynomials in any reduced Gröbner basis of I. D is well defined since it is the largest degree of any of the minimal generators for a monomial initial ideal of I and there are only finitely many initial ideals of I.

Theorem 6.30 *Let I be a homogeneous ideal with respect to a positive grading $(d_1, \ldots, d_n) \in \mathbb{N}^n_{>0}$. The normal fan of the state polytope of I is the Gröbner fan of I.*

The Gröbner fan given as the normal fan of the state polytope gives us a good intuitive understanding of the Gröbner fan. The edge graph of the state polytope is isomorphic to the graph of I when I is toric: two vertices of the state polytope are connected by an edge if and only if their normal cones share a facet (theorem 5.22) which again is the definition for the corresponding reduced Gröbner bases to be connected in the graph of I (proposition 5.13). However, we will only need theorem 6.30 in one future proof being the proof that the corresponding graph can be oriented in a simple way without cycles (proposition 7.10 and its lemma). Together with the fact that the oriented graph only has one sink and the fact that there are only finitely many monomial initial ideals this in turn implies that the graph is connected.

7 Computing the Gröbner fan

Representing the n-dimensional cones of a Gröbner fan of a toric ideal I by reduced Gröbner bases we are almost ready to compute the Gröbner fan by traversing the graph of I or equivalently by traversing the edge graph of the state polytope of I. Actually only two things are missing:

- Given a reduced Gröbner basis, how do we find its facet binomials?
- Given a reduced Gröbner basis and a facet binomial, how do we compute the adjacent Gröbner basis?

After having studied these two problems we present a graph traversal algorithm doing an exhaustive search. The exhaustive search algorithm has the disadvantage that at any point it needs to store all previously computed Gröbner bases to traverse the state polytope correctly.

We will also see how we can search for a reduced Gröbner basis with respect to a specified term order starting with any reduced Gröbner basis by walking in the graph of I. This is called the Gröbner walk. The paths we walk along searching for a Gröbner basis with respect to a term order starting at any reduced Gröbner basis form a tree called the search tree. A better way of computing the entire Gröbner fan is by doing a tree traversal of the search tree. This method is called the reverse search method.

In this section we will assume that the ideal $I \subset k[\mathbf{x}]$ is toric and homogeneous with respect to a positive grading and that $char(k) \neq 2$.

7.1 Algorithm flip

In this section we will study an algorithm given by [Huber,..]. Given a reduced Gröbner basis $\mathcal{G}$ for I and a facet binomial $x^{\alpha_i} - x^{\beta_i} \in \mathcal{G}$ it computes a reduced Gröbner basis $\mathcal{G}'$ of I with the property that the Gröbner cone at $\mathcal{G}'$ share a facet with the Gröbner cone at $\mathcal{G}$. The facet is the facet determined by the facet binomial. That is, the algorithm computes $\mathcal{G}' = flip(\mathcal{G}, x^{\alpha_i} - x^{\beta_i})$. The algorithm is a special case of a general algorithm given by [Sturmfels] (subroutine 3.7). However, the algorithm given by [Huber,..] takes advantage of the fact that I is toric and furthermore the algorithm never needs to know which term order $\mathcal{G}$ is a Gröbner basis for.

Algorithm 7.1 *Flip*
Input: *The reduced Gröbner basis* $\mathcal{G} = \{x^{\alpha_j} - x^{\beta_j} : j = 1, \ldots, t\}$ *for a toric ideal* I *with respect to some (unknown) term order* $\prec$ *and an index* i *such that* $x^{\alpha_i} - x^{\beta_i}$ *is a facet binomial of* $\mathcal{G}$.
Output: $flip(\mathcal{G}, x^{\alpha_i} - x^{\beta_i})$
{

$Old := \{x^{\alpha_i} - x^{\beta_i}\} \cup \{x^{\alpha_j}, j \neq i\}$;
$Temp := \{x^{\beta_i} - x^{\alpha_i}\} \cup \{x^{\alpha_j}, j \neq i\}$;
$New := Buchberger(Temp)$;
$\mathcal{G}'' := \{x^{\beta_i} - x^{\alpha_i}\} \cup \{h - h', h \in New \text{ a monomial}\}$;
reduce $\mathcal{G}''$ *to get* $\mathcal{G}'$;
}

The algorithm is explained in the following:

Most of the work has already been done in section 6.6. With the notation introduced in that section we notice that Old is the reduced Gröbner basis of $in_c(I)$ with respect to $\prec$. The other reduced Gröbner basis of $in_c(I)$ is computed by making x^{β_i} larger than x^{α_i} as suggested by $Temp$ and running Buchberger's algorithm on $Temp$. This can be done with no further information on the term order as explained in section 6.6. We store the reduced Gröbner basis in New.

For each monomial $h \in New$ let h' be the remainder when dividing h modulo $\mathcal{G}$ with respect to $\prec$. Clearly, $h - h' \in I$. Futhermore, $h \neq h'$ since $h \in in_c(I)$ and at least one initial term of Old divides h since Old is a Gröbner basis of $in_c(I)$ so atleast one initial term in $\mathcal{G}$ must divide h. $c \cdot \alpha_j > c \cdot \beta_j$ for $j \neq i$ and $c \cdot \alpha_i = c \cdot \beta_i$ guarantees that $in_c(h - h') = h$. (At least once we must use $x^{\alpha_j} - x^{\beta_j}$ with $j \neq i$ in the division. Otherwise, we would end up with $h - (x^{\alpha_i} - x^{\beta_i})g \notin in_\prec(\mathcal{G}) = in_\prec(I)$ for some $g \in k[\mathbf{x}]$. But $h - (x^{\alpha_i} - x^{\beta_i})g \in in_c(I)$ implying $in_\prec(h - (x^{\alpha_i} - x^{\beta_i})g) \in in_\prec(in_c(I)) = in_{\prec_c}(I) = in_\prec(I)$. This is a contradiction.) Hence, $in_{\prec'_c}(h - h') = in_{\prec'}(in_c(h - h')) = in_{\prec'}(h) = h$ implying $in_{\prec'_c}(\mathcal{G}'') = in_{\prec'}(New) = in_{\prec'}(in_c(I)) = in_{\prec'_c}(I)$. $\mathcal{G}''$ must be a basis for I with respect to $\prec'_c$ and reducing we get $\mathcal{G}' = \mathcal{G}_{\prec'_c}(I)$. (Knowing the initial terms is sufficient to do the reduction.) Using ϕ we get $\mathcal{G}' = \mathcal{G}_{\prec'}(I)$ as $in_{\prec'_c}(I) = in_{\prec'}(I)$ by the proof of proposition This completes the algorithm.

Remark 7.2 A very important property of the algorithm is that we never need to represent any term orders or weight vectors. The algorithm operates on monomials and binomials only.

7.1.1 Implementation details

Just like the Buchberger algorithm for binomials in toric ideals the Buchberger algorithm used in flip (see proof of proposition 6.26) can be transformed to an algorithm working on vectors alone. In this section we will see how we can represent the polynomials and which operations are needed.

A polynomial in the algorithm is either a pure binomial (including zero) or a monomial. A pure binomial $x^\alpha - x^\beta$ is represented by the vector $\alpha - \beta$ and a monomial x^α is represented by the vector α for $\alpha, \beta \in \mathbb{N}^n$.

The division algorithm: The input for the division algorithm is some monomials and possibly one pure binomial.

- When a monomial x^v is reduced by a monomial x^u the result is 0.
- When a binomial $x^\alpha - x^\beta$ with x^α as initial term is reduced by a monomial x^u the result is $-x^\beta$. Since we are not interested in signs in Buchberger's algorithm we represent the result by the vector β (monomial x^β). However, this can never happen since $\langle Old \rangle$ is not a monomial ideal.
- When a monomial x^v is reduced by a binomial $x^\alpha - x^\beta$ with x^α as initial term the result is the monomial $x^v - \frac{x^v}{x^\alpha}(x^\alpha - x^\beta) = x^{v-(\alpha-\beta)}$.
- Testing whether or not one initial terms divides an other initial term is done in the same way as in the saturating division algorithm 2.5.

Notice that the only binomial ever appearing is the one from the input. Consequently we do not need to know the term order but only which monomial in the input binomial is initial.

Buchberger's algorithm: No more than one binomial is needed since the input from the flip algorithm contains at most one binomial and the division algorithm does not produce new binomials. The S-polynomial of two monomials is zero. The S-polynomial of a monomial and the binomial is the monomial given by: $S(x^v, x^\alpha - x^\beta) = \frac{x^{\alpha \vee v}}{x^v} x^v - \frac{x^{\alpha \vee v}}{x^\alpha}(x^\alpha - x^\beta) = x^{\beta - \alpha + (\alpha \vee v)}$.

In contrast to the saturating Buchberger algorithm this algorithm does the same computations as the ordinary Buchberger algorithm does but it takes a few shortcuts sometimes.

7.2 Finding the facets of a Gröbner cone

The following theorem tells us how to determine which binomials in a reduced Gröbner basis of I are facet binomials.

Theorem 7.3 *Let $\prec$ be a term order and I a toric ideal homogeneous with respect to a positive grading. A binomial $x^{\alpha_i} - x^{\beta_i} \in \mathcal{G}_\prec(I) = \{x^{\alpha_1} - x^{\beta_1}, \ldots, x^{\alpha_t} - x^{\beta_t}\}$ is a facet binomial if and only if $\alpha_i - \beta_i$ cannot be written as a nonnegative linear combination of $\{\alpha_j - \beta_j | j \neq i\}$.*

This theorem follows immediately from the definition of facet binomials in section 6.6 and Farkas' lemma as it is given in theorem 3.5 of [Papadimitriou]:

Theorem 7.4 *Farkas' lemma*
Let $v, v_1, v_2, \ldots, v_t \in \mathbb{R}^n$. The following two statements are equivalent:

- *v is a non-negative linear combination of $v_1, \ldots, v_t$*
- *for vectors $x \in \mathbb{R}^n : v_1 \cdot x \geq 0 \wedge \ldots \wedge v_t \cdot x \geq 0 \Rightarrow v \cdot x \geq 0$*

To prove this theorem [Papadimitriou] uses the termination of the simplex algorithm for solving linear programming problems (see [Papadimitriou]). Inspired by [Grünbaum] (page 11) we choose another approach:

Proof. It is easy to see that the first statement implies the second statement. The other way around is harder and we will not prove it in the general case but only under the assumption that the vectors $v, v_1, v_2, \ldots, v_t$ are in $\mathbb{R}^{n-1} \times (\mathbb{R}_{>0})$. In this case by a projection the Farkas' lemma states that the two following statements equivalent (for $v, v_1, v_2, \ldots, v_t \in \mathbb{R}^{n-1}$):

- v is a convex linear combination of $v_1, \ldots, v_t$
- for vectors $(x, y) \in \mathbb{R}^{n-1} \times \mathbb{R} : v_1 \cdot x \geq y \wedge \ldots \wedge v_t \cdot x \geq y \Rightarrow v \cdot x \geq y$

The second statement says that no affine hyperplane separates v from $v_1, \ldots, v_t$. It remains to be shown that the second statement implies the first statement. That is, if v is not in $conv\{v_1, \ldots, v_t\}$ then there exists a separating plane. The distance to v assumes a minimum value in the set $conv\{v_1, \ldots, v_t\}$ in a point v' since the set is compact. $v' - v$ is not zero since the minimal distance is not zero. The affine plane with normal $v' - v$ and with equal distance to v and v' separates v from the set: In case $v = 0$ that is $v' \cdot (u - \frac{1}{2}v') \geq 0$ for $u \in conv(v_1, \ldots, v_t)$. Suppose this was false for some $u \in conv(v_1, \ldots, v_t)$. Moving along the line $u(t) = tu + (1-t)v'$ inside $conv(v_1, \ldots, v_t)$ we get $\frac{du(t) \cdot u(t)}{dt} = 2(u \cdot u + v' \cdot v' - 2u \cdot v')t - 2(v' \cdot v' - u \cdot v')$. With $-2(v' \cdot v' - u \cdot v') = 2((u - \frac{1}{2}v') \cdot v' - \frac{1}{2}v' \cdot v') < 0$ and thereby $\frac{du(t) \cdot u(t)}{dt} < 0$ for $t > 0$ sufficiently close to 0 contradicting that v' was the closest point to $0 = v$ in $conv(v_1, \ldots, v_t)$. If $v \neq 0$ we translate the vectors by $-v$ and repeat the argument. □

The reason that this weak version of Farkas' lemma suffices is that the binomials in theorem 7.3 are known to be ordered with respect to some term order and therefore also with respect to some vector. Hence, the vectors $\alpha_1 - \beta_1, \ldots, \alpha_t - \beta_t$ are all located in an open half space and the weak Farkas' lemma applies.

Theorem 7.3 allows us to test if a binomial in a reduced Gröbner basis is a facet binomial by deciding if a linear programming problem is feasible. How to do this is explained in [Papadimitriou]. I will not go into details but just mention that my program uses the implementation of the simplex algorithm given in [Huber,..].

7.3 Traversing the graph

Let each vertex in the graph of I be represented by a reduced Gröbner basis. Given a vertex we use theorem 7.3 and linear programming to determine its edges and for each edge we use the algorithm flip 7.1 to compute the adjacent vertex. Later we will see that the graph is connected implying that given just one reduced Gröbner basis we can compute the entire graph (Gröbner fan) by a simple graph traversal:

Algorithm 7.5 *Traversal by exhaustive search*
Input: *A reduced Gröbner basis $\mathcal{G}_0$ for a toric ideal I.*
Output: *All reduced Gröbner bases for I stored in S.*
$S := \emptyset$;
$R := \{\mathcal{G}_0\}$;
while$(R \neq \emptyset)$
{
    Choose a Gröbner basis $\mathcal{G}$ from R;
    $R := R \backslash \{\mathcal{G}\}$;
    Compute the facet binomials of $\mathcal{G}$;
    for each facet binomial f of $\mathcal{G}$ do
    {
        $\mathcal{G}' := flip(\mathcal{G}, f)$;
        if$(\mathcal{G}' \notin S \cup R) R := R \cup \{\mathcal{G}'\}$;
    }
    $S := S \cup \{\mathcal{G}\}$;
}

We keep "active" vertices in R and vertices which we are done with in S. Notice that after having computed $\mathcal{G}'$ we check that it is not already in the set $R \cup S$. In practice this method is not useful for Gröbner fans with many (more than 100.000) cones as storing the set will take up a lot of RAM. We will give a better algorithm for traversing the Gröbner fan in the following sections.

7.4 The toric Gröbner walk

The theory has another nice application. Given a single reduced Gröbner basis $\mathcal{G}$ for a toric ideal I we may find a reduced Gröbner basis with respect to a term order $\prec$ by walking in the graph of I. This is called the Gröbner walk:

Algorithm 7.6 *Toric Gröbner walk*
Input: *A term order $\prec$ and any reduced Gröbner basis $\mathcal{G}$ for a toric ideal I homogeneous with respect to a positive grading.*
Output: *The reduced Gröbner basis $\mathcal{G}'$ for I with respect to $\prec$.*
while$(\exists x^\alpha - x^\beta \in \mathcal{G} : x^\alpha \prec x^\beta)$
{
    Let $x^\alpha - x^\beta \in \mathcal{G}$ be a facet binomial satisfying $x^\alpha \prec x^\beta$;
    $\mathcal{G} := flip(\mathcal{G}, x^\alpha - x^\beta)$;
}
$\mathcal{G}' := \mathcal{G}$;

Clearly, if this algorithm terminates we have found the reduced Gröbner basis we were looking for by proposition 6.18. But there are still two things to consider:

- The while condition only guarantees the existence of a binomial not ordered with respect to $\prec$. Why is it possible to choose a facet binomial not ordered with respect to $\prec$?
- Does the algorithm terminate?

The first question is answered by the two following corollaries to Farkas' lemma:

Corollary 7.7 *Let $\mathcal{G}_\prec(I)$ be the reduced Gröbner basis of a toric ideal I with respect to a term order $\prec$. For any binomial $x^\alpha - x^\beta \in \mathcal{G}_\prec(I)$ the vector $\alpha - \beta$ can be written as a non-negative linear combination of the vectors $\{\alpha_i - \beta_i\}_i$ where $x^{\alpha_i} - x^{\beta_i}$ is a facet binomial in $\mathcal{G}_\prec(I)$ for all i.*

Proof. We will prove this by induction. Fix I and $\prec$. Let $\mathcal{F} \subset \mathcal{G}_\prec(I)$ be the set of facet binomials. Let P_m be the claim:

- There exists a subset $S \subset \mathcal{G}_\prec(I)$ of size m with S consisting of non-facet binomials such that for any binomial $x^\alpha - x^\beta \in \mathcal{G}_\prec(I)$, $\alpha - \beta$ can be written as a non-negative linear combination of vectors corresponding to the binomials in $S \cup \mathcal{F}$.

The claim is true for some m with $S = \mathcal{G}_\prec(I) \backslash \mathcal{F}$. We will prove that $P_m \Rightarrow P_{m-1}$ and thereby that P_0 is true being exactly the corallary.

So let P_m be true. Pick $x^a - x^b \in S$. Since $x^a - x^b$ is not facet Farkas' lemma implies that $a - b$ can be written as a non-negative linear combination of vectors corresponding to binomials in $\mathcal{G}_\prec(I) \backslash \{x^a - x^b\}$:

$$(1) \qquad a - b = \sum_i \gamma_i'(\alpha_i' - \beta_i')$$

Using P_m we may substitute:

$$(2) \qquad a - b = \gamma(a - b) + \sum_i \gamma_i(\alpha_i - \beta_i)$$

so that $x^{\alpha_i} - x^{\beta_i} \in \mathcal{F} \cup S \backslash \{x^a - x^b\}$, $\gamma_i \geq 0$, $\gamma \geq 0$. We cannot have $\gamma > 1$ as this would contradict the binomials being ordered with respect to a vector in $\mathbb{R}^n$. Suppose $\gamma = 1$ then for the same reason $\gamma_i = 0$ for all i. Hence, in the sum (1) the vectors $\alpha_i' - \beta_i'$ can only be positive linear combinations of $a - b$. An argument we have seen earlier gives that these can only be $a - b$ since $\mathcal{G}_\prec(I)$ is reduced. This is a contradiction as $x^a - x^b$ is not in $\mathcal{G}_\prec(I) \backslash \{x^a - x^b\}$ containing the $x^{\alpha_i'} - x^{\beta_i'}$'s.

γ must be less than 1. Finally rewriting (2) we get that $a - b$ is a non-negative linear combination of vectors corresponding to binomials in $\mathcal{F} \cup S \backslash \{x^a - x^b\}$.

To see that P_{m-1} is true we first use P_m to write $\alpha - \beta$ as a non-negative linear combination of vectors corresponding to binomials in $\mathcal{F} \cup S$. Next we

substitute $a-b$ using the final non-negative linear combination above giving a non-negative linear combination of vectors only corresponding to binomials in $\mathcal{F} \cup (S \backslash \{x^a - x^b\})$. $\square$

Corollary 7.8 *Let I be a toric ideal homogeneous with respect to a positive grading. Let $\mathcal{G}$ be any reduced Gröbner basis for I and $\prec$ any term order. If for all facet binomials $x^{\alpha_i} - x^{\beta_i} \in \mathcal{G}$, $x^{\beta_i} \prec x^{\alpha_i}$ then for any binomial $x^\alpha - x^\beta \in \mathcal{G}$, $x^\beta \prec x^\alpha$.*

Proof. Let $x^\alpha - x^\beta \in \mathcal{G}$. By the corollary we may write $\gamma(\alpha-\beta) = \sum_i \gamma_i(\alpha_i - \beta_i)$ where $\gamma \in \mathbb{N}_{>0}, \gamma_i \in \mathbb{N}$ and $x^{\alpha_i} - x^{\beta_i}$ is a facet binomial in $\mathcal{G}$ and thereby, $x^{\beta_i} \prec x^{\alpha_i}$ for all i. That is, $\gamma\alpha + \sum_i \gamma_i\beta_i = \gamma\beta + \sum_i \gamma_i\alpha_i$, implying $(x^\alpha)^\gamma \prod_i (x^{\beta_i})^{\gamma_i} = (x^\beta)^\gamma \prod_i (x^{\alpha_i})^{\gamma_i}$. $x^{\beta_i} \prec x^{\alpha_i}$ implies $\prod_i (x^{\beta_i})^{\gamma_i} \prec \prod_i (x^{\alpha_i})^{\gamma_i}$. If $x^\alpha \prec x^\beta$ then $(x^\alpha)^\gamma \prec (x^\beta)^\gamma$ contradicting $(x^\alpha)^\gamma \prod_i (x^{\beta_i})^{\gamma_i} = (x^\beta)^\gamma \prod_i (x^{\alpha_i})^{\gamma_i}$. Hence $x^\beta \prec x^\alpha$ or $x^\alpha = x^\beta$. The monomials cannot be equal since $x^\alpha - x^\beta$ is in the reduced basis. We must have $x^\beta \prec x^\alpha$. $\square$

Remark 7.9 Corollary 7.7 has another application. It states that the inequalities coming from facet binomials are the only one necessary to describe the Gröbner cone. Using proposition 5.13 we see that any facet of the Gröbner cone is given by a facet binomial.

To answer the second question recall that there are only finitely many monomial initial ideals of the ideal I_A (proposition 6.6). If we can show that we cannot return to a vertex that we have already been at we know that we cannot walk forever and algorithm 7.6 must terminate. To prove that we cannot cycle in algorithm 7.6 we need the fact that the Gröbner fan is the normal fan of the state polytope. To make the argument clear we let $S_\prec(I)$ denote the graph of I with the edges oriented in the following way: Let $\mathcal{G}$ be a vertex in $S_\prec(I)$ and let $x^\alpha - x^\beta$ be a facet binomial in $\mathcal{G}$. $\mathcal{G}$ is connected to $flip(\mathcal{G}, x^\alpha - x^\beta)$ if $x^\alpha - x^\beta$ is not ordered with respect to $\prec$. That is, if $x^\alpha \prec x^\beta$. When doing the Gröbner walk we walk along oriented paths in $S_\prec(I)$. An example is given in section 7.6.

Proposition 7.10 *$S_\prec(I)$ is acyclic.*

Before we prove this proposition we have to show the following lemma:

Lemma 7.11 *Let $\mathcal{G}_\prec(I)$ and $\mathcal{G}_{\prec'}(I)$ be connected in the graph of I. That is, $\mathcal{G}_{\prec'}(I) = flip(\mathcal{G}_\prec(I), x^\alpha - x^\beta)$ for some facet binomial $x^\alpha - x^\beta \in \mathcal{G}_\prec(I)$. The corresponding vertices $v, v' \in state(I) \subset \mathbb{R}^n$ satisfy $v - v' = t(\alpha - \beta)$ for some positive $t \in \mathbb{R}$.*

Proof. By section 6.6 the corresponding Gröbner cones share a facet F contained in $H_{\alpha-\beta 0}$. By theorem 6.30 and proposition 5.22 F is the closure of the normal cone of the edge e between v and v' in $state(I)$. A vector $u \in \mathcal{N}_e(state(I))$

satisfies $face_u(state(I)) = e$. Hence, for any vector $u \in F$, $u \cdot v = u \cdot v'$ as $v, v' \in e$. This means that both $\alpha - \beta$ and $v - v'$ belong to $F^\perp$. Since $F^\perp$ is one dimensional we get that $v - v' = t(\alpha - \beta)$ for some $t \in \mathbb{R}$.

To show that t is positive let $\omega \in \mathbb{R}^n$ be a vector such that $in_\omega(I) = in_\prec(I)$. We may assume that ω is positive by corollary 6.15. $in_{\prec_\omega}(I) = in_\prec(in_\omega(I)) = in_\prec(in_\prec(I)) = in_\prec(I) \Rightarrow \mathcal{G}_\prec(I) = \mathcal{G}_{\prec_\omega}(I)$. By remark 6.13 the initial form of an element in $\mathcal{G}_{\prec_\omega}(I)$ with respect to ω is the initial term with respect to $\prec$. Hence, $\omega \cdot (\alpha - \beta) > 0$ and ω is in the interior of the Gröbner cone of $\mathcal{G}_\prec(I)$ or equivalently the normal fan of v. $face_\omega(state(I)) = v$. Hence, $\omega \cdot (v - v') > 0$. $\omega \cdot (v - v') = \omega \cdot (t(\alpha - \beta)) = t\omega \cdot (\alpha - \beta) > 0$ and we conclude that $t > 0$. □

We return to the proof of the proposition:

Proof. Suppose there was a cycle with vertices $v_0, v_1, \ldots, v_k$ where $v_k = v_0$ and $v_j \in \mathbb{R}^n$ for all j. Let $x^{\alpha_j} - x^{\beta_j}$ be the facet binomial in the reduced Gröbner basis corresponding to v_j and connecting v_j to v_{j+1}. By lemma 7.11 $0 = \sum_{j=0}^{k-1} v_j - v_{j+1} = \sum_{j=0}^{k-1} t_j(\alpha_j - \beta_j)$ with $t_j \in \mathbb{R}_{>0}$. In fact $t_j \in \mathbb{Q}$ since $\alpha_j, \beta_j, v_j \in \mathbb{Z}^n$. Hence, rewriting we get $0 = \sum_{j=0}^{k-1} n_j(\alpha_j - \beta_j)$ with $n_j \in \mathbb{N}_{>0}$. For all j we know that $x^{\alpha_j} \prec x^{\beta_j}$ and therefore $x^{n_j\alpha_j} \prec x^{n_j\beta_j}$. But then $\prod_{j=0}^{k-1} x^{n_j\alpha_j} \prec \prod_{j=0}^{k-1} x^{n_j\beta_j}$ contradicting $\sum_{j=0}^{k-1} n_j\alpha_j = \sum_{j=0}^{k-1} n_j\beta_j$. □

7.5 The reverse search tree method

When the state polytope has many vertices the exhaustive search algorithm 7.5 could use up all memory (RAM) on a computer system since it has to store all previously computed Gröbner bases. The computer would have to swap its memory to disk. This could slow down the computation since the algorithm has to compare a newly computed basis to previously computed bases. In this section an algorithm which does not have to store all the vertices (Gröbner bases) is presented. It is based on the Gröbner walk from the previous section.

Fix a term order $\prec$. We will study the paths the Gröbner walk walks along when searching for $\mathcal{G}_\prec(I)$. We have already seen that the paths are paths in the oriented graph $S_\prec(I)$. The Gröbner walk is a nondeterministic algorithm meaning that we might not take the same path in another run. Making the algorithm deterministic in each vertex we see that the paths of the Gröbner walk form a tree. Making the algorithm deterministic can be done in the following way: when we have to choose a facet binomial there could be several choices of facet binomials not ordered with respect to $\prec$. Notice that the terms with coefficient 1 are distinct since the basis is reduced with respect to the current term order. Picking the largest one with respect to the lexicographic term order makes the algorithm deterministic in each vertex. The paths we walk along searching for $\mathcal{G}_\prec(I)$ now forms a reverse tree. We will denote this tree by $T_\prec(I)$. An example is given in section 7.6.

The root of the tree is $\mathcal{G}_\prec(I)$ since it has no outgoing edges. Any other vertex in the graph is connected to $\mathcal{G}_\prec(I)$ by a unique path.

The idea is to traverse the tree $T_\prec(I)$ starting at the root since we only have to store a very little set of vertices to keep track on which parts of the tree we have not been at. Finding the in-going edges for a vertex is done by checking for each facet binomial if the deterministic Gröbner walk run on the adjacent vertex would take us to the current vertex in its next step. This involves for each facet binomial to compute the adjacent reduced Gröbner basis and some of its facet binomials.

The final algorithm for traversing the edge graph of $state(I)$:

Algorithm 7.12 *Traversal by reverse search*
Input: *A term order $\prec$ and the corresponding reduced Gröbner basis $\mathcal{G}_0$ for a toric ideal I homogeneous with respect to a positive grading.*
Output: *This algorithm runs through $T_\prec(I)$ and stores all reduced Gröbner bases of I in S. Each reduced Gröbner basis is inserted into S only once.*

$S := \emptyset$;
$R := \{\mathcal{G}_0\}$;
while$(R \neq \emptyset)$
{
  Choose a Gröbner basis $\mathcal{G} \in R$;
  $R := R \backslash \{\mathcal{G}\}$;
  $S := S \cup \{\mathcal{G}\}$;
  compute all facet binomial in $\mathcal{G}$ *and store them in the set* F;
  for all facet binomials $x^\alpha - x^\beta \in F$ *do*
  {
    if$(x^\beta \prec x^\alpha)$
    {
      $\mathcal{G}' = flip(\mathcal{G}, x^\alpha - x^\beta)$;
      compute all facet binomials in $\mathcal{G}'$ *and store them in* F';
      if$(\forall x^{\alpha'} - x^{\beta'} \in F' : x^{\beta'} \prec x^{\alpha'} \vee x^\beta \geq_{lex} x^{\alpha'})$
      {
        $R := R \cup \{\mathcal{G}'\}$;
      }
    }
  }
}

The easiest way to implement this algorithm is as a recursive procedure for traversing a subtree. That is, instead of adding $\mathcal{G}'$ to the set R we call the procedure recursively on $\mathcal{G}'$. In this way the set R is given as the recursion stack and can never be larger than the tree depth.

7.6 Example

Let $A = (1\ 2\ 3\ 6) \in mat_{14}(\mathbb{Z})$ and let $\prec$ be the graded reverse lexicographic order with respect to the vector $(1, 2, 3, 6)^T$ and with $a^6 \succ b^3 \succ c^2 \succ d$. There are 12 reduced Gröbner bases for $I_A \subset k[a, b, c, d]$. $\mathcal{G}_1$ is the Gröbner basis corresponding to $\prec$.

$\mathcal{G}_0 = \{ \boxed{\boxed{b - \underline{a^2}}}_9 \ \boxed{c - \underline{a^3}}_{10} \ \boxed{d - \underline{a^6}}_{11} \ \}$

$\mathcal{G}_1 = \{ \boxed{\underline{c^2} - d}_7 \ \underline{ab} - c \ \boxed{\underline{b^2} - ac}_4 \ \boxed{\underline{a^2} - b}_2 \ \}$

$\mathcal{G}_2 = \{ \boxed{\underline{c^2} - d}_{10} \ \boxed{\underline{a^3} - c}_{11} \ \boxed{\boxed{b - \underline{a^2}}}_1 \ \}$

$\mathcal{G}_3 = \{ \boxed{\underline{b^3} - c^2}_8 \ \underline{ab} - c \ \underline{a^2} - b \ \boxed{\boxed{ac - \underline{b^2}}}_7 \ \boxed{d - \underline{c^2}}_5 \ \}$

$\mathcal{G}_4 = \{ \ \underline{b^3} - d \ \boxed{\underline{b^2 c} - ad}_5 \ \underline{c^2} - d \ \boxed{\underline{ab} - c}_6 \ \underline{a^2} - b \ \boxed{\boxed{ac - \underline{b^2}}}_1 \ \}$

$\mathcal{G}_5 = \{ \boxed{\underline{b^3} - d}_8 \ \boxed{\underline{c^2} - d}_3 \ \underline{ab} - c \ \underline{a^2} - b \ ac - \underline{b^2} \ \boxed{\boxed{ad - \underline{b^2 c}}}_4 \ \}$

$\mathcal{G}_6 = \{ \boxed{\underline{b^3} - d}_9 \ \boxed{\underline{a^2} - b}_{11} \ \boxed{\boxed{c - \underline{ab}}}_4 \ \}$

$\mathcal{G}_7 = \{ \ \underline{ab} - c \ \boxed{\underline{b^2} - ac}_3 \ \boxed{\underline{a^2} - b}_{10} \ \boxed{\boxed{d - \underline{c^2}}}_1 \ \}$

$\mathcal{G}_8 = \{ \boxed{\underline{ab} - c}_9 \ \underline{a^2} - b \ ac - \underline{b^2} \ \boxed{\boxed{c^2 - \underline{b^3}}}_3 \ \boxed{d - \underline{b^3}}_5 \ \}$

$\mathcal{G}_9 = \{ \boxed{\underline{a^2} - b}_0 \ \boxed{\boxed{c - \underline{ab}}}_8 \ \boxed{d - \underline{b^3}}_6 \ \}$

$\mathcal{G}_{10} = \{ \boxed{\underline{a^3} - c}_0 \ \boxed{\boxed{b - \underline{a^2}}}_7 \ \boxed{d - \underline{c^2}}_2 \ \}$

$\mathcal{G}_{11} = \{ \boxed{\underline{a^6} - d}_0 \ \boxed{\boxed{b - \underline{a^2}}}_6 \ \boxed{c - \underline{a^3}}_2 \ \}$

The facet binomials are marked with a box and the subscript tells which Gröbner basis is adjacent with respect to this facet binomial. The edge graph of the state polytope is planar since the state polytope is a three dimensional polytope in $\mathbb{R}^4$:

To get the oriented graph $S_\prec(I)$ we order the edges corresponding to facet binomials with respect to $\prec$. For each binomial in the list the leading term with respect to $\prec$ is underlined telling us which direction of the edge to keep.

After this we delete edges to get the tree $T_\prec(I)$. The remaining edges correspond to the binomials in the list marked with a double box. For each reduced Gröbner basis this is the facet binomial (among facet binomials corresponding to outgoing edges) with largest initial term with respect to the lexicographic term order.

8 Computational experience

The algorithms described have all been implemented in C++. For finding the facet binomials the simplex algorithm implementation in [Huber,..] was used. For further details on the program see appendix C.

In this section we will compare the various methods for computing a reduced Gröbner basis for a toric ideal and the various methods for traversing the graph of a toric ideal by running the algorithms on some examples. Most effort was put into experimenting with the traversal algorithms and bringing their running times down. We will compare my implementation with the one by [Huber,..]. Finally we will see in which parts of the reverse search algorithm most time is spent.

8.1 Timing examples

The timing experiments were done on a 1333 MHz AMD Athlon processor with 512 MB RAM. The following set of matrices was used. Some of them also appeared in [Huber,..] allowing us to compare the running times.

Pent: $\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 1 & 2 & 1 \end{pmatrix}$

V23: $\begin{pmatrix} 2 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 2 \end{pmatrix}$

gti: $\begin{pmatrix} 20 & 24 & 25 & 31 \end{pmatrix}$

PV33: $\begin{pmatrix} 3 & 2 & 2 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 & 0 & 3 & 2 & 1 & 0 \\ 0 & 0 & 1 & 0 & 2 & 0 & 1 & 2 & 3 \end{pmatrix}$

K5: $\begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$

K6: $\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}$

A2: $\begin{pmatrix} 1 & 2 \end{pmatrix}$

A3: $\begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$

A4: $\begin{pmatrix} 1 & 2 & 3 & 4 \end{pmatrix}$

A5: $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \end{pmatrix}$

A6: $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \end{pmatrix}$

A7: $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{pmatrix}$

A8: $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{pmatrix}$

A9: $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{pmatrix}$

A10: $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{pmatrix}$

A11: $\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 \end{pmatrix}$

HA3: $\begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \end{pmatrix}$

HA4: $\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \end{pmatrix}$

HA5: $\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 \end{pmatrix}$

HA6: $\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 \end{pmatrix}$

HA7: $\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{pmatrix}$

HA8: $\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{pmatrix}$

HA9: $\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{pmatrix}$

HM: $\begin{pmatrix} 247 & 248 & 345 & 15 \end{pmatrix}$

Rm denotes a random matrix with entries in the range $0, 1, \ldots, m-1$.

For computing a single reduced Gröbner basis six different algorithms were used. The timing results are listed in the following table:

A	d	n	kd	$\lvert G_1\rvert$	$\lvert G_2\rvert$	Alg.1	Alg.1b	Alg.2	Alg.2b	BB	GW
PV33	3	9	6	23	18	0.00	0.01	0.02	0.01	0.01	0.03
K5	5	10	5	10	11	0.00	0.01	0.02	0.02	0.02	0.02
K6	6	15	9	30	36	0.01	0.01	0.14	0.12	0.13	0.15
A2	1	2	1	1	1	0.00	0.00	0.00	0.00	0.00	0.00
A3	1	3	2	4	3	0.00	0.00	0.00	0.00	0.00	0.00
A5	1	5	4	14	10	0.01	0.00	0.00	0.00	0.00	0.00
A7	1	7	6	29	21	0.00	0.00	0.00	0.01	0.01	0.01
A9	1	9	8	48	36	0.01	0.00	0.01	0.01	0.01	0.02
A10	1	10	9	59	45	0.00	0.00	0.01	0.01	0.01	0.05
HA3	2	3	1	1	1	0.00	0.00	0.00	0.00	0.00	0.00
HA4	2	4	2	3	3	0.00	0.01	0.00	0.00	0.00	0.00
HA5	2	5	3	6	6	0.01	0.00	0.00	0.00	0.01	0.00
HA7	2	7	5	15	15	0.00	0.00	0.00	0.00	0.01	0.02
HA9	2	9	7	28	28	0.00	0.00	0.01	0.01	0.02	0.04
R10	3	6	3	15	11	0.05	0.05	0.01	0.02	0.01	0.01
R10	4	6	2	18	17	0.20	0.60	0.00	0.01	0.01	0.01
R10	3	7	4	60	38	0.21	0.22	0.04	0.04	0.03	0.06
R10	3	10	7	137	134	2.20	3.99	0.82	1.11	0.88	1.74
R10	4	8	4	58	79	3.13	16.82	0.05	0.06	0.05	0.16
R100	3	6	3	104	90	183.70	946.58	0.05	0.05	0.09	0.12
R5	5	10	5	349	98	37.82	114.75	0.36	0.36	0.88	3.26
R6	5	10	5	790	426	$>$1000	$>$1000	19.63	20.92	28.75	276.90
HM	1	4	3	11	4	0.01	0.00	0.00	0.00	0.00	0.00

d is the number of rows in A and n is the number of columns. kd is the dimension of $ker(A)$ which is equal to the dimension of the state polytope. $|G_1|$ is the number of elements in the reduced Gröbner basis G_1 for I_A with respect to the lexicographic order with $x_1 \succ x_2 \succ \ldots \succ x_n$ and $|G_2|$ is the number of elements in the reduced Gröbner basis G_2 for I_A with respect to a graded reverse lexicographic order with $x_1 \succ x_2 \succ \ldots \succ x_n$. The grading is given by any of the positive vectors in the row-space of A. The timing results are given in seconds. The various algorithms are described in section 8.2.

For computing all reduced Gröbner bases of an ideal four different algorithms were used. The results are shown in the following table:

A	d	n	kd	vertices	edges	ES	RS	ES2	H
Pent	3	5	2	8	8	0.00	0.00	0.00	0.00
V23	3	6	3	29	45	0.01	0.01	0.00	0.01
gti	1	4	3	288	467	0.11	0.05	0.06	0.06
K5	5	10	5	102	255	0.07	0.04	0.05	0.05
K6	6	15	9	195720	951390	11128.00	633.04	-	707.90
A2	1	2	1	2	1	0.00	0.00	0.01	0.00
A3	1	3	2	6	6	0.00	0.00	0.00	0.00
A4	1	4	3	20	31	0.00	0.00	0.01	0.01
A5	1	5	4	114	249	0.04	0.02	0.03	0.03
A6	1	6	5	488	1394	0.37	0.17	0.21	0.21
A7	1	7	6	4073	14800	6.67	3.08	3.67	3.81
A8	1	8	7	25334	111558	71.42	34.36	40.53	42.40
A9	1	9	8	206444	1080981	-	528.43	-	-
A10	1	10	9	1499772	9105736	-	6758.00	-	-
A11	1	11	10	16379587	114202401	-	36 h	-	-
HA3	2	3	1	2	1	0.00	0.00	0.00	0.00
HA4	2	4	2	8	8	0.00	0.00	0.00	0.00
HA5	2	5	3	42	65	0.00	0.00	0.01	0.01
HA6	2	6	4	356	778	0.20	0.09	0.12	0.12
HA7	2	7	5	3079	8830	4.40	2.04	2.46	2.50
HA8	2	8	6	40284	147086	120.57	60.46	69.53	73.67
HA9	2	9	7	583371	2615804	-	2017.00	-	-
R10	4	6	2	36	36	0.02	0.01	0.01	0.01
HM	1	4	3	904	1546	0.73	0.37	0.41	0.40

"vertices" is the number of vertices in the state polytope and "edges" is the number of edges. The tested algorithms are: exhaustive search (ES) (algorithm 7.5), reverse search (RS) (algorithm 7.12) and two other algorithms ES2 and H explained in section 8.3. Except for A11 the timings are given i seconds. ES and ES2 ran out of RAM on examples A9, HA9, A10 and A11. This made the program swap to disk thereby slowing it down till it almost stalled. In example K6 the ES finished its computations despite the swapping but it clearly affected the running time. ES2 was dropped on this example as I was getting impatient.

8.2 Computing a single reduced Gröbner basis

We have presented two algorithms for computing generators for toric ideals. The first one was algorithm 2.2 denoted "Alg.1" in the first table in the previous section. The second one was algorithm 3.18 denoted "Alg.2". There were a few open questions in these algorithms. First of all, in algorithm 2.2 there were several choices for the elimination term order. The column denoted "Alg.1" shows the running times when using the lexicographic term order thereby computing G_1. Computing with this order is usually slow. To improve the running time I

changed the term order to the one defined in remark 2.3 with $\prec'$ being a graded reverse lexicographic order. This term order has the advantage of being graded reverse lexicographic on the two subsets of variables. The resulting Gröbner basis is G_2. The running times are listed in column "Alg.1b". The graded reverse lexicographic order is usually faster than the lexicographic one. Surprisingly this did not improve the running times as expected.

The second algorithm "Alg.2" for computing G_2 was expected to be faster than "Alg.1" since "Alg.2" did not introduce new variables and computed Gröbner bases with respect to graded lexicographic orders. The experiments confirm this on most examples. On large examples the difference is huge.

The LLL-algorithm was used two times in "Alg.2". The first time to compute the lattice kernel of A and a second time to reduce the generators for the lattice kernel. The second run of the LLL-algorithm is not necessary but was introduced in hope for smaller exponents in the binomials and thereby faster running times. The column "Alg.2b" shows timings for the algorithm when the second LLL-reduction is left out. The table shows that "Alg.2b" is faster than "Alg.2" on some of our examples. One thing to notice is that the LLL-implementation was not optimised as well as suggested in [Lenstra,..] (see remark 4.10). This could affect the comparison between "Alg.1" and "Alg.2" (and also the comparison between "Alg.2" and "Alg.2b"). This might be the case in example K6 where it turns out that most time is spent during the LLL-reductions.

The Gröbner walk procedure was introduced in order to construct the search tree $T_\prec(I)$. In the literature an other reason for introducing it is for quickly changing one Gröbner basis for a general polynomial ideal into a Gröbner basis with respect to another term order. The two last columns compare the Gröbner walk procedure to Buchberger's algorithm for transforming G_2 into G_1. The timings include the time for computing G_2 using algorithm "Alg.2". The table shows that the saturating Buchberger algorithm usually is faster at this on our examples. One reason for this could be that the implementation of the Gröbner walk walks rather randomly in the state polytope towards G_1. It is possible that there is a better strategy for choosing facet binomials that would make the search paths shorter. Compared to "Alg.1" the Buchberger algorithm and the Gröbner walk are usually faster at computing G_1.

The conclusion must be that there still are many questions to be answered. What is known about computing Gröbner bases for general polynomial ideals efficiently in general is not necessarily true for toric ideals. (For example the saturating Buchberger algorithm seems to outperform the Gröbner walk.) However, three general observations for toric ideals are done:

- Algorithm 3.18 is usually faster than algorithm 2.2
- Even computing a Gröbner basis with respect to a lexicographic order using algorithm 3.18 combined with the Gröbner walk or the Buchberger algorithm is usually faster than algorithm 2.2

- It is sometimes useful to apply the LLL-reduction a second time in algorithm 3.18

8.3 Computing all reduced Gröbner bases

Two methods were presented for computing the Gröbner fan of a toric ideal. Let us compare the two methods. The ES algorithm

1. does two flip operations for every edge in the graph.
2. searches the set of previously computed Gröbner bases.
3. determines all facet binomials for every reduced Gröbner basis once.

The RS algorithm

1. does one flip operation for every edge in the graph.
2. determines at least one facet binomials after each of these flip operations.
3. determines all (correctly ordered) facet binomials of the computed Gröbner fan if the edge leading to the compute Gröbner basis is in the tree $T_\prec(I)$. That is, it determines all correctly ordered facet binomials of each vertex.

One obvious improvement of ES is orienting the graph to cut down the number of flips and searches in the set of previously computed bases. This cuts down the running time as shown in column ES2.

The choice is between the ES2 algorithm doing a lot of searching when the graph is large and the RS algorithm determining a lot of facet binomials. On small examples they are almost equally fast. On large examples however ES2 uses too much memory to run on the test computer or spends to much time searching (K6).

It would be nice if we also were able to cut down the memory used to store the set of computed Gröbner bases. This can be done by deleting a Gröbner basis when all the in-going edges have been traversed. This will also cut down the time used for searching. An attempt to do this is algorithm H. Unfortunately this introduced slightly more complex data structures and did not cut down memory usage to an acceptable level. This is because the set containing the previously computed Gröbner bases still has to be large when the kernel dimension of A is high. Again, too much space is used and too much time is spent searching. Example K6 is very good at illustrating the differences among the algorithms. On this example ES and ES2 start swapping while H does not.

The reverse search algorithm is clearly the best. Its memory usage is below 2 MB on all examples.

8.4 Compared to TiGERS

TiGERS is the implementation of the reverse search and exhaustive search algorithms written in C by [Huber,..]. My implementation and their implementation are very much alike. They both use the flip algorithm and the same traversal algorithms and they use the same simplex algorithm implementation. In addition they both use a criterion based on the following remark to easily rule out the possibility of a binomial being a facet binomial.

Remark 8.1 Let $\mathcal{G} = \{x^{a_1} - x^{b_1}, \ldots, x^{a_t} - x^{b_t}\}$ be a reduced Gröbner basis for a toric ideal I. If $x^{a_1} - x^{b_1}$ is a facet binomial then by proposition 6.10 $\{x^{a_1} - x^{b_1}, x^{a_2}, \ldots, x^{a_t}\}$ is a Gröbner basis for the ideal $in_c(I)$ (using the notation introduced in section 6.6). By the Buchberger's S-criterion it is easy to check if this is a Gröbner basis since the initial terms are known. So if the answer is no then $x^{\alpha_1} - x^{\beta_1}$ cannot be facet. This dramatically reduces the number of possible facet binomials. A set of 60 binomials will typically reduce to 10. This is very good not only because we have to call the simplex algorithm less often but also because the size of the linear programming problems is smaller. It should be noticed that this test does take some time to perform.

Despite the similarities the running times differ with a factor 10-120 with the factor increasing when the size of the problem increases. Listed in order of importance are the reasons why my implementation is faster:

- The C++ programming language offers a lot of easy-to-use data structures. The implementation of these is efficient. Especially, this was useful for efficient lookup to see if a Gröbner basis had already been seen in the exhaustive search and for sorting elements of a Gröbner basis before comparing with another one. In TiGERS not all structures were efficiently implemented possibly because it is a lot of work doing in C.

- The timings of TiGERS in [Huber,..] were done in 1999 on a 450 MHz Intel Pentium processor. Today computers are approximately 3-4 times faster. The tests of my implementation were done on a 1333 MHz AMD Athlon.

- When testing whether two monomials are ordered with respect to the graded reverse lexicographic term order it is worth noticing that the monomials compared always are terms in a binomial being homogeneous with respect to the row-space of A. Hence, the degree test can be omitted and we only have to test "reverse lexicographically". Usually this means that we only have to look at very few of the entries in the vector defining the binomial. TiGERS does not take advantage of this.

- Since the division algorithm is used a lot we are interested in efficiently determining if the initial term of one polynomial divides the initial term

of an other polynomial. As suggested by [Hosten,..] we store with each polynomial (monomial or binomial) a bit-vector of n-bits describing the support of the vector corresponding to the initial term. Storing the bit vector in a word it is fast to check if the support of one initial term is contained in the support of the initial term of an other polynomial using boolean arithmetics. This is a necessary condition for the first initial term to divide the second.

- TiGERS does not take full advantage of the structure of toric ideals. For example it uses two vectors for each binomial even though the binomials (at least when doing the graph traversal) are pure and therefore can be represented by a single vector.

Even though the list above is ordered it is hard to tell how much a single trick improves the running time as this depends on the order in which the tricks are added to the program.

All together this results in a factor 120 on the largest examples. Probably a factor 30-40 if run on a 450 MHz processor. The faster implementation allows us to compute the Gröbner fan of ideals (I_{A10} and I_{A11}) which [Huber,..] had to drop.

8.5 Where is the time spent?

Using a performance analyser program it was possible to find the parts of the program in which most time was spent. In example A8 using the reverse search algorithm the performance analyser program gave the following result:

Simplex algorithm	25 %
Memory management	19 %
Data structures	11 %
Vector/binomial class	8 %
Division algorithm	12 %
Flip algorithm	22 %
Total	97 %

The listed program parts are the ones where most time was spent. Since the simplex algorithm is an isolated part of the program depending on no other parts the 25 % is a reliable estimate for the time spent in this algorithm. The rest of the listed subroutines are parts of either the flip algorithm or the tree traversal algorithm. The import thing to notice is that no single routine takes a large part of the time. Hence, we cannot improve the running time dramatically without doing some major changes.

8.6 Reliability

The program gets the same results as TiGERS does on the common examples. Most parts of the program are purely combinatorial and the probability for an error in the implementation in these parts is small. However, the LLL-algorithm and the simplex algorithm use floating-point arithmetics. It cannot be ruled out that especially the simplex algorithm might do a round off error on some input data leading to a wrong result. On the examples listed this was not detected. Even though an error has not been detected it is possible that an error occurred – the increase from the size of $state(I_{A10})$ to the size of $state(I_{A11})$ sure looks suspicious. One way to get around these errors is by using a library doing calculations as fractions. This has not been done. I expect TiGERS to have the same limitations.

9 Postscript

The toric ideals were defined and their basic properties were discovered. Two algorithms for computing them were presented. We had to introduce the ideal quotients and the LLL-reduction techniques for the second algorithm.

After an introduction to polyhedral theory the Gröbner fan was defined for positive-homogeneous ideals. For toric ideals we saw that there was a bijection between the n-dimensional Gröbner cones and the reduced Gröbner bases. For toric ideals this lead us to the definition of the graph of I whose vertices were reduced Gröbner bases. In general the Gröbner fan is the normal fan of a polytope called the state polytope of I. We did not prove this but referred to [Sturmfels]. This connection told us that the graph of I was isomorphic to edge graph of the state polytope of I.

To walk along edges in the edge graph the flip algorithm was introduced. Combined with the simplex algorithm this gave a method for traversing the graph of the ideal. In this way an exhaustive search could be used to compute all reduced Gröbner bases since the graph was connected. A reverse search method for traversing the graph was presented with the advantage that it did not need to store the entire graph.

The algorithms were implemented and tested on some examples. It was difficult to say something general about the time spent on computing the first reduced Gröbner basis. The effort was but into making the traversal algorithms fast. Finally we saw how a few tricks could improve the speed. Among those were some taking advantage of the structure of toric ideals. Other improvements came from computer science.

A Sources

In this section I will briefly describe where the ideas and material for the various parts came from. The references given are not to where the ideas original came from but to where I got them. I have presented the material in a way I find suitable for our purpose. And I have given the proofs the way I understand them. Sometimes lemmas and propositions were added to fill out some gaps.

In section 1 the group ring is a standard construction in algebra. The properties of toric ideals in section 1.3 and their proofs were given in [Sturmfels]. The hint on the back of [Sturmfels] : "They (toric ideals) are characterized as those prime ideals that are generated by monomial differences" lead to proposition 1.13 and the proof in the following section.

The method in section 2 for computing generators of a toric ideal was given in [Sturmfels] without proof. Representing binomials by vectors was suggested several places in the literature. [Bigitta,..] define the saturating remainder and the saturating S-polynomial formally and leaves many proofs to the reader. My covering of this has very little in common with [Bigitta,..]. I have tried to describe the saturating division algorithm, the saturating Buchberger algorithm and their applications in detail.

Saturating ideals to compute generators was suggested by [Hosten,..]. Section 3 consists of propositions given by [Bigitta,..] and [Sturmfels]. The definitions and properties of homogeneous ideals are standard.

Section 4 is based on [Lenstra,..] and [Cohen]. The proofs are slightly different but essentially the same as in [Lenstra,..] and [Cohen]. The application for computing lattice bases of integer kernels was given in [Cohen].

Section 5 was originally based on [Grünbaum] but changed to fit the application in a less theoretical way, thereby introducing material from [Sturmfels].

Subsections 6.1, 6.2 and 6.3 consist of various needed propositions. Some were implicitly given by [Sturmfels] and some had incomplete proofs in [Sturmfels]. [Huber,..] claimed the existence of the bijection between the reduced Gröbner bases of a toric ideal and its initial ideals. The construction of the Gröbner fan and the state polytope was given in [Sturmfels]. For the description of the toric Gröbner cone I was inspired by [Huber,..].

Section 7 was based on [Huber,..]. Some gaps were filled out here.

B Notation

- $\mathbb{N} = \{0, 1, 2, \ldots\}$
- $A_> = \{a \in A | a > 0\}$
- $A \subset B$: A is a subset of B
- $a \prec b$: a is strictly less than b with respect to the term order $\prec$
- $k[\mathbf{x}] = k[x_1, \ldots, x_n]$
- for $v \in \mathbb{Z}^n$: $x^v = \prod_{i=1}^n x_i^{v_i}$
- for polynomial p and a term order $\prec$: $in_\prec(p)$ is the initial term of p.
- for polynomial p and a term order $\prec$: $tail_\prec(p) = p - in_\prec(p)$
- $in_\prec(\mathcal{G}) = \langle in_\prec(g_1), \ldots, in_\prec(g_t)\rangle$ where $\mathcal{G} = \{g_1, \ldots, g_t\}$
- $\mathcal{G}_\prec(I)$ the reduced Gröbner basis for I with respect to the term order $\prec$
- for $v \in \mathbb{Z}^n$: $v^+ \in \mathbb{N}^n$, $v_i^+ = \max(v_i, 0)$
- for $v \in \mathbb{Z}^n$: $v^- \in \mathbb{N}^n$, $v_i^- = -\min(v_i, 0)$
- observe $v = v^+ - v^-$
- for $u, v \in \mathbb{Z}^n$: $u \wedge v \in Z^n$, $(u \wedge v)_i = \min(u_i, v_i)$
- for $u, v \in \mathbb{Z}^n$: $u \vee v \in Z^n$, $(u \vee v)_i = \max(u_i, v_i)$
- for $v \in \mathbb{Z}^n$: $p_v = x^{v^+} - x^{v^-}$
- $mat_{dn}(A)$ the set of matrices with entries in A with d rows and n columns
- $conv(U) = \{a_1u_1 + \ldots + a_tu_t | t \in \mathbb{N}, 0 \leq a_j \in \mathbb{R}, u_j \in U, a_1 + \ldots + a_t = 1 \text{ for all } j\}$ for $U \subset \mathbb{R}^n$

Algorithms are written with C-style control statements like “for”, “if” and “while”. “{” and “}” are used for grouping statements together allowing them to act as a single (sub)statement in a control statement. “ :=” is used for assignments.

C A tiny user manual

The program is available at http://www.daimi.au.dk/ ~u950710/speciale.html. It should compile on any UNIX/LINUX platform. The program is compiled using `make`. This results in two executable files. The first one (`main`) lets you select one of ten computation methods. Six methods for computing a single reduced Gröbner basis and four methods for computing all reduced Gröbner basis for a toric ideal. The methods are the one described in section 8.2 and in section 8.3.

After having selected the computation method you are asked if the program should LaTeX its output. Finally, you tell the program what the matrix A should be. If you want to use a random matrix you type in its dimension: number of rows, number of columns and a maximum for the nonnegative random entries (+1). In some cases this will give an ideal not homogeneous with respect to a positive vector and the behaviour of the program is undefined. If you want to specify a matrix you type in a 0 followed by the entries given in the format: `{(1,1)(1,2)(1,3)}` corresponding to the matrix HA3 in section 8.1.

After this the program starts computing. When finished the results can be read from the file `ud.tex` and `ud.dvi` if LaTeX is available. The final list of reduced Gröbner bases and how they are connected in the graph can only be printed if the exhaustive search method was chosen. The printing of the list is very slow even on medium sized examples.

The second executable file is used for generating tables containing running times and for printing the various test matrices. Right now running this program will take several hours. The output is written to `tableA.tex`, `tableB.tex` and `tableC.tex`.

References

[Bigitta,..] Anna Maria Bigatti, Robertola Scala, Lorenzo Robbiano, "Computing Toric Ideals," *J. Symbolic Computation,* **27**, (1999): 351–365.

[Cohen] Henri Cohen, "A Course in Computational Algebraic Number Theory," (Springer-Verlag, 1993): 78–100.

[Cox, Little and O'Shea] Cox, Little and O'Shea, "Ideals, Varieties and Algorithms," (Springer Verlag, 1992): 47–111.

[Grünbaum] Branko Grünbaum, "Convex polytopes," (Interscience publishers, 1967).

[Hosten,..] Serkan Hosten and Bernd Sturmfels, "GRIN: An implementation of Gröbner Bases for Integer Programming," in *Integer Programming and Combinatorial Optimization,* (Springer Lecture Notes in Computer Science, **920**, 1995): 267–276.

[Huber,..] Birkett Huber and Rekha R. Thomas, "Computing Gröbner Fans of Toric Ideals," *Experimental Mathematics, Vol. 9 (2000), No. 3*: 321–331.

[Lauritzen] Niels Lauritzen, "Algebra 1," Course Notes (Department of Mathematical Sciences, University of Aarhus, Denmark, 2000).

[Lenstra,..] A. K. Lenstra, H. W. Lenstra, L. Lovász, "Factoring Polynomials with Rational Coefficients," *Mathematische Annalen* (Springer Verlag, 1982): 515–534.

[Papadimitriou] Christos H. Papadimitriou, "Combinatorial Optimization: Algorithms and Complexity," (Prentice Hall, 1982).

[Papadimitriou 1994] Christos H. Papadimitriou, "Computational Complexity," (Addison Wesley, 1994).

[Sturmfels] B. Sturmfels, "Gröbner Bases and Convex Polytopes," (American Mathematical Society, University Lectures, 1996).